

TournamentPlanner

Generated by Doxygen 1.9.1

1 TournamentPlanner	1
2 The MIT License (MIT)	3
3 Namespace Index	5
3.1 Packages	5
4 Hierarchical Index	7
4.1 Class Hierarchy	7
5 Class Index	13
5.1 Class List	13
6 Namespace Documentation	21
6.1 TournamentPlanner Namespace Reference	21
6.2 TournamentPlanner.Algorithms Namespace Reference	21
6.3 TournamentPlanner.Algorithms.ConstructionAlgorithms Namespace Reference	21
6.4 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns Namespace Reference	21
6.5 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction Namespace Reference	21
6.6 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction Namespace Reference	22
6.7 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction Namespace Reference	22
6.8 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction Namespace Reference	23
6.9 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares Namespace Reference	23
6.10 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA Namespace Reference	23
6.11 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal Namespace Reference	24
6.12 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal Namespace Reference	24
6.13 TournamentPlanner.Algorithms.Testing Namespace Reference	24
6.14 TournamentPlanner.Algorithms.TournamentPlanners Namespace Reference	25
6.15 TournamentPlanner.Algorithms.TournamentPlanners.Spouses Namespace Reference	25
6.16 TournamentPlanner.Algorithms.TournamentPlanners.Whist Namespace Reference	25
6.17 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even Namespace Reference	26
6.18 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd Namespace Reference	26
6.19 TournamentPlanner.Algorithms.Utils Namespace Reference	26
6.20 TournamentPlanner.Datastore Namespace Reference	27
6.21 TournamentPlanner.Model Namespace Reference	27
6.22 TournamentPlanner.Model.Designs Namespace Reference	27
6.23 TournamentPlanner.Model.Designs.BlockDesigns Namespace Reference	27
6.24 TournamentPlanner.Model.Designs.LatinSquares Namespace Reference	28
6.25 TournamentPlanner.Model.Testing Namespace Reference	28
6.26 TournamentPlanner.Model.Tournaments Namespace Reference	28

6.27	TournamentPlanner.Pages Namespace Reference	29
6.28	TournamentPlanner.Pages.PageModels Namespace Reference	29
6.29	TournamentPlanner.Pages.PageModels.Designs Namespace Reference	29
6.29.1	Enumeration Type Documentation	30
6.29.1.1	Method	30
6.29.1.2	MOLSMETHOD	30
6.30	TournamentPlanner.Pages.PageModels.Testing Namespace Reference	30
6.31	TournamentPlanner.Pages.PageModels.Tournaments Namespace Reference	31
6.32	TournamentPlanner.Pages.Shared Namespace Reference	31
6.33	TournamentPlanner.Pages.Tournament Namespace Reference	31
6.34	TournamentPlanner.Utils Namespace Reference	31
6.35	TournamentPlannerTests Namespace Reference	32
6.36	TournamentPlannerTests.Algorithms Namespace Reference	32
6.37	TournamentPlannerTests.Algorithms.ConstructionAlgorithms Namespace Reference	32
6.38	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns Namespace Reference	32
6.39	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction Namespace Reference	32
6.40	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction Namespace Reference	32
6.41	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction Namespace Reference	33
6.42	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction Namespace Reference	33
6.43	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares Namespace Reference	33
6.44	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal Namespace Reference	33
6.45	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal Namespace Reference	34
6.46	TournamentPlannerTests.Algorithms.Testing Namespace Reference	34
6.47	TournamentPlannerTests.Algorithms.TournamentPlanners Namespace Reference	34
6.48	TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses Namespace Reference	34
6.49	TournamentPlannerTests.Algorithms.TournamentPlanners.Whist Namespace Reference	35
6.50	TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even Namespace Reference	35
6.51	TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd Namespace Reference	35
6.52	TournamentPlannerTests.Datastore Namespace Reference	35
6.53	TournamentPlannerTests.Designs Namespace Reference	36
6.54	TournamentPlannerTests.Model Namespace Reference	36
6.55	TournamentPlannerTests.Model.Designs Namespace Reference	36
6.56	TournamentPlannerTests.Model.Designs.BlockDesigns Namespace Reference	36
6.57	TournamentPlannerTests.Model.Designs.LatinSquares Namespace Reference	36
6.58	TournamentPlannerTests.Model.Tournaments Namespace Reference	37
6.59	TournamentPlannerTests.Utils Namespace Reference	37

7 Class Documentation

39

7.1 TournamentPlanner.Algorithms.TournamentPlanners.Balanced3PlayerTournamentPlanner Class Reference	39
7.1.1 Detailed Description	39
7.1.2 Constructor & Destructor Documentation	40
7.1.2.1 Balanced3PlayerTournamentPlanner()	40
7.2 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction Class Reference	40
7.2.1 Detailed Description	41
7.2.2 Constructor & Destructor Documentation	41
7.2.2.1 BaseBIBDConstruction()	41
7.2.3 Member Function Documentation	42
7.2.3.1 ConstructDesign()	42
7.2.4 Property Documentation	42
7.2.4.1 B	42
7.2.4.2 K	42
7.2.4.3 Lambda	43
7.2.4.4 R	43
7.2.4.5 V	43
7.3 TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign Class Reference	43
7.3.1 Detailed Description	44
7.3.2 Constructor & Destructor Documentation	44
7.3.2.1 BaseBlockDesign()	44
7.3.3 Property Documentation	44
7.3.3.1 Blocks	44
7.3.3.2 V	45
7.4 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.BaseMOLSCONSTRUCTION Class Reference	45
7.4.1 Detailed Description	46
7.4.2 Constructor & Destructor Documentation	46
7.4.2.1 BaseMOLSCONSTRUCTION()	46
7.4.3 Member Function Documentation	46
7.4.3.1 ConstructDesign()	46
7.4.3.2 ValidateOrder()	47
7.4.4 Property Documentation	47
7.4.4.1 ArgumentExceptionMessage	47
7.4.4.2 Order	47
7.5 TournamentPlanner.Pages.PageModels.BasePageModel Class Reference	48
7.5.1 Detailed Description	48
7.5.2 Property Documentation	49
7.5.2.1 DisplayResults	49
7.5.2.2 ErrorMessage	49
7.6 TournamentPlanner.Model.Tournaments.BaseTournament Class Reference	49
7.6.1 Detailed Description	50

7.6.2 Constructor & Destructor Documentation	50
7.6.2.1 BaseTournament()	50
7.6.3 Member Function Documentation	50
7.6.3.1 AddRound()	51
7.6.3.2 GameToString()	51
7.6.4 Property Documentation	51
7.6.4.1 Order	51
7.6.4.2 Rounds	51
7.7 TournamentPlanner.Algorithms.TournamentPlanners.Whist.BaseWhistPlanner Class Reference	52
7.7.1 Detailed Description	53
7.7.2 Constructor & Destructor Documentation	53
7.7.2.1 BaseWhistPlanner()	53
7.7.3 Member Function Documentation	53
7.7.3.1 PlanTournament()	53
7.7.4 Property Documentation	54
7.7.4.1 V	54
7.8 TournamentPlanner.Model.Designs.BlockDesigns.BIBD Class Reference	54
7.8.1 Detailed Description	55
7.8.2 Constructor & Destructor Documentation	55
7.8.2.1 BIBD() [1/2]	55
7.8.2.2 BIBD() [2/2]	55
7.8.3 Member Function Documentation	56
7.8.3.1 AddBlock()	56
7.8.3.2 AddBlocks()	56
7.8.4 Property Documentation	56
7.8.4.1 B	56
7.8.4.2 K	57
7.8.4.3 Lambda	57
7.8.4.4 R	57
7.9 TournamentPlanner.Pages.PageModels.Designs.BIBDModel Class Reference	57
7.9.1 Detailed Description	58
7.9.2 Property Documentation	58
7.9.2.1 BIBD	58
7.9.2.2 BlockBuildingMethod	58
7.9.2.3 K	58
7.9.2.4 L	59
7.9.2.5 V	59
7.10 TournamentPlanner.Model.Tournaments.BipartiteTournament Class Reference	59
7.10.1 Detailed Description	60
7.10.2 Constructor & Destructor Documentation	60
7.10.2.1 BipartiteTournament()	60
7.11 TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournamentModel Class Reference	60

7.11.1 Detailed Description	61
7.11.2 Property Documentation	61
7.11.2.1 TeamSize	61
7.11.2.2 Tournament	62
7.12 TournamentPlanner.Algorithms.TournamentPlanners.BipartiteTournamentPlanner Class Reference	62
7.12.1 Detailed Description	62
7.12.2 Constructor & Destructor Documentation	62
7.12.2.1 BipartiteTournamentPlanner()	63
7.12.3 Member Function Documentation	63
7.12.3.1 PlanTournament()	63
7.13 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Bose↔STSCreation Class Reference	63
7.13.1 Detailed Description	64
7.13.2 Constructor & Destructor Documentation	64
7.13.2.1 BoseSTSCreation()	64
7.13.3 Member Function Documentation	65
7.13.3.1 ConstructDesign()	65
7.14 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔BoseSTSCreationTests Class Reference	65
7.14.1 Detailed Description	65
7.14.2 Member Function Documentation	65
7.14.2.1 BoseSTSCreationTest()	66
7.15 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.CommutativeIdempotent↔LatinSquareConstruction Class Reference	66
7.15.1 Detailed Description	66
7.15.2 Constructor & Destructor Documentation	67
7.15.2.1 CommutativeIdempotentLatinSquareConstruction()	67
7.15.3 Member Function Documentation	67
7.15.3.1 ConstructDesign()	67
7.16 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Constraint↔ProgrammingMOLSPairConstruction Class Reference	67
7.16.1 Detailed Description	68
7.17 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Constraint↔ProgrammingMOLSPairConstructionTests Class Reference	69
7.17.1 Detailed Description	69
7.17.2 Member Function Documentation	69
7.17.2.1 InvalidOrderShouldThrowException()	69
7.17.2.2 OrthogonalityTest()	69
7.18 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlanner Class Reference	70
7.18.1 Detailed Description	70
7.18.2 Constructor & Destructor Documentation	70
7.18.2.1 CyclicOddWhistPlanner()	70

7.19	TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlannerTests Class Reference	71
7.19.1	Detailed Description	71
7.19.2	Member Function Documentation	71
7.19.2.1	TestWhist33Construction()	71
7.19.2.2	TestWhist57Construction()	71
7.20	TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlanner Class Reference	72
7.20.1	Detailed Description	72
7.20.2	Constructor & Destructor Documentation	72
7.20.2.1	CyclicWhistPlanner()	72
7.21	TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlannerTests Class Reference	73
7.21.1	Detailed Description	73
7.21.2	Member Function Documentation	73
7.21.2.1	CyclicWhistConstructionsTest()	73
7.22	TournamentPlannerTests.Datastore.DataReaderTests Class Reference	73
7.22.1	Detailed Description	74
7.23	TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Datastore↔MOLSCONSTRUCTION Class Reference	74
7.23.1	Detailed Description	75
7.24	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Datastore↔MOLSCONSTRUCTIONTESTS Class Reference	75
7.24.1	Detailed Description	75
7.24.2	Member Function Documentation	75
7.24.2.1	SeveralOrdersTest()	75
7.25	TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Difference↔MOLSPAIRCONSTRUCTION Class Reference	76
7.25.1	Detailed Description	77
7.26	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Difference↔MOLSPAIRCONSTRUCTIONTESTS Class Reference	77
7.26.1	Detailed Description	77
7.26.2	Member Function Documentation	77
7.26.2.1	InvalidOrderTest()	77
7.26.2.2	OrthogonalityTest()	77
7.27	TournamentPlanner.Pages.ErrorModel Class Reference	78
7.28	TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlanner Class Reference	78
7.28.1	Detailed Description	79
7.28.2	Constructor & Destructor Documentation	79
7.28.2.1	EvenEvenProductWhistPlanner()	79
7.29	TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhist↔PlannerTests Class Reference	80
7.29.1	Detailed Description	80
7.30	TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenOddProductWhistPlanner Class Reference	80

7.30.1 Detailed Description	81
7.30.2 Constructor & Destructor Documentation	81
7.30.2.1 EvenOddProductWhistPlanner()	81
7.31 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.EvenOddProductWhist↔ PlannerTests Class Reference	82
7.31.1 Detailed Description	82
7.32 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPair↔ Construction Class Reference	82
7.32.1 Detailed Description	83
7.33 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Four↔ KMOLSPairConstructionTests Class Reference	84
7.33.1 Detailed Description	84
7.33.2 Member Function Documentation	84
7.33.2.1 InvalidOrderTest()	84
7.33.2.2 OrthogonalityTest()	84
7.34 TournamentPlanner.Model.Designs.BlockDesigns.GDD Class Reference	85
7.34.1 Detailed Description	86
7.34.2 Constructor & Destructor Documentation	86
7.34.2.1 GDD() [1/5]	86
7.34.2.2 GDD() [2/5]	86
7.34.2.3 GDD() [3/5]	87
7.34.2.4 GDD() [4/5]	87
7.34.2.5 GDD() [5/5]	87
7.34.3 Member Function Documentation	88
7.34.3.1 AddBlock()	88
7.34.3.2 AddParallelClassAndBlocks()	88
7.34.3.3 GetGroupSizes()	88
7.34.4 Property Documentation	89
7.34.4.1 BlockSizes	89
7.34.4.2 Groups	89
7.34.4.3 K	89
7.34.4.4 ParallelClasses	89
7.35 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.GDDFor↔ WhistConstruction Class Reference	90
7.35.1 Detailed Description	90
7.35.2 Constructor & Destructor Documentation	90
7.35.2.1 GDDForWhistConstruction()	90
7.36 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.↔ GDDForWhistConstructionTests Class Reference	91
7.36.1 Detailed Description	91
7.37 TournamentPlanner.Pages.PageModels.Designs.GDDModel Class Reference	91
7.37.1 Detailed Description	92
7.37.2 Property Documentation	92

7.37.2.1 GDD	92
7.37.2.2 M	93
7.37.2.3 V	93
7.38 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner Class Reference	93
7.38.1 Detailed Description	94
7.38.2 Constructor & Destructor Documentation	94
7.38.2.1 GeneralEvenWhistPlanner()	94
7.39 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner↔ Tests Class Reference	94
7.39.1 Detailed Description	95
7.39.2 Member Function Documentation	95
7.39.2.1 LargerConstructionWithoutValidationTest()	95
7.39.2.2 LargerWhistConstructionTest()	95
7.39.2.3 SmallerWhistConstructionTest()	95
7.40 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.General↔ MOLSPairConstruction Class Reference	96
7.40.1 Detailed Description	96
7.41 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.GeneralOddWhistPlanner Class Reference	97
7.41.1 Detailed Description	97
7.41.2 Constructor & Destructor Documentation	97
7.41.2.1 GeneralOddWhistPlanner()	97
7.42 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.General↔ PBD3NPlus1Construction Class Reference	98
7.42.1 Detailed Description	98
7.42.2 Constructor & Destructor Documentation	99
7.42.2.1 GeneralPBD3NPlus1Construction()	99
7.43 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔ GurobiIncrementalBIBDConstructionTests Class Reference	99
7.43.1 Detailed Description	99
7.43.2 Member Function Documentation	99
7.43.2.1 IncrementalBIBDConstructionTest()	100
7.44 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔ GurobiQuadraticProgrammingBIBDConstructionTests Class Reference	100
7.44.1 Detailed Description	100
7.45 TournamentPlanner.Model.Designs.IDesign Interface Reference	100
7.45.1 Detailed Description	101
7.45.2 Property Documentation	101
7.45.2.1 V	101
7.46 TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< IDesign > Interface Template Reference	101
7.46.1 Detailed Description	101
7.46.2 Member Function Documentation	102

7.46.2.1 ConstructDesign()	102
7.47 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Incremental← BIBDConstruction Class Reference	102
7.47.1 Detailed Description	103
7.48 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.← IncrementalBIBDConstructionTests Class Reference	103
7.48.1 Detailed Description	103
7.48.2 Member Function Documentation	103
7.48.2.1 IncrementalBIBDConstructionTest()	103
7.49 TournamentPlanner.Pages.IndexModel Class Reference	104
7.50 TournamentPlanner.Model.Designs.IResolvableDesign Interface Reference	104
7.50.1 Detailed Description	105
7.50.2 Property Documentation	105
7.50.2.1 ParallelClasses	105
7.51 TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< BaseTournament > In- terface Template Reference	105
7.51.1 Detailed Description	105
7.51.2 Member Function Documentation	106
7.51.2.1 PlanTournament()	106
7.52 TournamentPlanner.Model.Designs.BlockDesigns.KTS Class Reference	106
7.52.1 Detailed Description	107
7.52.2 Constructor & Destructor Documentation	107
7.52.2.1 KTS() [1/2]	107
7.52.2.2 KTS() [2/2]	107
7.52.3 Member Function Documentation	108
7.52.3.1 AddParallelClass()	108
7.52.3.2 AddParallelClassAndBlocks()	108
7.53 TournamentPlanner.Pages.PageModels.Designs.KTSModel Class Reference	108
7.53.1 Detailed Description	109
7.53.2 Property Documentation	109
7.53.2.1 KTS	109
7.53.2.2 Multiplier	110
7.53.2.3 V	110
7.54 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Larger← OrderPBDConstruction Class Reference	110
7.54.1 Detailed Description	111
7.54.2 Constructor & Destructor Documentation	111
7.54.2.1 LargerOrderPBDConstruction()	111
7.55 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.← LargerOrderPBDConstructionTests Class Reference	111
7.55.1 Detailed Description	111
7.55.2 Member Function Documentation	111
7.55.2.1 PBD178ConstructionTest()	112

7.55.2.2 TestPBD49To76Construction()	112
7.56 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.LargerWhistPlanner Class Reference	112
7.56.1 Detailed Description	113
7.57 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.LargerWhistPlannerTests Class Reference	113
7.57.1 Detailed Description	113
7.58 TournamentPlanner.Model.Designs.LatinSquares.LatinSquare Class Reference	114
7.58.1 Detailed Description	114
7.58.2 Constructor & Destructor Documentation	114
7.58.2.1 LatinSquare()	114
7.58.3 Property Documentation	115
7.58.3.1 Matrix	115
7.58.3.2 V	115
7.59 TournamentPlanner.Model.Designs.LatinSquares.MOLS Class Reference	115
7.59.1 Detailed Description	116
7.59.2 Constructor & Destructor Documentation	116
7.59.2.1 MOLS() [1/4]	116
7.59.2.2 MOLS() [2/4]	117
7.59.2.3 MOLS() [3/4]	117
7.59.2.4 MOLS() [4/4]	117
7.59.3 Member Function Documentation	117
7.59.3.1 AddOrthogonalLatinSquare()	118
7.59.3.2 GetMOLSMatrices()	118
7.59.3.3 GetNumberOfMOLS()	118
7.59.4 Property Documentation	118
7.59.4.1 LatinSquares	118
7.59.4.2 V	119
7.60 TournamentPlanner.Pages.PageModels.Designs.MOLSModel Class Reference	119
7.60.1 Detailed Description	120
7.60.2 Property Documentation	120
7.60.2.1 MOLS	120
7.60.2.2 MOLSConstructionMethod	120
7.60.2.3 OA	120
7.60.2.4 Order	120
7.61 TournamentPlannerTests.Designs.MOLSSetValidatorTests Class Reference	120
7.61.1 Detailed Description	121
7.61.2 Member Function Documentation	121
7.61.2.1 InvalidMOLSTest()	121
7.61.2.2 ValidMOLSTest()	121
7.62 TournamentPlanner.Datastore.MongoMOLS Class Reference	121
7.62.1 Detailed Description	122
7.62.2 Property Documentation	122

7.62.2.1 Matrices	122
7.62.2.2 NbMOLS	122
7.62.2.3 Order	122
7.63 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Moore↔ RBIBDConstruction Class Reference	123
7.63.1 Detailed Description	123
7.63.2 Constructor & Destructor Documentation	123
7.63.2.1 MooreRBIBDConstruction()	124
7.63.3 Member Function Documentation	124
7.63.3.1 ConstructDesign()	124
7.64 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔ MooreRBIBDConstructionTests Class Reference	124
7.64.1 Detailed Description	125
7.64.2 Member Function Documentation	125
7.64.2.1 ConstructionTestForSeveralOrdersOfRBIBD()	125
7.65 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlanner Class Reference	125
7.65.1 Detailed Description	126
7.65.2 Constructor & Destructor Documentation	126
7.65.2.1 MooreWhistPlanner()	126
7.66 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlannerTests Class Reference	126
7.66.1 Detailed Description	127
7.66.2 Member Function Documentation	127
7.66.2.1 TestValidOrdersOfMooreConstruction()	127
7.67 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPair↔ Construction Class Reference	127
7.67.1 Detailed Description	128
7.68 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Odd↔ MOLSPairConstructionTests Class Reference	128
7.68.1 Detailed Description	128
7.68.2 Member Function Documentation	128
7.68.2.1 EvenOrderShouldThrowExceptionTest()	129
7.68.2.2 OrthogonalityTest()	129
7.69 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.OddSAMDRRWhistPlanner Class Reference	129
7.69.1 Detailed Description	130
7.69.2 Constructor & Destructor Documentation	130
7.69.2.1 OddSAMDRRWhistPlanner()	130
7.70 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.OddSAMDRRWhist↔ PlannerTests Class Reference	130
7.70.1 Detailed Description	131
7.70.2 Member Function Documentation	131
7.70.2.1 TestWhistConstructionUsingSAMDRR()	131
7.71 TournamentPlanner.Model.Designs.OrthogonalArray Class Reference	131

7.71.1 Detailed Description	132
7.71.2 Constructor & Destructor Documentation	132
7.71.2.1 OrthogonalArray()	132
7.71.3 Member Function Documentation	132
7.71.3.1 Height()	132
7.71.3.2 RepresentedMOLSCount()	133
7.71.3.3 Width()	133
7.71.4 Property Documentation	133
7.71.4.1 Matrix	133
7.71.4.2 V	133
7.72 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.OrthogonalArray↔ Construction Class Reference	134
7.72.1 Detailed Description	134
7.72.2 Constructor & Destructor Documentation	134
7.72.2.1 OrthogonalArrayConstruction()	134
7.72.3 Member Function Documentation	135
7.72.3.1 ConstructDesign()	135
7.73 TournamentPlanner.Tests.Utils.OrthogonalArrayMOLSCConverterTests Class Reference	135
7.73.1 Detailed Description	135
7.73.2 Member Function Documentation	135
7.73.2.1 MOLS2OATest()	136
7.73.2.2 OA2MOLSTest()	136
7.74 TournamentPlanner.Pages.Pages__ViewImports Class Reference	136
7.75 TournamentPlanner.Pages.Pages__ViewStart Class Reference	137
7.76 TournamentPlanner.Pages.Pages_Error Class Reference	138
7.76.1 Property Documentation	138
7.76.1.1 ModelExpressionProvider	138
7.77 TournamentPlanner.Pages.Pages_Index Class Reference	139
7.78 TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_BIBD Class Reference	140
7.79 TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_GDD Class Reference	141
7.79.1 Property Documentation	141
7.79.1.1 ModelExpressionProvider	142
7.80 TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_KTS Class Reference	142
7.81 TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_MOLS Class Reference	143
7.82 TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_PBD Class Reference	144
7.83 TournamentPlanner.Pages.PageModels.Testing.Pages_PageModels_Testing_PairwiseTesting Class Reference	145
7.84 TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_↔ BipartiteTournament Class Reference	146
7.85 TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Three↔ PlayerTournaments Class Reference	147
7.86 TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Two↔ PlayerTournament Class Reference	148

7.86.1 Property Documentation	149
7.86.1.1 ModelExpressionProvider	149
7.87 TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Uniform↔ Games Class Reference	150
7.88 TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Whist↔ Tournaments Class Reference	151
7.89 TournamentPlanner.Pages.Pages_Privacy Class Reference	152
7.90 TournamentPlanner.Pages.Shared.Pages_Shared__Layout Class Reference	153
7.90.1 Property Documentation	153
7.90.1.1 ModelExpressionProvider	153
7.91 TournamentPlanner.Pages.Shared.Pages_Shared__ValidationScriptsPartial Class Reference	154
7.92 TournamentPlanner.Pages.Tournament.Pages_Tournament_BipartiteTournament Class Reference	155
7.93 TournamentPlanner.Pages.Tournament.Pages_Tournament_TwoPlayerTournament Class Reference	156
7.93.1 Property Documentation	156
7.93.1.1 ModelExpressionProvider	157
7.94 TournamentPlanner.Model.Testing.PairwiseTestCases Class Reference	157
7.94.1 Detailed Description	158
7.94.2 Constructor & Destructor Documentation	158
7.94.2.1 PairwiseTestCases()	158
7.94.3 Member Function Documentation	158
7.94.3.1 TranslatedTestCases()	158
7.94.4 Property Documentation	158
7.94.4.1 TestCases	158
7.94.4.2 VariableNames	159
7.94.4.3 VariableValues	159
7.95 TournamentPlanner.Algorithms.Testing.PairwiseTestCasesConstruction Class Reference	159
7.95.1 Detailed Description	159
7.95.2 Constructor & Destructor Documentation	159
7.95.2.1 PairwiseTestCasesConstruction()	159
7.95.3 Member Function Documentation	160
7.95.3.1 ConstructPairwiseTestCases()	160
7.96 TournamentPlannerTests.Algorithms.Testing.PairwiseTestCasesConstructionTests Class Reference	160
7.97 TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel Class Reference	161
7.97.1 Detailed Description	161
7.97.2 Constructor & Destructor Documentation	161
7.97.2.1 PairwiseTestingModel()	162
7.97.3 Property Documentation	162
7.97.3.1 TestCases	162
7.97.3.2 VariableValues	162
7.98 TournamentPlanner.Algorithms.TournamentPlanners.PartiallyBalanced3PlayerTournamentPlanner Class Reference	162
7.98.1 Detailed Description	163
7.98.2 Constructor & Destructor Documentation	163

7.98.2.1 PartiallyBalanced3PlayerTournamentPlanner()	163
7.99 TournamentPlanner.Model.Designs.BlockDesigns.PBD Class Reference	163
7.99.1 Detailed Description	164
7.99.2 Constructor & Destructor Documentation	164
7.99.2.1 PBD() [1/3]	165
7.99.2.2 PBD() [2/3]	165
7.99.2.3 PBD() [3/3]	165
7.99.3 Member Function Documentation	165
7.99.3.1 AddBlock()	166
7.99.4 Property Documentation	166
7.99.4.1 K	166
7.100 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.PBD34↔ Construction Class Reference	166
7.100.1 Detailed Description	167
7.100.2 Member Function Documentation	167
7.100.2.1 ConstructDesign()	167
7.101 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.↔ PBD34ConstructionTest Class Reference	167
7.102 TournamentPlanner.Pages.PageModels.Designs.PBDModel Class Reference	168
7.102.1 Detailed Description	168
7.102.2 Property Documentation	168
7.102.2.1 Multiplier	169
7.102.2.2 PBD	169
7.102.2.3 V	169
7.103 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlanner Class Reference	169
7.103.1 Detailed Description	170
7.104 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlannerTests Class Reference	170
7.104.1 Detailed Description	170
7.104.2 Member Function Documentation	170
7.104.2.1 Whist21ConstructionTest()	170
7.105 TournamentPlannerTests.Utils.PrimeFactorizationTests Class Reference	171
7.105.1 Detailed Description	171
7.105.2 Member Function Documentation	171
7.105.2.1 IsPrimeOrPrimePowerTest()	171
7.105.2.2 IsPrimeTest()	171
7.106 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PrimePowerWhistPlanner Class Reference	172
7.106.1 Detailed Description	172
7.106.2 Constructor & Destructor Documentation	172
7.106.2.1 PrimePowerWhistPlanner()	172
7.107 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.PrimePowerWhistPlanner↔ Tests Class Reference	173

7.107.1 Detailed Description	173
7.107.2 Member Function Documentation	173
7.107.2.1 TestSeveralValidInputs()	173
7.108 TournamentPlanner.Pages.PrivacyModel Class Reference	174
7.109 TournamentPlanner.Program Class Reference	174
7.110 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction. \leftrightarrow QuadraticProgrammingBIBDConstruction Class Reference	175
7.110.1 Detailed Description	175
7.111 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction. \leftrightarrow QuadraticProgrammingBIBDConstructionTests Class Reference	176
7.111.1 Detailed Description	176
7.111.2 Member Function Documentation	176
7.111.2.1 QudraticConstructionTest()	176
7.112 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Quasigroup \leftrightarrow PBDConstruction Class Reference	176
7.112.1 Detailed Description	177
7.112.2 Constructor & Destructor Documentation	177
7.112.2.1 QuasigroupPBDConstruction()	177
7.113 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction. \leftrightarrow QuasigroupPBDConstructionTests Class Reference	178
7.113.1 Detailed Description	178
7.113.2 Member Function Documentation	178
7.113.2.1 QuasigroupConstructionTest()	178
7.114 TournamentPlanner.Model.Designs.BlockDesigns.RBIBD Class Reference	179
7.114.1 Detailed Description	179
7.114.2 Constructor & Destructor Documentation	180
7.114.2.1 RBIBD()	180
7.114.3 Member Function Documentation	180
7.114.3.1 AddParallelClass()	180
7.115 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction. \leftrightarrow RecursiveKTSConstruction Class Reference	180
7.115.1 Detailed Description	181
7.115.2 Constructor & Destructor Documentation	181
7.115.2.1 RecursiveKTSConstruction()	181
7.115.3 Member Function Documentation	182
7.115.3.1 ConstructBlockInducedParallelClasses()	182
7.115.3.2 ConstructDesign()	182
7.116 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction. \leftrightarrow RecursiveKTSConstructionTests Class Reference	183
7.116.1 Detailed Description	183
7.116.2 Member Function Documentation	183
7.116.2.1 RecursiveKTSConstructionTest()	183
7.117 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction. \leftrightarrow Resolvable3GDDConstruction Class Reference	183

7.117.1 Detailed Description	184
7.117.2 Constructor & Destructor Documentation	184
7.117.2.1 Resolvable3GDDConstruction()	184
7.117.3 Member Function Documentation	185
7.117.3.1 ConstructDesign()	185
7.118 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.↔ Resolvable3GDDConstructionTests Class Reference	185
7.118.1 Detailed Description	185
7.118.2 Member Function Documentation	185
7.118.2.1 GDDConstructionTest()	185
7.119 TournamentPlanner.Model.Designs.BlockDesigns.RTD Class Reference	186
7.119.1 Detailed Description	186
7.119.2 Constructor & Destructor Documentation	187
7.119.2.1 RTD()	187
7.119.3 Member Function Documentation	187
7.119.3.1 AddParallelClass()	187
7.120 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.RTDConstruction Class Reference	187
7.120.1 Detailed Description	188
7.120.2 Constructor & Destructor Documentation	188
7.120.2.1 RTDConstruction()	188
7.120.3 Member Function Documentation	189
7.120.3.1 ConstructDesign()	189
7.121 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.↔ RTDConstructionTests Class Reference	189
7.121.1 Detailed Description	189
7.121.2 Member Function Documentation	189
7.121.2.1 RTDConstructionTest()	190
7.122 TournamentPlanner.Model.Tournaments.SAMDRR Class Reference	190
7.122.1 Detailed Description	190
7.122.2 Constructor & Destructor Documentation	191
7.122.2.1 SAMDRR()	191
7.123 TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR38Planner Class Reference	191
7.123.1 Detailed Description	192
7.123.2 Member Function Documentation	192
7.123.2.1 PlanTournament()	192
7.124 TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SAMDRR38PlannerTests Class Reference	192
7.124.1 Detailed Description	192
7.125 TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR8Planner Class Reference	193
7.125.1 Detailed Description	193
7.125.2 Member Function Documentation	193
7.125.2.1 PlanTournament()	193

7.126	TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SAMDRR8PlannerTests Class Reference	194
7.126.1	Detailed Description	194
7.127	TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlanner Class Reference	194
7.127.1	Detailed Description	195
7.127.2	Constructor & Destructor Documentation	195
7.127.2.1	SimpleSAMDRRPlanner()	195
7.127.3	Member Function Documentation	195
7.127.3.1	PlanTournament()	195
7.128	TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlannerTests Class Reference	195
7.128.1	Detailed Description	196
7.128.2	Member Function Documentation	196
7.128.2.1	TestSAMDRR11Construction()	196
7.129	TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstruction Class Reference	196
7.129.1	Detailed Description	197
7.129.2	Constructor & Destructor Documentation	197
7.129.2.1	SingleLatinSquareConstruction()	197
7.129.3	Member Function Documentation	197
7.129.3.1	ConstructDesign()	197
7.130	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstructionTests Class Reference	198
7.130.1	Detailed Description	198
7.130.2	Member Function Documentation	198
7.130.2.1	SeveralOrdersTest()	198
7.130.2.2	SingleLatinSquareOfOrder10Test()	198
7.131	TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SkolemSTSCConstruction Class Reference	199
7.131.1	Detailed Description	199
7.131.2	Constructor & Destructor Documentation	199
7.131.2.1	SkolemSTSCConstruction()	199
7.131.3	Member Function Documentation	200
7.131.3.1	ConstructDesign()	200
7.132	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SkolemSTSCConstructionTests Class Reference	200
7.132.1	Detailed Description	200
7.132.2	Member Function Documentation	201
7.132.2.1	SkolemSTSCConstructionTest()	201
7.132.2.2	SkolemSTSExceptionTest()	201
7.133	TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmallBIBDForWhistConstruction Class Reference	201
7.133.1	Detailed Description	202

7.133.2 Constructor & Destructor Documentation	202
7.133.2.1 SmallBIBDForWhistConstruction()	202
7.134 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔ SmallBIBDForWhistConstructionTests Class Reference	202
7.134.1 Detailed Description	203
7.135 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Small↔ GDDConstruction Class Reference	203
7.135.1 Detailed Description	203
7.135.2 Constructor & Destructor Documentation	203
7.135.2.1 SmallGDDConstruction()	204
7.135.3 Member Function Documentation	204
7.135.3.1 ConstructDesign()	204
7.136 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.↔ SmallGDDConstructionTests Class Reference	204
7.136.1 Detailed Description	204
7.136.2 Member Function Documentation	205
7.136.2.1 TestSmallGDDConstructionForAvailableOrders()	205
7.137 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Small↔ OrderPBDConstruction Class Reference	205
7.137.1 Detailed Description	205
7.137.2 Constructor & Destructor Documentation	206
7.137.2.1 SmallOrderPBDConstruction()	206
7.137.3 Member Function Documentation	206
7.137.3.1 ConstructDesign()	206
7.137.4 Property Documentation	206
7.137.4.1 V	206
7.138 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.↔ SmallOrderPBDConstructionTests Class Reference	207
7.138.1 Detailed Description	207
7.138.2 Member Function Documentation	207
7.138.2.1 PBD79And82ConstructionTest()	207
7.138.2.2 SmallPBDUpToOrder46ConstructionTest()	207
7.139 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Smooth↔ KTSCONSTRUCTION Class Reference	208
7.139.1 Detailed Description	208
7.139.2 Constructor & Destructor Documentation	208
7.139.2.1 SmoothKTSCONSTRUCTION()	209
7.139.3 Member Function Documentation	209
7.139.3.1 ConstructDesign()	209
7.140 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔ SmoothKTSCONSTRUCTIONTests Class Reference	209
7.140.1 Detailed Description	210
7.140.2 Member Function Documentation	210
7.140.2.1 SmoothKTSCONSTRUCTIONTest()	210

7.141	TournamentPlanner.Model.Designs.LatinSquares.SOLS Class Reference	210
7.141.1	Detailed Description	211
7.141.2	Constructor & Destructor Documentation	211
7.141.2.1	SOLS()	211
7.141.3	Member Function Documentation	211
7.141.3.1	Transpose()	211
7.142	TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNoGCD6Construction Class Reference	212
7.142.1	Detailed Description	212
7.142.2	Constructor & Destructor Documentation	212
7.142.2.1	SOLSWithNoGCD6Construction()	212
7.142.3	Member Function Documentation	213
7.142.3.1	ConstructDesign()	213
7.143	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNoGCD6ConstructionTests Class Reference	213
7.143.1	Detailed Description	213
7.143.2	Member Function Documentation	213
7.143.2.1	SOLSConstructionTest()	214
7.144	TournamentPlanner.Startup Class Reference	214
7.145	TournamentPlanner.Model.Designs.BlockDesigns.STS Class Reference	214
7.145.1	Detailed Description	215
7.145.2	Constructor & Destructor Documentation	215
7.145.2.1	STS() [1/2]	215
7.145.2.2	STS() [2/2]	215
7.146	TournamentPlanner.Model.Designs.BlockDesigns.TD Class Reference	216
7.146.1	Detailed Description	216
7.146.2	Constructor & Destructor Documentation	217
7.146.2.1	TD() [1/2]	217
7.146.2.2	TD() [2/2]	217
7.146.3	Property Documentation	217
7.146.3.1	Groups	217
7.146.3.2	K	218
7.146.3.3	M	218
7.147	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TD17ConstructionTests Class Reference	218
7.147.1	Detailed Description	218
7.148	TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction Class Reference	218
7.148.1	Detailed Description	219
7.148.2	Constructor & Destructor Documentation	219
7.148.2.1	TDConstruction()	219
7.148.3	Member Function Documentation	219
7.148.3.1	ConstructDesign()	219

7.148.4 Member Data Documentation	220
7.148.4.1 k	220
7.148.4.2 m	220
7.149 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.TDWhistPlanner Class Reference	220
7.149.1 Detailed Description	221
7.149.2 Constructor & Destructor Documentation	221
7.149.2.1 TDWhistPlanner()	221
7.150 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.TDWhistPlannerTests Class Reference	221
7.150.1 Detailed Description	222
7.151 TournamentPlanner.Model.Tournaments.ThreePlayerTournament Class Reference	222
7.151.1 Detailed Description	222
7.151.2 Constructor & Destructor Documentation	223
7.151.2.1 ThreePlayerTournament()	223
7.151.3 Member Function Documentation	223
7.151.3.1 TotalNumberOfGames()	223
7.151.4 Property Documentation	223
7.151.4.1 PlayerGroups	223
7.152 TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournamentsModel Class Reference	224
7.152.1 Detailed Description	224
7.152.2 Property Documentation	224
7.152.2.1 GDDTournament	225
7.152.2.2 KTSTournament	225
7.152.2.3 NbPlayers	225
7.153 TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin Class Reference	225
7.153.1 Detailed Description	226
7.153.2 Constructor & Destructor Documentation	226
7.153.2.1 TwoPlayerRoundRobin()	226
7.153.3 Member Function Documentation	226
7.153.3.1 AddRoundSchedule() [1/2]	226
7.153.3.2 AddRoundSchedule() [2/2]	227
7.153.4 Member Data Documentation	227
7.153.4.1 InactiveByRounds	227
7.154 TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlanner Class Reference	227
7.154.1 Detailed Description	228
7.154.2 Constructor & Destructor Documentation	228
7.154.2.1 TwoPlayerRoundRobinPlanner()	228
7.154.3 Member Function Documentation	229
7.154.3.1 PlanTournament()	229
7.154.4 Property Documentation	229
7.154.4.1 N	229
7.154.4.2 NbPlayers	229

7.154.4.3 OddNbPlayers	229
7.155 TournamentPlanner.Pages.PageModels.Tournaments.TwoPlayerTournamentModel Class Reference	230
7.155.1 Detailed Description	230
7.155.2 Property Documentation	230
7.155.2.1 NbPlayers	231
7.155.2.2 Tournament	231
7.156 TournamentPlanner.Model.Tournaments.UniformGameSizeTournament Class Reference	231
7.156.1 Detailed Description	232
7.156.2 Constructor & Destructor Documentation	232
7.156.2.1 UniformGameSizeTournament()	232
7.156.3 Member Function Documentation	232
7.156.3.1 TotalNumberOfGames()	232
7.156.4 Property Documentation	233
7.156.4.1 LeftOverPlayersByRounds	233
7.156.4.2 ObjectiveValue	233
7.157 TournamentPlanner.Algorithms.TournamentPlanners.UniformGameSizeTournamentPlanner Class Reference	233
7.157.1 Detailed Description	234
7.157.2 Constructor & Destructor Documentation	234
7.157.2.1 UniformGameSizeTournamentPlanner()	234
7.158 TournamentPlannerTests.Algorithms.TournamentPlanners.UniformGameSizeTournamentPlannerTests Class Reference	234
7.158.1 Detailed Description	235
7.159 TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel Class Reference	235
7.159.1 Detailed Description	236
7.159.2 Property Documentation	236
7.159.2.1 GameSize	236
7.159.2.2 NbPlayers	236
7.159.2.3 NbRounds	236
7.159.2.4 TimeLimit	236
7.159.2.5 Tournament	237
7.160 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.Whist152Planner Class Reference	237
7.160.1 Detailed Description	237
7.160.2 Constructor & Destructor Documentation	238
7.160.2.1 Whist152Planner()	238
7.161 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.Whist152PlannerTests Class Reference	238
7.161.1 Detailed Description	238
7.161.2 Member Function Documentation	238
7.161.2.1 Whist152ConstructionTest()	238
7.162 TournamentPlanner.Model.Tournaments.WhistTournament Class Reference	239
7.162.1 Detailed Description	239
7.162.2 Constructor & Destructor Documentation	239

7.162.2.1 WhistTournament() [1/3]	239
7.162.2.2 WhistTournament() [2/3]	240
7.162.2.3 WhistTournament() [3/3]	240
7.162.3 Property Documentation	240
7.162.3.1 LeftOverByRounds	240
7.163 TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel Class Reference	241
7.163.1 Detailed Description	241
7.163.2 Property Documentation	241
7.163.2.1 NbPlayers	242
7.163.2.2 Whist	242
7.164 TournamentPlannerTests.Model.Tournaments.WhistTournamentValidator Class Reference	242
7.164.1 Detailed Description	242
7.164.2 Member Function Documentation	242
7.164.2.1 ValidateWhistTournament()	242
7.165 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Wilson↔ GDDConstruction Class Reference	243
7.165.1 Detailed Description	243
7.165.2 Constructor & Destructor Documentation	244
7.165.2.1 WilsonGDDConstruction()	244
7.165.3 Member Function Documentation	244
7.165.3.1 ConstructDesign()	244
7.166 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.↔ WilsonGDDConstructionTests Class Reference	244
7.166.1 Detailed Description	245
Index	247

Chapter 1

TournamentPlanner

This repository contains both the practical and the theoretical (writing) part of my master's thesis.

To be able to run the project, you need to add a reference to the `Gurobi90.NET.dll` (or a newer version) at `TournamentPlanner.csproj` file (or using VS's add reference option).

The `latex` folder in the root directory contains the writing part of the thesis in the `latex/dp.pdf` file.

Chapter 2

The MIT License (MIT)

Copyright Jörn Zaefferer

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Chapter 3

Namespace Index

3.1 Packages

Here are the packages with brief descriptions (if available):

TournamentPlanner	21
TournamentPlanner.Algorithms	21
TournamentPlanner.Algorithms.ConstructionAlgorithms	21
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns	21
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction	21
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction	22
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction	22
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction	23
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares	23
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA	23
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal	24
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal	24
TournamentPlanner.Algorithms.Testing	24
TournamentPlanner.Algorithms.TournamentPlanners	25
TournamentPlanner.Algorithms.TournamentPlanners.Spouses	25
TournamentPlanner.Algorithms.TournamentPlanners.Whist	25
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even	26
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd	26
TournamentPlanner.Algorithms.Utils	26
TournamentPlanner.Datastore	27
TournamentPlanner.Model	27
TournamentPlanner.Model.Designs	27
TournamentPlanner.Model.Designs.BlockDesigns	27
TournamentPlanner.Model.Designs.LatinSquares	28
TournamentPlanner.Model.Testing	28
TournamentPlanner.Model.Tournaments	28
TournamentPlanner.Pages	29
TournamentPlanner.Pages.PageModels	29
TournamentPlanner.Pages.PageModels.Designs	29
TournamentPlanner.Pages.PageModels.Testing	30
TournamentPlanner.Pages.PageModels.Tournaments	31
TournamentPlanner.Pages.Shared	31
TournamentPlanner.Pages.Tournament	31
TournamentPlanner.Utils	31
TournamentPlannerTests	32

TournamentPlannerTests.Algorithms	32
TournamentPlannerTests.Algorithms.ConstructionAlgorithms	32
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns	32
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction	32
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction	32
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction	33
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction	33
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares	33
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal	33
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal	34
TournamentPlannerTests.Algorithms.Testing	34
TournamentPlannerTests.Algorithms.TournamentPlanners	34
TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses	34
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist	35
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even	35
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd	35
TournamentPlannerTests.Datastore	35
TournamentPlannerTests.Designs	36
TournamentPlannerTests.Model	36
TournamentPlannerTests.Model.Designs	36
TournamentPlannerTests.Model.Designs.BlockDesigns	36
TournamentPlannerTests.Model.Designs.LatinSquares	36
TournamentPlannerTests.Model.Tournaments	37
TournamentPlannerTests.Utils	37

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

TournamentPlanner.Model.Tournaments.BaseTournament	49
TournamentPlanner.Model.Tournaments.BipartiteTournament	59
TournamentPlanner.Model.Tournaments.SAMDRR	190
TournamentPlanner.Model.Tournaments.ThreePlayerTournament	222
TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin	225
TournamentPlanner.Model.Tournaments.UniformGameSizeTournament	231
TournamentPlanner.Model.Tournaments.WhistTournament	239
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Bose↔ STSCConstructionTests	65
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Constraint↔ ProgrammingMOLSPairConstructionTests	69
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlannerTests	71
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlannerTests	73
TournamentPlannerTests.Datastore.DataReaderTests	73
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Datastore↔ MOLSCConstructionTests	75
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Difference↔ MOLSPairConstructionTests	77
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlannerTests	80
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.EvenOddProductWhistPlannerTests	82
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPair↔ ConstructionTests	84
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.GDDFor↔ WhistConstructionTests	91
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlannerTests	94
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Gurobi↔ IncrementalBIBDConstructionTests	99
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Gurobi↔ QuadraticProgrammingBIBDConstructionTests	100
TournamentPlanner.Model.Designs.IDesign	100
TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign	43
TournamentPlanner.Model.Designs.BlockDesigns.BIBD	54
TournamentPlanner.Model.Designs.BlockDesigns.KTS	106
TournamentPlanner.Model.Designs.BlockDesigns.RBIBD	179
TournamentPlanner.Model.Designs.BlockDesigns.STS	214

TournamentPlanner.Model.Designs.BlockDesigns.GDD	85
TournamentPlanner.Model.Designs.BlockDesigns.PBD	163
TournamentPlanner.Model.Designs.BlockDesigns.TD	216
TournamentPlanner.Model.Designs.BlockDesigns.RTD	186
TournamentPlanner.Model.Designs.IResolvableDesign	104
TournamentPlanner.Model.Designs.BlockDesigns.GDD	85
TournamentPlanner.Model.Designs.BlockDesigns.KTS	106
TournamentPlanner.Model.Designs.BlockDesigns.RBIBD	179
TournamentPlanner.Model.Designs.BlockDesigns.RTD	186
TournamentPlanner.Model.Designs.LatinSquares.LatinSquare	114
TournamentPlanner.Model.Designs.LatinSquares.SOLS	210
TournamentPlanner.Model.Designs.LatinSquares.MOLS	115
TournamentPlanner.Model.Designs.OrthogonalArray	131
TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< IDesign >	101
TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< BIBD >	101
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Base↔	
BIBDConstruction	40
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Bose↔	
STSConstruction	63
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔	
IncrementalBIBDConstruction	102
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Moore↔	
RBIBDConstruction	123
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔	
QuadraticProgrammingBIBDConstruction	175
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔	
RecursiveKTSConstruction	180
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔	
SkolemSTSConstruction	199
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Small↔	
BIBDForWhistConstruction	201
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔	
SmoothKTSConstruction	208
TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< GDD >	101
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.GDDFor↔	
WhistConstruction	90
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Resolvable3↔	
GDDConstruction	183
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Small↔	
GDDConstruction	203
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Wilson↔	
GDDConstruction	243
TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< LatinSquare >	101
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.CommutativeIdempotentLatin↔	
SquareConstruction	66
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstruction	196
TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< MOLS >	101
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.BaseMOLSConstruction	45
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Constraint↔	
ProgrammingMOLSPairConstruction	67
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Datastore↔	
MOLSConstruction	74
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Difference↔	
MOLSPairConstruction	76
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPair↔	
Construction	82

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.General↔	
MOLSPairConstruction	96
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPair↔	
Construction	127
TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< OrthogonalArray >	101
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.OrthogonalArrayConstruction	134
TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< PBD >	101
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.General↔	
PBD3NPlus1Construction	98
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Larger↔	
OrderPBDConstruction	110
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.PBD34↔	
Construction	166
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Small↔	
OrderPBDConstruction	205
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Quasigroup↔	
PBDConstruction	176
TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< SOLS >	101
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNo↔	
GCD6Construction	212
TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< TD >	101
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction	218
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.RTDCConstruction	187
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Incremental↔	
BIBDConstructionTests	103
TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< BaseTournament >	105
TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< BipartiteTournament >	105
TournamentPlanner.Algorithms.TournamentPlanners.BipartiteTournamentPlanner	62
TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< SAMDRR >	105
TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR38Planner	191
TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR8Planner	193
TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlanner	194
TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< ThreePlayerTournament >	105
TournamentPlanner.Algorithms.TournamentPlanners.Balanced3PlayerTournamentPlanner	39
TournamentPlanner.Algorithms.TournamentPlanners.PartiallyBalanced3PlayerTournamentPlanner	162
TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< TwoPlayerRoundRobin >	105
TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlanner	227
TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< UniformGameSizeTournament	
>	105
TournamentPlanner.Algorithms.TournamentPlanners.UniformGameSizeTournamentPlanner	233
TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< WhistTournament >	105
TournamentPlanner.Algorithms.TournamentPlanners.Whist.BaseWhistPlanner	52
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlanner	72
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlanner	78
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenOddProductWhistPlanner	80
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner	93
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.LargerWhistPlanner	112
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlanner	125
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.OddSAMDRRWhistPlanner	129
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.TDWhistPlanner	220
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.Whist152Planner	237
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlanner	70
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.GeneralOddWhistPlanner	97
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlanner	169
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PrimePowerWhistPlanner	172

TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Larger↔ OrderPBDConstructionTests	111
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.LargerWhistPlannerTests	113
TournamentPlannerTests.Designs.MOLSSetValidatorTests	120
TournamentPlanner.Datastore.MongoMOLS	121
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Moore↔ RBIBDConstructionTests	124
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlannerTests	126
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPair↔ ConstructionTests	128
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.OddSAMDRRWhistPlannerTests	130
TournamentPlannerTests.Utils.OrthogonalArrayMOLSConverterTests	135
global.MicrosoftAspNetCore.Mvc.RazorPages.Page	
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_BIBD	140
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_BIBD	140
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_GDD	141
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_GDD	141
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_KTS	142
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_KTS	142
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_MOLS	143
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_MOLS	143
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_PBD	144
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_PBD	144
TournamentPlanner.Pages.PageModels.Testing.Pages_PageModels_Testing_PairwiseTesting	145
TournamentPlanner.Pages.PageModels.Testing.Pages_PageModels_Testing_PairwiseTesting	145
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Bipartite↔ Tournament	146
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Three↔ PlayerTournaments	147
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Three↔ PlayerTournaments	147
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Two↔ PlayerTournament	148
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Uniform↔ Games	150
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Uniform↔ Games	150
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Whist↔ Tournaments	151
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_Whist↔ Tournaments	151
TournamentPlanner.Pages.Pages_Error	138
TournamentPlanner.Pages.Pages_Error	138
TournamentPlanner.Pages.Pages_Index	139
TournamentPlanner.Pages.Pages_Index	139
TournamentPlanner.Pages.Pages_Privacy	152
TournamentPlanner.Pages.Pages_Privacy	152
TournamentPlanner.Pages.Tournament.Pages_Tournament_BipartiteTournament	155
TournamentPlanner.Pages.Tournament.Pages_Tournament_TwoPlayerTournament	156
PageModel	
TournamentPlanner.Pages.ErrorModel	78
TournamentPlanner.Pages.IndexModel	104
TournamentPlanner.Pages.PageModels.BasePageModel	48
TournamentPlanner.Pages.PageModels.Designs.BIBDModel	57
TournamentPlanner.Pages.PageModels.Designs.GDDModel	91
TournamentPlanner.Pages.PageModels.Designs.KTSModel	108
TournamentPlanner.Pages.PageModels.Designs.MOLSModel	119
TournamentPlanner.Pages.PageModels.Designs.PBDModel	168

TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel	161
TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournamentModel	60
TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournamentsModel	224
TournamentPlanner.Pages.PageModels.Tournaments.TwoPlayerTournamentModel	230
TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel	235
TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel	241
TournamentPlanner.Pages.PrivacyModel	174
TournamentPlanner.Model.Testing.PairwiseTestCases	157
TournamentPlanner.Algorithms.Testing.PairwiseTestCasesConstruction	159
TournamentPlannerTests.Algorithms.Testing.PairwiseTestCasesConstructionTests	160
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.PBD34↔ ConstructionTest	167
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlannerTests	170
TournamentPlannerTests.Utils.PrimeFactorizationTests	171
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.PrimePowerWhistPlannerTests	173
TournamentPlanner.Program	174
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Quadratic↔ ProgrammingBIBDConstructionTests	176
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Quasigroup↔ PBDConstructionTests	178
global.Microsoft.AspNetCore.Mvc.Razor.RazorPage	
TournamentPlanner.Pages.Pages__ViewImports	136
TournamentPlanner.Pages.Pages__ViewImports	136
TournamentPlanner.Pages.Pages__ViewStart	137
TournamentPlanner.Pages.Pages__ViewStart	137
TournamentPlanner.Pages.Shared.Pages_Shared__Layout	153
TournamentPlanner.Pages.Shared.Pages_Shared__Layout	153
TournamentPlanner.Pages.Shared.Pages_Shared__ValidationScriptsPartial	154
TournamentPlanner.Pages.Shared.Pages_Shared__ValidationScriptsPartial	154
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Recursive↔ KTSCONSTRUCTIONTests	183
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Resolvable3↔ GDDConstructionTests	185
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.RTDConstruction↔ Tests	189
TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SAMDRR38PlannerTests	192
TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SAMDRR8PlannerTests	194
TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlannerTests	195
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstruction↔ Tests	198
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Skolem↔ STSCONSTRUCTIONTests	200
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Small↔ BIBDForWhistConstructionTests	202
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Small↔ GDDConstructionTests	204
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Small↔ OrderPBDConstructionTests	207
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Smooth↔ KTSCONSTRUCTIONTests	209
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWith↔ NoGCD6ConstructionTests	213
TournamentPlanner.Startup	214
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TD17↔ ConstructionTests	218
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.TDWhistPlannerTests	221
TournamentPlannerTests.Algorithms.TournamentPlanners.UniformGameSizeTournamentPlannerTests	234
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.Whist152PlannerTests	238

TournamentPlannerTests.Model.Tournaments.WhistTournamentValidator	242
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Wilson \leftrightarrow GDDConstructionTests	244

Chapter 5

Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

TournamentPlanner.Algorithms.TournamentPlanners.Balanced3PlayerTournamentPlanner	
Plane a balanced three player tournament. Perfectly balanced three player tournament can only be planed, if the number of players $V = 6N + 3$	39
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction	
Base class for the BIBD construction algorithms	40
TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign	
Base class for the block designs	43
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.BaseMOLSCOnstruction	
Base class for the mutually orthogonal latin square construction algorithms	45
TournamentPlanner.Pages.PageModels.BasePageModel	
The base class for the page models of designs and tournaments	48
TournamentPlanner.Model.Tournaments.BaseTournament	
Base abstract class for the tournaments	49
TournamentPlanner.Algorithms.TournamentPlanners.Whist.BaseWhistPlanner	
The base class for whist tournament planners	52
TournamentPlanner.Model.Designs.BlockDesigns.BIBD	
Class representing a balanced incomplete block design	54
TournamentPlanner.Pages.PageModels.Designs.BIBDModel	
The page model for BIBDs	57
TournamentPlanner.Model.Tournaments.BipartiteTournament	
Class representing a bipartite tournament. In a bipartite tournament, there are 2 teams with the same number of players. The goal is, that every player from the first team should play a game with every player from the second team	59
TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournamentModel	
Page model for the construction of bipartite tournaments	60
TournamentPlanner.Algorithms.TournamentPlanners.BipartiteTournamentPlanner	
Bipartite tournament planner using Latin squares	62
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BoseSTSCOnstruction	
Bose's algorithm for the construction of STS of size $V = 6N + 3$. http://math.ucdenver.edu/~wcherowi/courses/m6406/sts.pdf	63
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BoseSTSCOnstructionTests	
Tests for the BoseSTSCOnstruction	65
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.CommutativeIdempotentLatinSquareConstruction	
Algorithm for the creation od commutative idempotent Latin squares of odd order	66

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.ConstraintProgrammingMOLSPairConstruction	
Constraint a pair of MOLS using constraint programming approach with CP-SAT solver. This method only works for MOLS of small orders	67
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.ConstraintProgrammingMOLSPairConstruction	
Tests for the ConstraintProgrammingMOLSPairConstruction	69
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlanner	
Cyclic odd whist planner for $v = 33, 57$	70
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlannerTests	
Tests for the CyclicOddWhistPlanner	71
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlanner	
Cyclic construction method for several orders of Whist(V = 4N)	72
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlannerTests	
Tests for the CyclicWhistPlanner	73
TournamentPlannerTests.Datastore.DataReaderTests	
Tests for the DataReader	73
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DatastoreMOLSConstruction	
This class creates MOLS objects based on the ones in the data store. This is not a construction algorithm per se, but acts like one. It implements the same interface as the other MOLS construction methods so it is easier to use in other algorithms	74
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DatastoreMOLSConstructionTests	
Tests for the DatastoreMOLSConstruction	75
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DifferenceMOLSPairConstruction	
Construct a pair of MOLS of order $3M + 1$. This way, MOLS for all orders $12t + 10$ (which is a subset of $4k + 2$) can be constructed. This method was introduced in https://www.math.uci.edu/~brusso/[14]BosShrParCJM1960.pdf	76
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DifferenceMOLSPairConstructionTests	
Tests for the DifferenceMOLSPairConstruction	77
TournamentPlanner.Pages.ErrorModel	78
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlanner	
Construction of a Whist tournament as a product of 2 existing Wh(4n) tournaments	78
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlannerTests	
Tests for the ProductWhistPlanner	80
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenOddProductWhistPlanner	
Construction of a Whist tournament as a product of a Wh(4n) and Wh(4n + 1)	80
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.EvenOddProductWhistPlannerTests	
Tests for the EvenOddProductWhistPlanner	82
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPairConstruction	
Construction of MOLS of order $V = 4 * K$	82
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPairConstructionTests	
Tests for the FourKMOLSPairConstruction	84
TournamentPlanner.Model.Designs.BlockDesigns.GDD	
Class representing a group divisible design	85
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.GDDForWhistConstruction	
Construction of $GDD(17g, + 4u + 4v, \{5, 17\}, \{g, g + 4u, g + 4v\})$. These are used to construct whist tournaments	90
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.GDDForWhistConstructionTests	
Tests for the GDDForWhistConstruction	91
TournamentPlanner.Pages.PageModels.Designs.GDDModel	
Page model for resolvable group divisible designs with block size 3	91
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner	
Plan a whist tournament of order $4n$ or $4n + 1$ using some of the implemented approaches. This class basically just selects the method to use for the construction	93
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlannerTests	
Tests for the General4NWhistPlanner	94
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.GeneralMOLSPairConstruction	
Class facilitating the construction of MOLS of all the supported orders. Uses other construction algorithms	96

TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.GeneralOddWhistPlanner	
Construct some Whist tournaments of order $4n + 1$. Not all orders are supported unfortunately	97
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.GeneralPBD3NPlus1Construction	
Class that is used for construction of any $PBD(3n+1)$. It uses the required depending on the order. It is also used in the recursive step of the LargerOrderPBDConstruction	98
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.GurobiIncrementalBIBDConstruction	
Tests for the GurobiIncrementalBIBDConstruction	99
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.GurobiQuadraticProgrammingBIBDConstruction	
Tests for the QuadraticProgrammingBIBDConstruction	100
TournamentPlanner.Model.Designs.IDesign	
Common interface for all the design structures	100
TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< IDesign >	
Interface for the design construction algorithms	101
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.IncrementalBIBDConstruction	
Incremental method that constructs BIBDs block by block. Implementation of the algorithm described in https://www.researchgate.net/publication/220660995_A_mathematical_programming_approach_to_the_construction_of_BIBDs . Uses CP-SAT as solver	102
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.IncrementalBIBDConstructionTests	
Test for the IncrementalBIBDConstruction	103
TournamentPlanner.Pages.IndexModel	104
TournamentPlanner.Model.Designs.IResolvableDesign	
Interface for resolvable designs	104
TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< BaseTournament >	
Interface for the tournament planners	105
TournamentPlanner.Model.Designs.BlockDesigns.KTS	
A class representing a Kirkman triple system. A <i>KTS</i> is a resolvable $BIBD(V, 3, 1)$, where $V = 6N + 3$. Basically, a resolvable <i>STS</i> is a <i>KTS</i>	106
TournamentPlanner.Pages.PageModels.Designs.KTSMModel	
Page model for the Kirkman triple system creation	108
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.LargerOrderPBDConstruction	
Construction of larger order PBDs, where the order is $V = 3N + 1$	110
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.LargerOrderPBDConstructionTests	
Tests for the LargerOrderPBDConstruction	111
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.LargerWhistPlanner	
Construct $Wh(4n)$, where $n > 80$	112
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.LargerWhistPlannerTests	
Tests for the LargerWhistPlanner	113
TournamentPlanner.Model.Designs.LatinSquares.LatinSquare	
The Latin square	114
TournamentPlanner.Model.Designs.LatinSquares.MOLS	
Set of mutually orthogonal Latin squares. For simplicity, it is allowed for the set of MOLS to contain only a single Latin square	115
TournamentPlanner.Pages.PageModels.Designs.MOLSMModel	
Page model for mutually orthogonal Latin squares	119
TournamentPlannerTests.Designs.MOLSSetValidatorTests	
Tests for the MOLSSetValidator	120
TournamentPlanner.Datastore.MongoMOLS	
Class representing a MOLS in mongo DB	121
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.MooreRBIBDConstruction	
Construction of $RBIBD(3q+1, 4, 1)$ where q is a prime power and can be written as $q = 4m + 1$. These $RBIBD$ are used to construct Whist tournaments	123
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.MooreRBIBDConstructionTests	
Tests for the MooreRBIBDConstruction	124
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlanner	
This class implements the construction of $Wh(v = 4n)$, where $v = 3q + 1$ and $q = 1 \pmod{4} = 4m + 1$ and is a prime power. $v = 28, 40, 52, 76, 88 \dots$ https://www.jstor.org/stable/2369860?seq=1 ::metadata_info_tab_contents	125

TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlannerTests	
Tests for the MooreWhistPlanner	126
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPairConstruction	
Algorithm to construct pairs of mutually orthogonal Latin squares of odd order. https://www.whitman.edu/mathematics/cgt_online/book/section04.03.html	127
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPairConstructionTests	
Tests for the OddMOLSPairConstruction	128
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.OddSAMDRRWhistPlanner	
Use odd order SAMMDR to construct several orders of Whist(4n)	129
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.OddSAMDRRWhistPlannerTests	
Tests for the SAMDRRWhistPlanner	130
TournamentPlanner.Model.Designs.OrthogonalArray	
Class representing the Orthogonal array structure. In some literature, this representation is actually the transpose of an orthogonal array	131
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.OrthogonalArrayConstruction	
Construction method for OAs using MOLS	134
TournamentPlannerTests.Utils.OrthogonalArrayMOLSConverterTests	
Tests for the OrthogonalArrayMOLSConverter	135
TournamentPlanner.Pages.Pages__ViewImports	136
TournamentPlanner.Pages.Pages__ViewStart	137
TournamentPlanner.Pages.Pages_Error	138
TournamentPlanner.Pages.Pages_Index	139
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_BIBD	140
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_GDD	141
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_KTS	142
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_MOLS	143
TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_PBD	144
TournamentPlanner.Pages.PageModels.Testing.Pages_PageModels_Testing_PairwiseTesting	145
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_BipartiteTournament	146
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_ThreePlayerTournaments	147
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_TwoPlayerTournament	148
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_UniformGames	150
TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_WhistTournaments	151
TournamentPlanner.Pages.Pages_Privacy	152
TournamentPlanner.Pages.Shared.Pages_Shared__Layout	153
TournamentPlanner.Pages.Shared.Pages_Shared__ValidationScriptsPartial	154
TournamentPlanner.Pages.Tournament.Pages_Tournament_BipartiteTournament	155
TournamentPlanner.Pages.Tournament.Pages_Tournament_TwoPlayerTournament	156
TournamentPlanner.Model.Testing.PairwiseTestCases	
A class representing the pairwise test cases created using orthogonal arrays and some post processing methods	157
TournamentPlanner.Algorithms.Testing.PairwiseTestCasesConstruction	
Class implementing the construction of the pairwise test cases using orthogonal arrays	159
TournamentPlannerTests.Algorithms.Testing.PairwiseTestCasesConstructionTests	160
TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel	
Page model for the construction of pairwise tests	161
TournamentPlanner.Algorithms.TournamentPlanners.PartiallyBalanced3PlayerTournamentPlanner	
This class facilitates the construction of partially balanced three player tournaments. The number of players in the tournament has to be divisible by 3. For the construction of the tournament, GDD are used	162
TournamentPlanner.Model.Designs.BlockDesigns.PBD	
Class representing a Pairwise balanced design. PBD is similar to BIBD, but the block sizes are from K - they don't have to be uniform. We only consider PBDs with $\lambda = 1$	163

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.PBD34Construction	
The construction of PBD 34 was supposed to be in the SmallOrderPBDConstruction , but was a bit too complicated so i extracted it to its custom class. For the construction of PBD34 a PBD27 needs to be constructed first. Then it is modified and new blocks are added to get a PBD34 . . .	166
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.PBD34ConstructionTest	167
TournamentPlanner.Pages.PageModels.Designs.PBDModel	
Page model for the pairwise balanced design. Construction of PBD of orders $3M + 1$ is implemented	168
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlanner	
Class facilitating the construction of Whist(21) and Whist(69) using PBDs	169
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlannerTests	
Tests for the PBDWhistPlannerTests	170
TournamentPlannerTests.Utils.PrimeFactorizationTests	
Tests for the PrimeFactorization static class	171
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PrimePowerWhistPlanner	
Whist tournament planner for prime power number of players in the form of $v = 4n + 1$. It uses finite fields for the tournament construction	172
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.PrimePowerWhistPlannerTests	
Tests for the PrimeNWhistPlanner	173
TournamentPlanner.Pages.PrivacyModel	174
TournamentPlanner.Program	174
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.QuadraticProgrammingBIBDConstruction	
A quadratic programming approach to the construction of BIBDs as described in the thesis. Uses CP-SAT as solver	175
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.QuadraticProgrammingBIBDConstructionTests	
Tests for the QuadraticProgrammingBIBDConstruction	176
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.QuasigroupPBDConstruction	
Method for construction of PBDs of orders 13, 16 and 28. This method uses pairs of MOLS as quasigroups	176
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.QuasigroupPBDConstructionTests	
Tests for the QuasigroupPBDConstruction	178
TournamentPlanner.Model.Designs.BlockDesigns.RBIBD	
Class representing a resolvable balanced incomplete blocks design	179
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.RecursiveKTSConstruction	
Algorithm for the recursive construction of KTS of order $6N + 3$. Method described in https://books.google.tg/books?id=OoRFisqn-aIC . KTSs of orders 9, 15, 21 and 39 are constructed statically, all the others are constructed in a recursive manner	180
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.RecursiveKTSConstructionTests	
Tests for the RecursiveKTSConstruction	183
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Resolvable3GDDConstruction	
Construct resolvable group divisible designs with block size 3 using MOLS. Should be able to construct GDDs where the order is a multiple of 3 except 6 and 18	183
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Resolvable3GDDConstructionTests	
Tests the creation of resolvable GDDs by Resolvable3GDDConstruction	185
TournamentPlanner.Model.Designs.BlockDesigns.RTD	
Resolvable transversal design	186
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.RTDConstruction	
Construction method for resolvable transversal designs	187
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.RTDConstructionTests	
Tests for the RTDConstruction	189
TournamentPlanner.Model.Tournaments.SAMDRR	
Spouse-avoiding mixed doubles round robin tournament. Games are represented as 4 integers, $(i, l, j, k) - H_j W_l \vee H_j W_k$	190
TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR38Planner	
SAMDRR 38 planner. $GCD(38, 6) \neq 1$ so a special construction is required. SAMDRR 38 is used to construct Whist(152)	191

TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SAMDRR38PlannerTests	
Tests for the SAMDRR38Planner	192
TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR8Planner	
SAMDRR 8 construction class	193
TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SAMDRR8PlannerTests	
Tests for the SAMDRR8Planner	194
TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlanner	
SAMDRR planner using cyclic SOLS such that for the SOLS order n , $\text{GCD}(n, 6) = 1$	194
TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlannerTests	
Tests for the SimpleSAMDRRPlanner	195
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstruction	
Recursive construction of a single Latin square of any order	196
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstructionTests	
Tests for the SingleLatinSquareConstruction	198
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SkolemSTSConstruction	
Skolem construction algorithm for STS of size $6N + 1$. http://math.ucdenver.edu/~wcherowi/courses/m6406/sts.pdf	199
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SkolemSTSConstructionTests	
Tests for the SkolemSTSConstruction	200
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmallBIBDForWhistConstruction	
Static construction of BIBD(21, 5, 1) and BIBD(25, 5, 1). These designs have been constructed by the incremental / quadratic methods. Those methods are, however, unreliable (license and memory issues) so the constructed designs are stored here for faster and more reliable access	201
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmallBIBDForWhistConstructionTests	
Tests for SmallBIBDForWhistConstruction	202
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.SmallGDDConstruction	
Construct a few small order GDDs which are then used in Wilsons fundamental construction	203
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.SmallGDDConstructionTests	
Tests for the SmallGDDConstruction	204
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.SmallOrderPBDConstruction	
A class facilitating the construction of PBDs of smaller orders in form $3N + 1$. All the constructed design have block sizes in {4, 7, 10, 19}. There are several construction methods, not a single unified one	205
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.SmallOrderPBDConstructionTests	
Tests for the SmallOrderPBDConstruction	207
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmoothKTSConstruction	
Algorithm for construction of smooth Kirkman triple systems. The size of smooth KTS is a power of 3. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.501.8928&rep=rep1&type=pdf	208
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmoothKTSConstructionTests	
Tests for the SmoothKTSConstruction	209
TournamentPlanner.Model.Designs.LatinSquares.SOLS	
Self orthogonal Latin square	210
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNoGCD6Construction	
Construct a cyclic SOLS such that $\text{GCD}(n,6) = 1$	212
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNoGCD6ConstructionTests	
Tests for SOLSWithNoGCD6Construction	213
TournamentPlanner.Startup	214
TournamentPlanner.Model.Designs.BlockDesigns.STS	
Class representing a Steiner triple system. A Steiner triple system is a BIBD with $K = 3$ and $\text{Lambda} = 1$	214
TournamentPlanner.Model.Designs.BlockDesigns.TD	
Class representing the transversal design	216
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TD17ConstructionTests	
Tests for the TD17Construction	218
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction	
Construction of TD(k, g) that are then used to construct Whist tournaments	218

TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.TDWhistPlanner	
Same method as in the larger whist construction but with TD (which is also a GDD basically). TODO unite	220
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.TDWhistPlannerTests	
Tests for the TDWhistPlanner	221
TournamentPlanner.Model.Tournaments.ThreePlayerTournament	
Class representing a three player tournament	222
TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournamentsModel	
Page model for three player tournaments	224
TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin	
Represents a 2 player round robin tournament schedule	225
TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlanner	
Tournament planner for two player round robins	227
TournamentPlanner.Pages.PageModels.Tournaments.TwoPlayerTournamentModel	
Page model for the construction of two player round robin tournaments	230
TournamentPlanner.Model.Tournaments.UniformGameSizeTournament	
Model for the tournament, where the sizes of the games are uniform - all the games have k- players	231
TournamentPlanner.Algorithms.TournamentPlanners.UniformGameSizeTournamentPlanner	
CP approach to the planning of tournaments, where the game sizes are uniform	233
TournamentPlannerTests.Algorithms.TournamentPlanners.UniformGameSizeTournamentPlannerTests	
Tests for the UniformGameSizeTournamentPlanner	234
TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel	
Page model for the tournament, in which the games are uniformly sized	235
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.Whist152Planner	
Class for the construction of Whist 152. Uses even SAMDRR	237
TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.Whist152PlannerTests	
Tests for the Whist152Planner	238
TournamentPlanner.Model.Tournaments.WhistTournament	
The model of the whist tournament	239
TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel	
Page model for the Whist tournaments	241
TournamentPlannerTests.Model.Tournaments.WhistTournamentValidator	
A validator for whist tournaments	242
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.WilsonGDDConstruction	
Wilson's construction methods using smaller GDDs to construct bigger ones. https↔ ://www.semanticscholar.org/paper/An-Existence-Theory-for-Pairwise-Balanced-Designs-↔ Wilson/b9c1c6bb1d7828b4f47f2ee3330dab26adb946dc	243
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.WilsonGDDConstructionTests	
Tests for the WilsonGDDConstruction	244

Chapter 6

Namespace Documentation

6.1 TournamentPlanner Namespace Reference

Classes

- class [Program](#)
- class [Startup](#)

6.2 TournamentPlanner.Algorithms Namespace Reference

6.3 TournamentPlanner.Algorithms.ConstructionAlgorithms Namespace Reference

Classes

- interface [IDesignConstruction](#)
Interface for the design construction algorithms.

6.4 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block↔ Designs Namespace Reference

6.5 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block↔ Designs.BIBDConstruction Namespace Reference

Classes

- class [BaseBIBDConstruction](#)
Base class for the BIBD construction algorithms.
- class [BoseSTSConstruction](#)

Bose's algorithm for the construction of STS of size $V = 6N + 3$. <http://math.ucdenver.edu/~wcherowi/courses/m6406/sts.pdf>

- class [IncrementalBIBDConstruction](#)
Incremental method that constructs BIBDs block by block. Implementation of the algorithm described in https://www.researchgate.net/publication/220660995_A_mathematical_programming_approach_to_the_construction_of_BIBDs. Uses CP-SAT as solver.
- class [MooreRBIBDConstruction](#)
Construction of RBIBD($3q+1, 4, 1$) where q is a prime power and can be written as $q = 4m + 1$. These RBIBD are used to construct Whist tournaments.
- class [QuadraticProgrammingBIBDConstruction](#)
A quadratic programming approach to the construction of BIBDs as described in the thesis. Uses CP-SAT as solver.
- class [RecursiveKTSConstruction](#)
Algorithm for the recursive construction of KTS of order $6N + 3$. Method described in <https://books.google.com/books?id=OoRFisqn-aIC>. KTSs of orders 9, 15, 21 and 39 are constructed statically, all the others are constructed in a recursive manner.
- class [SkolemSTSConstruction](#)
Skolem construction algorithm for STS of size $6N + 1$. <http://math.ucdenver.edu/~wcherowi/courses/m6406/sts.pdf>
- class [SmallBIBDForWhistConstruction](#)
Static construction of BIBD($21, 5, 1$) and BIBD($25, 5, 1$). These designs have been constructed by the incremental / quadratic methods. Those methods are, however, unreliable (license and memory issues) so the constructed designs are stored here for faster and more reliable access.
- class [SmoothKTSConstruction](#)
Algorithm for construction of smooth Kirkman triple systems. The size of smooth KTS is a power of 3. <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.501.8928&rep=rep1&type=pdf>

6.6 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block Designs.GDDConstruction Namespace Reference

Classes

- class [GDDForWhistConstruction](#)
Construction of GDD($17g, + 4u + 4v, \{5, 17\}, \{g, g + 4u, g + 4v\}$). These are used to construct whist tournaments.
- class [Resolvable3GDDConstruction](#)
Construct resolvable group divisible designs with block size 3 using MOLS. Should be able to construct GDDs where the order is a multiple of 3 except 6 and 18.
- class [SmallGDDConstruction](#)
Construct a few small order GDDs which are then used in Wilsons fundamental construction.
- class [WilsonGDDConstruction](#)
Wilson's construction methods using smaller GDDs to construct bigger ones. <https://www.semanticscholar.org/paper/An-Existence-Theory-for-Pairwise-Balanced-Designs-Wilson/b9c1c6bb1d7828b4f47f2ee3330dab26adb946dc>

6.7 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block Designs.PBDConstruction Namespace Reference

Classes

- class [GeneralPBD3NPlus1Construction](#)
Class that is used for construction of any PBD($3n+1$). It uses the required depending on the order. It is also used in the recursive step of the [LargerOrderPBDConstruction](#).

- class [LargerOrderPBDConstruction](#)
Construction of larger order PBDs, where the order is $V = 3N + 1$.
- class [PBD34Construction](#)
The construction of PBD 34 was supposed to be in the [SmallOrderPBDConstruction](#), but was a bit too complicated so i extracted it to its custom class. For the construction of PBD34 a PBD27 needs to be constructed first. Then it is modified and new blocks are added to get a PBD34.
- class [QuasigroupPBDConstruction](#)
Method for construction of PBDs of orders 13, 16 and 28. This method uses pairs of MOLS as quasigroups.
- class [SmallOrderPBDConstruction](#)
A class facilitating the construction of PBDs of smaller orders in form $3N + 1$. All the constructed design have block sizes in $\{4, 7, 10, 19\}$. There are several construction methods, not a single unified one.

6.8 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block↔ Designs.TDConstruction Namespace Reference

Classes

- class [RTDConstruction](#)
Construction method for resolvable transversal designs.
- class [TDConstruction](#)
Construction of $TD(k, g)$ that are then used to construct Whist tournaments.

6.9 TournamentPlanner.Algorithms.ConstructionAlgorithms.Latin↔ Squares Namespace Reference

Classes

- class [CommutativeIdempotentLatinSquareConstruction](#)
Algorithm for the creation of commutative idempotent Latin squares of odd order
- class [SingleLatinSquareConstruction](#)
Recursive construction of a single Latin square of any order.

6.10 TournamentPlanner.Algorithms.ConstructionAlgorithms.Latin↔ Squares.OA Namespace Reference

Classes

- class [OrthogonalArrayConstruction](#)
Construction method for OAs using MOLS.

6.11 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal Namespace Reference

Classes

- class [BaseMOLSConstruction](#)
Base class for the mutually orthogonal latin square construction algorithms.
- class [ConstraintProgrammingMOLSPairConstruction](#)
Constraint a pair of MOLS using constraint programming approach with CP-SAT solver. This method only works for MOLS of small orders.
- class [DatastoreMOLSConstruction](#)
This class creates MOLS objects based on the ones in the data store. This is not a construction algorithm per se, but acts like one. It implements the same interface as the other MOLS construction methods so it is easier to use in other algorithms.
- class [DifferenceMOLSPairConstruction](#)
Construct a pair of MOLS of order $3M + 1$. This way, MOLS for all orders $12t + 10$ (which is a subset of $4k + 2$) can be constructed. This method was introduced in [https://www.math.uci.edu/~brusso/\[14\]BosShrParCJM1960.pdf](https://www.math.uci.edu/~brusso/[14]BosShrParCJM1960.pdf).
- class [FourKMOLSPairConstruction](#)
*Construction of MOLS of order $V = 4 * K$.*
- class [GeneralMOLSPairConstruction](#)
Class facilitating the construction of MOLS of all the supported orders. Uses other construction algorithms.
- class [OddMOLSPairConstruction](#)
Algorithm to construct pairs of mutually orthogonal Latin squares of odd order. https://www.whitman.edu/mathematics/cgt_online/book/section04.03.html

6.12 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal Namespace Reference

Classes

- class [SOLSWithNoGCD6Construction](#)
Construct a cyclic SOLS such that $GCD(n,6) = 1$.

6.13 TournamentPlanner.Algorithms.Testing Namespace Reference

Classes

- class [PairwiseTestCasesConstruction](#)
Class implementing the construction of the pairwise test cases using orthogonal arrays.

6.14 TournamentPlanner.Algorithms.TournamentPlanners Namespace Reference

Classes

- class [Balanced3PlayerTournamentPlanner](#)
Plane a balanced three player tournament. Perfectly balanced three player tournament can only be planed, if the number of players $V = 6N + 3$.
- class [BipartiteTournamentPlanner](#)
Bipartite tournament planner using Latin squares.
- interface [ITournamentPlanner](#)
Interface for the tournament planners.
- class [PartiallyBalanced3PlayerTournamentPlanner](#)
This class facilitates the construction of partially balanced three player tournaments. The number of players in the tournament has to be divisible by 3. For the construction of the tournament, GDD are used.
- class [TwoPlayerRoundRobinPlanner](#)
Tournament planner for two player round robins.
- class [UniformGameSizeTournamentPlanner](#)
CP approach to the planning of tournaments, where the game sizes are uniform.

6.15 TournamentPlanner.Algorithms.TournamentPlanners.Spouses Namespace Reference

Classes

- class [SAMDRR38Planner](#)
SAMDRR 38 planner. $GCD(38, 6) \neq 1$ so a special construction is required. SAMDRR 38 is used to construct Whist(152).
- class [SAMDRR8Planner](#)
SAMDRR 8 construction class.
- class [SimpleSAMDRRPlanner](#)
SAMDRR planner using cyclic SOLS such that for the SOLS order n , $GCD(n, 6) = 1$.

6.16 TournamentPlanner.Algorithms.TournamentPlanners.Whist Namespace Reference

Classes

- class [BaseWhistPlanner](#)
The base class for whist tournament planners.

6.17 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even Namespace Reference

Classes

- class [CyclicWhistPlanner](#)
Cyclic construction method for several orders of [Whist](#)($V = 4N$).
- class [EvenEvenProductWhistPlanner](#)
Construction of a [Whist](#) tournament as a product of 2 existing $Wh(4n)$ tournaments.
- class [EvenOddProductWhistPlanner](#)
Construction of a [Whist](#) tournament as a product of a $Wh(4n)$ and $Wh(4n + 1)$.
- class [GeneralEvenWhistPlanner](#)
Plan a whist tournament of order $4n$ or $4n + 1$ using some of the implemented approaches. This class basically just selects the method to use for the construction.
- class [LargerWhistPlanner](#)
Construct $Wh(4n)$, where $n > 80$.
- class [MooreWhistPlanner](#)
This class implements the construction of $Wh(v = 4n)$, where $v = 3q + 1$ and $q = 1 \pmod{4} = 4m + 1$ and is a prime power. $v = 28, 40, 52, 76, 88 \dots$ https://www.jstor.org/stable/2369860?seq=1::metadata_info_tab_contents.
- class [OddSAMDRRWhistPlanner](#)
Use odd order SAMMDR to construct several orders od $Whist(4n)$.
- class [TDWhistPlanner](#)
Same method as in the larger whist construction but with TD (which is also a GDD basically). TODO unite.
- class [Whist152Planner](#)
Class for the construction of [Whist](#) 152. Uses even SAMDRR.

6.18 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd Namespace Reference

Classes

- class [CyclicOddWhistPlanner](#)
Cyclic odd whist planner for $v = 33, 57$.
- class [GeneralOddWhistPlanner](#)
Construct some [Whist](#) tournaments of order $4n + 1$. Not all orders are supported unfortunately.
- class [PBDWhistPlanner](#)
Class facilitating the construction of $Whist(21)$ and $Whist(69)$ using PBDs.
- class [PrimePowerWhistPlanner](#)
[Whist](#) tournament planner for prime power number of players in the form of $v = 4n + 1$. It uses finite fields for the tournament construction.

6.19 TournamentPlanner.Algorithms.Utils Namespace Reference

Classes

- class [OrthogonalArrayMOLSConverter](#)
Static class facilitating the conversion between MOLS and orthogonal arrays. MOLS and OAs are equivalent structures for some parameter values.

6.20 TournamentPlanner.Datastore Namespace Reference

Classes

- class **DataReader**
Static class that facilitates the access to the data stored in file or MongoDB.
- class **MongoMOLS**
Class representing a MOLS in mongo DB.

6.21 TournamentPlanner.Model Namespace Reference

6.22 TournamentPlanner.Model.Designs Namespace Reference

Classes

- interface **IDesign**
Common interface for all the design structures.
- interface **IResolvableDesign**
Interface for resolvable designs.
- class **OrthogonalArray**
Class representing the Orthogonal array structure. In some literature, this representation is actually the transpose of an orthogonal array.

6.23 TournamentPlanner.Model.Designs.BlockDesigns Namespace Reference

Classes

- class **BaseBlockDesign**
Base class for the block designs.
- class **BIBD**
Class representing a balanced incomplete block design.
- class **GDD**
Class representing a group divisible design.
- class **KTS**
*A class representing a Kirkman triple system. A **KTS** is a resolvable **BIBD**($V, 3, 1$), where $V = 6N + 3$. Basically, a resolvable **STS** is a **KTS**.*
- class **PBD**
*Class representing a Pairwise balanced design. **PBD** is similar to **BIBD**, but the block sizes are from K - they don't have to be uniform. We only consider **PBDs** with $\lambda = 1$.*
- class **RBIBD**
Class representing a resolvable balanced incomplete blocks design.
- class **RTD**
Resolvable transversal design.
- class **STS**
*Class representing a Steiner triple system. A Steiner triple system is a **BIBD** with $K = 3$ and $\lambda = 1$.*
- class **TD**
Class representing the transversal design.

6.24 TournamentPlanner.Model.Designs.LatinSquares Namespace Reference

Classes

- class [LatinSquare](#)
The Latin square.
- class [MOLS](#)
Set of mutually orthogonal Latin squares. For simplicity, it is allowed for the set of [MOLS](#) to contain only a single Latin square.
- class [SOLS](#)
Self orthogonal Latin square.

6.25 TournamentPlanner.Model.Testing Namespace Reference

Classes

- class [PairwiseTestCases](#)
A class representing the pairwise test cases created using orthogonal arrays a some post processing methods.

6.26 TournamentPlanner.Model.Tournaments Namespace Reference

Classes

- class [BaseTournament](#)
Base abstract class for the tournaments.
- class [BipartiteTournament](#)
Class representing a bipartite tournament. In a bipartite tournament, there are 2 teams with the same number of players. The goal is, that every player from the first team should play a game with every player from the second team.
- class [SAMDRR](#)
Spouse-avoiding mixed doubles round robin tournament. Games are represented as 4 integers, $(i, l, j, k) - H_i W_l v H_j W_k$
- class [ThreePlayerTournament](#)
Class representing a three player tournament.
- class [TwoPlayerRoundRobin](#)
Represents a 2 player round robin tournament schedule.
- class [UniformGameSizeTournament](#)
Model for the tournament, where the sizes of the games are uniform - all the games have k-players.
- class [WhistTournament](#)
The model of the whist tournament.

6.27 TournamentPlanner.Pages Namespace Reference

Classes

- class [Pages__ViewImports](#)
- class [Pages__ViewStart](#)
- class [Pages_Error](#)
- class [Pages_Index](#)
- class [Pages_Privacy](#)
- class [ErrorModel](#)
- class [IndexModel](#)
- class [PrivacyModel](#)

6.28 TournamentPlanner.Pages.PageModels Namespace Reference

Classes

- class [BasePageModel](#)
The base class for the page models of designs and tournaments.

6.29 TournamentPlanner.Pages.PageModels.Designs Namespace Reference

Classes

- class [Pages_PageModels_Designs_BIBD](#)
- class [Pages_PageModels_Designs_GDD](#)
- class [Pages_PageModels_Designs_KTS](#)
- class [Pages_PageModels_Designs_MOLS](#)
- class [Pages_PageModels_Designs_PBD](#)
- class [BIBDModel](#)
The page model for BIBDs.
- class [GDDModel](#)
Page model for resolvable group divisible designs with block size 3.
- class [KTSModel](#)
Page model for the Kirkman triple system creation.
- class [MOLSModel](#)
Page model for mutually orthogonal Latin squares.
- class [PBDMModel](#)
Page model for the pairwise balanced design. Construction of PBD of orders $3M + 1$ is implemented.

Enumerations

- enum class [Method](#) { [QuadraticProgramming](#) , [BoseBlockBuilding](#) , [SkolemBlockBuilding](#) , [IncrementalBlockBuilding](#) }
- Enum of the possible construction methods.*
- enum class [MOLSMethod](#) { [Automatic](#) , [OddOrderMOLS](#) , [FourKOrder](#) , [CPApproach](#) , [DifferenceApproach](#) }
- Enum of the possible construction methods.*

6.29.1 Enumeration Type Documentation

6.29.1.1 Method

enum `TournamentPlanner.Pages.PageModels.Designs.Method` [strong]

Enum of the possible construction methods.

Enumerator

QuadraticProgramming	Quadratic programming construction method.
BoseBlockBuilding	Bose STS construction method.
SkolemBlockBuilding	Skolem STS construction method.
IncrementalBlockBuilding	Incremental BIBD construction method.

6.29.1.2 MOLSMethod

enum `TournamentPlanner.Pages.PageModels.Designs.MOLSMethod` [strong]

Enum of the possible construction methods.

Enumerator

Automatic	Select the method according to the order automatically.
OddOrderMOLS	Construction method for MOLS of odd order.
FourKOrder	Construction method for MOLS of order 4K.
CPApproach	Construction method using constraint programming.
DifferenceApproach	Construction method using the difference set approach.

6.30 TournamentPlanner.Pages.PageModels.Testing Namespace Reference

Classes

- class [Pages_PageModels_Testing_PairwiseTesting](#)
- class [PairwiseTestingModel](#)

Page model for the construction of pairwise tests.

6.31 TournamentPlanner.Pages.PageModels.Tournaments Namespace Reference

Classes

- class [Pages_PageModels_Tournaments_BipartiteTournament](#)
- class [Pages_PageModels_Tournaments_ThreePlayerTournaments](#)
- class [Pages_PageModels_Tournaments_TwoPlayerTournament](#)
- class [Pages_PageModels_Tournaments_UniformGames](#)
- class [Pages_PageModels_Tournaments_WhistTournaments](#)
- class [BipartiteTournamentModel](#)
Page model for the construction of bipartite tournaments.
- class [ThreePlayerTournamentsModel](#)
Page model for three player tournaments.
- class [TwoPlayerTournamentModel](#)
Page model for the construction of two player round robin tournaments.
- class [UniformGamesModel](#)
Page model for the tournament, in which the games are uniformly sized.
- class [WhistTournamentsModel](#)
Page model for the Whist tournaments.

6.32 TournamentPlanner.Pages.Shared Namespace Reference

Classes

- class [Pages_Shared__Layout](#)
- class [Pages_Shared__ValidationScriptsPartial](#)

6.33 TournamentPlanner.Pages.Tournament Namespace Reference

Classes

- class [Pages_Tournament_BipartiteTournament](#)
- class [Pages_Tournament_TwoPlayerTournament](#)

6.34 TournamentPlanner.Utils Namespace Reference

Classes

- class **MultidimensionalArrayExtension**
Extension methods of 2D multidimensional arrays to get rows and columns as single 1D arrays..
- class **PrimeFactorization**
Static class providing factorization methods.

6.35 TournamentPlannerTests Namespace Reference

6.36 TournamentPlannerTests.Algorithms Namespace Reference

6.37 TournamentPlannerTests.Algorithms.ConstructionAlgorithms Namespace Reference

6.38 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.[↔](#) BlockDesigns Namespace Reference

6.39 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.[↔](#) BlockDesigns.BIBDConstruction Namespace Reference

Classes

- class [BoseSTSCONSTRUCTIONTests](#)
Tests for the BoseSTSCONSTRUCTION.
- class [GurobiIncrementalBIBDConstructionTests](#)
Tests for the GurobiIncrementalBIBDConstruction.
- class [GurobiQuadraticProgrammingBIBDConstructionTests](#)
Tests for the QuadraticProgrammingBIBDConstruction.
- class [IncrementalBIBDConstructionTests](#)
Test for the IncrementalBIBDConstruction.
- class [MooreRBIBDConstructionTests](#)
Tests for the MooreRBIBDConstruction.
- class [QuadraticProgrammingBIBDConstructionTests](#)
Tests for the QuadraticProgrammingBIBDConstruction.
- class [RecursiveKTSCONSTRUCTIONTests](#)
Tests for the RecursiveKTSCONSTRUCTION.
- class [SkolemSTSCONSTRUCTIONTests](#)
Tests for the SkolemSTSCONSTRUCTION.
- class [SmallBIBDForWhistConstructionTests](#)
Tests for SmallBIBDForWhistConstruction.
- class [SmoothKTSCONSTRUCTIONTests](#)
Tests for the SmoothKTSCONSTRUCTION.

6.40 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.[↔](#) BlockDesigns.GDDConstruction Namespace Reference

Classes

- class [GDDForWhistConstructionTests](#)
Tests for the GDDForWhistConstruction.
- class [Resolvable3GDDConstructionTests](#)
Tests the creation of resolvable GDDs by Resolvable3GDDConstruction.
- class [SmallGDDConstructionTests](#)
Tests for the SmallGDDConstruction.
- class [WilsonGDDConstructionTests](#)
Tests for the WilsonGDDConstruction.

6.41 TournamentPlannerTests.Algorithms.ConstructionAlgorithms. BlockDesigns.PBDConstruction Namespace Reference

Classes

- class [LargerOrderPBDConstructionTests](#)
Tests for the LargerOrderPBDConstruction.
- class [PBD34ConstructionTest](#)
- class [QuasigroupPBDConstructionTests](#)
Tests for the QuasigroupPBDConstruction.
- class [SmallOrderPBDConstructionTests](#)
Tests for the SmallOrderPBDConstruction.

6.42 TournamentPlannerTests.Algorithms.ConstructionAlgorithms. BlockDesigns.TDConstruction Namespace Reference

Classes

- class [RTDConstructionTests](#)
Tests for the RTDConstruction.
- class [TD17ConstructionTests](#)
Tests for the TD17Construction.

6.43 TournamentPlannerTests.Algorithms.ConstructionAlgorithms. LatinSquares Namespace Reference

Classes

- class [SingleLatinSquareConstructionTests](#)
Tests for the SingleLatinSquareConstruction.

6.44 TournamentPlannerTests.Algorithms.ConstructionAlgorithms. LatinSquares.Orthogonal Namespace Reference

Classes

- class [ConstraintProgrammingMOLSPairConstructionTests](#)
Tests for the ConstraintProgrammingMOLSPairConstruction.
- class [DatastoreMOLSCONSTRUCTIONTests](#)
Tests for the DatastoreMOLSCONSTRUCTION.
- class [DifferenceMOLSPairConstructionTests](#)
Tests for the DifferenceMOLSPairConstruction.
- class [FourKMOLSPairConstructionTests](#)
Tests for the FourKMOLSPairConstruction.
- class [OddMOLSPairConstructionTests](#)
Tests for the OddMOLSPairConstruction.

6.45 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal Namespace Reference

Classes

- class [SOLSWithNoGCD6ConstructionTests](#)
Tests for SOLSWithNoGCD6Construction.

6.46 TournamentPlannerTests.Algorithms.Testing Namespace Reference

Classes

- class [PairwiseTestCasesConstructionTests](#)

6.47 TournamentPlannerTests.Algorithms.TournamentPlanners Namespace Reference

Classes

- class [UniformGameSizeTournamentPlannerTests](#)
Tests for the UniformGameSizeTournamentPlanner.

6.48 TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses Namespace Reference

Classes

- class [SAMDRR38PlannerTests](#)
Tests for the SAMDRR38Planner.
- class [SAMDRR8PlannerTests](#)
Tests for the SAMDRR8Planner.
- class [SimpleSAMDRRPlannerTests](#)
Tests for the SimpleSAMDRRPlanner.

6.49 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist Namespace Reference

6.50 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.↔ Even Namespace Reference

Classes

- class [CyclicWhistPlannerTests](#)
Tests for the CyclicWhistPlanner.
- class [EvenEvenProductWhistPlannerTests](#)
Tests for the ProductWhistPlanner.
- class [EvenOddProductWhistPlannerTests](#)
Tests for the EvenOddProductWhistPlanner.
- class [GeneralEvenWhistPlannerTests](#)
Tests for the General4NWhistPlanner.
- class [LargerWhistPlannerTests](#)
Tests for the LargerWhistPlanner.
- class [MooreWhistPlannerTests](#)
Tests for the MooreWhistPlanner.
- class [OddSAMDRRWhistPlannerTests](#)
Tests for the SAMDRRWhistPlanner.
- class [TDWhistPlannerTests](#)
Tests for the TDWhistPlanner.
- class [Whist152PlannerTests](#)
Tests for the Whist152Planner.

6.51 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.↔ Odd Namespace Reference

Classes

- class [CyclicOddWhistPlannerTests](#)
Tests for the CyclicOddWhistPlanner.
- class [PBDWhistPlannerTests](#)
Tests for the PBDWhistPlannerTests.
- class [PrimePowerWhistPlannerTests](#)
Tests for the PrimeNWhistPlanner.

6.52 TournamentPlannerTests.Datastore Namespace Reference

Classes

- class [DataReaderTests](#)
Tests for the DataReader.

6.53 TournamentPlannerTests.Designs Namespace Reference

Classes

- class [MOLSSetValidatorTests](#)
Tests for the MOLSSetValidator.

6.54 TournamentPlannerTests.Model Namespace Reference

6.55 TournamentPlannerTests.Model.Designs Namespace Reference

Classes

- class **ResolvableDesignValidator**
Validator for resolvable designs.

6.56 TournamentPlannerTests.Model.Designs.BlockDesigns Namespace Reference

Classes

- class **BIBDValidator**
A static class providing methods for BIBD validation.
- class **GDDValidator**
A static class for validating GDDs.
- class **PBDValidator**
A static class for validating PBDs.
- class **TDValidator**
Validator for transversal designs.

6.57 TournamentPlannerTests.Model.Designs.LatinSquares Namespace Reference

Classes

- class **LatinSquareValidator**
A static class to validate the properties of a Latin square.
- class **MOLSValidator**
Validator for MOLS. Validates the mutually orthogonality of all the Latin squares in the set.

6.58 TournamentPlannerTests.Model.Tournaments Namespace Reference

Classes

- class **SAMDRRValidator**
SAMDRR validator.
- class [WhistTournamentValidator](#)
A validator for whist tournaments.

6.59 TournamentPlannerTests.Utils Namespace Reference

Classes

- class [OrthogonalArrayMOLSCConverterTests](#)
Tests for the OrthogonalArrayMOLSCConverter.
- class [PrimeFactorizationTests](#)
Tests for the PrimeFactorization static class.

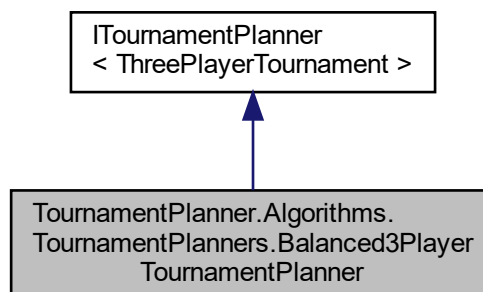
Chapter 7

Class Documentation

7.1 TournamentPlanner.Algorithms.TournamentPlanners.Balanced3PlayerTournamentPlanner Class Reference

Plane a balanced three player tournament. Perfectly balanced three player tournament can only be planed, if the number of players $V = 6N + 3$.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Balanced3PlayerTournamentPlanner:



Public Member Functions

- `Balanced3PlayerTournamentPlanner` (int v)
Initializes a new instance of the `Balanced3PlayerTournamentPlanner` class.
- `ThreePlayerTournament PlanTournament` ()

7.1.1 Detailed Description

Plane a balanced three player tournament. Perfectly balanced three player tournament can only be planed, if the number of players $V = 6N + 3$.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 `Balanced3PlayerTournamentPlanner()`

```
TournamentPlanner.Algorithms.TournamentPlanners.Balanced3PlayerTournamentPlanner.Balanced3PlayerTournamentPlanner (
    int v )
```

Initializes a new instance of the [Balanced3PlayerTournamentPlanner](#) class.

Parameters

<code>v</code>	Number of players.
----------------	--------------------

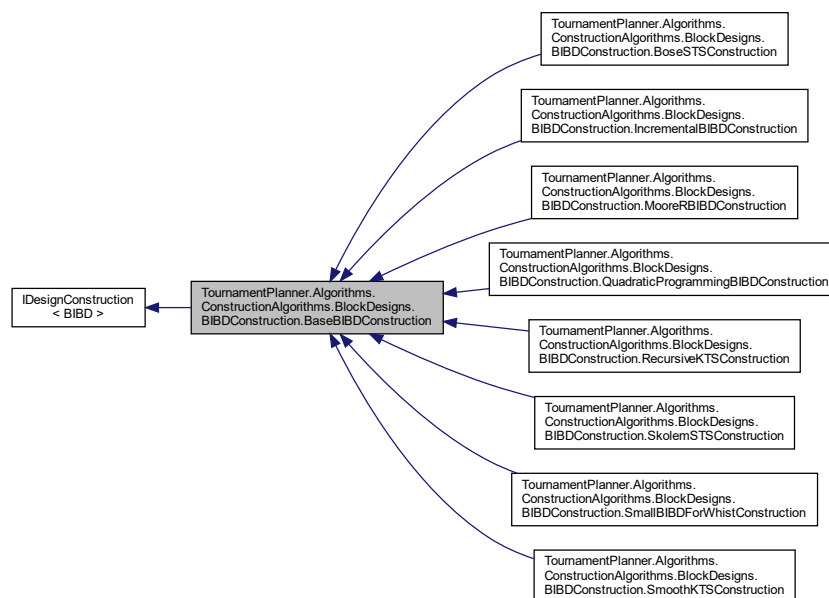
The documentation for this class was generated from the following file:

- `Balanced3PlayerTournamentPlanner.cs`

7.2 `TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction` Class Reference

Base class for the BIBD construction algorithms.

Inheritance diagram for `TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction`:



Public Member Functions

- abstract [BIBD ConstructDesign](#) ()
Construct the BIBD.

Protected Member Functions

- [BaseBIBDConstruction](#) (int v, int k, int lambda)
Initializes a new instance of the [BaseBIBDConstruction](#) class.

Properties

- int [V](#) [get]
Gets the number of elements in the BIBD to construct.
- int [K](#) [get]
Gets the size of a block in the BIBD to construct.
- int [Lambda](#) [get]
Gets the number of times each pair of elements should occur together.
- int [R](#) [get]
Gets the number of occurrences of an element in the blocks.
- int [B](#) [get]
Gets the supposed number of block the design should contain.

7.2.1 Detailed Description

Base class for the BIBD construction algorithms.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 BaseBIBDConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction.↔  
BaseBIBDConstruction (  
    int v,  
    int k,  
    int lambda ) [protected]
```

Initializes a new instance of the [BaseBIBDConstruction](#) class.

Parameters

<i>v</i>	Set size.
<i>k</i>	Block size.
<i>lambda</i>	Pair occurrences.

7.2.3 Member Function Documentation

7.2.3.1 ConstructDesign()

```
abstract BIBD TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔
BaseBIBDConstruction.ConstructDesign ( ) [pure virtual]
```

Construct the BIBD.

Returns

Constructed BIBD.

Implemented in [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmoothKTSConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmallBIBDForWhistConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SkolemSTSConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.RecursiveKTSConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.QuadraticProgrammingBIBDConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.MooreRBIBDConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.IncrementalBIBDConstruction](#), and [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BoseSTSConstruction](#).

7.2.4 Property Documentation

7.2.4.1 B

```
int TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Base↔
BIBDConstruction.B [get], [protected]
```

Gets the supposed number of block the design should contain.

7.2.4.2 K

```
int TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Base↔
BIBDConstruction.K [get], [protected]
```

Gets the size of a block in the BIBD to construct.

7.2.4.3 Lambda

```
int TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Base↔
BIBDConstruction.Lambda [get], [protected]
```

Gets the number of times each pair of elements should occur together.

7.2.4.4 R

```
int TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Base↔
BIBDConstruction.R [get], [protected]
```

Gets the number of occurrences of an element in the blocks.

7.2.4.5 V

```
int TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Base↔
BIBDConstruction.V [get], [protected]
```

Gets the number of elements in the BIBD to construct.

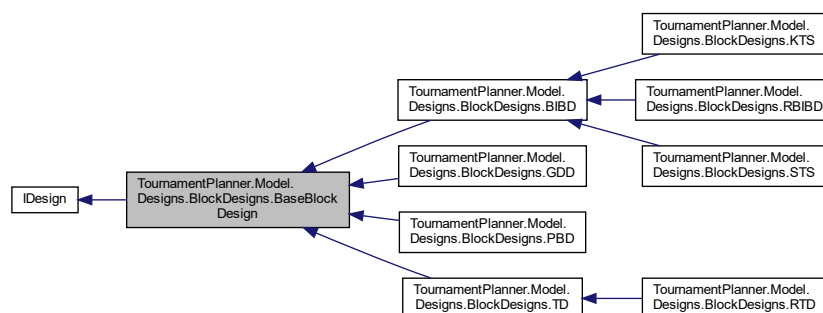
The documentation for this class was generated from the following file:

- BaseBIBDConstruction.cs

7.3 TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign Class Reference

Base class for the block designs.

Inheritance diagram for TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign:



Protected Member Functions

- [BaseBlockDesign](#) (int v)
Initializes a new instance of the [BaseBlockDesign](#) class.

Properties

- int **V** [get, protected set]
Gets or sets the number of elements (treatments) in the block design.
- List< List< int > > **Blocks** = new List<List<int>>() [get, protected set]
Gets or sets the blocks of the block design.

7.3.1 Detailed Description

Base class for the block designs.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 BaseBlockDesign()

```
TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign.BaseBlockDesign (
    int v ) [protected]
```

Initializes a new instance of the [BaseBlockDesign](#) class.

Parameters

v	Number of elements.
---	---------------------

7.3.3 Property Documentation

7.3.3.1 Blocks

```
List<List<int> > TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign.Blocks = new
List<List<int>>() [get], [protected set]
```

Gets or sets the blocks of the block design.

7.3.3.2 V

```
int TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign.V [get], [protected set]
```

Gets or sets the number of elements (treatments) in the block design.

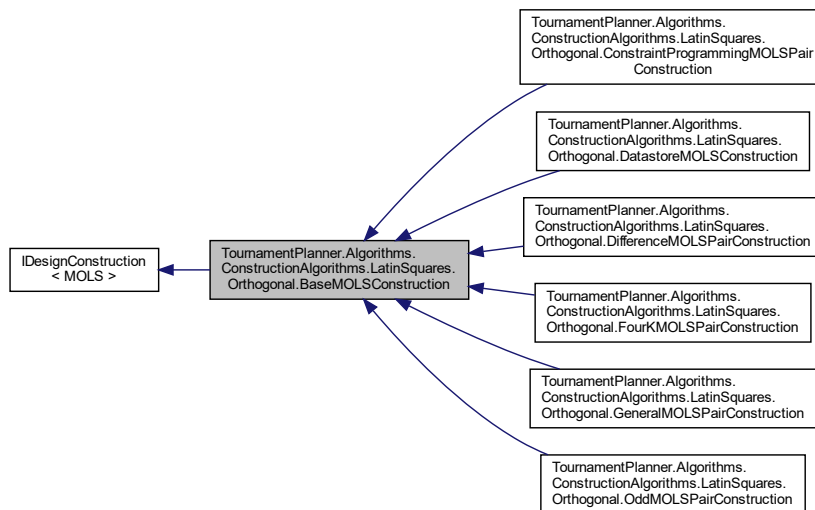
The documentation for this class was generated from the following file:

- BaseBlockDesign.cs

7.4 TournamentPlanner.Algorithms.ConstructionAlgorithms.Latin↔
 Squares.Orthogonal.BaseMOLSCONSTRUCTION Class Reference

Base class for the mutually orthogonal latin square construction algorithms.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Base↔
 MOLSCONSTRUCTION:



Public Member Functions

- abstract [MOLS ConstructDesign](#) ()
Construct MOLS.

Protected Member Functions

- [BaseMOLSCONSTRUCTION](#) (int order)
Initializes a new instance of the BaseMOLSCONSTRUCTION class.
- abstract bool [ValidateOrder](#) (int order)
Method to validate the order in the constructor.

Properties

- int [Order](#) [get]
Gets the order of the MOLS to construct.
- abstract string [ArgumentExceptionMessage](#) [get]
Gets the argument exception message of the MOLS construction algorithm.

7.4.1 Detailed Description

Base class for the mutually orthogonal latin square construction algorithms.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 BaseMOLSConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.BaseMOLSConstruction.↔
BaseMOLSConstruction (
    int order ) [protected]
```

Initializes a new instance of the [BaseMOLSConstruction](#) class.

Parameters

<i>order</i>	The order of the MOLS.
--------------	------------------------

7.4.3 Member Function Documentation

7.4.3.1 ConstructDesign()

```
abstract MOLS TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.↔
BaseMOLSConstruction.ConstructDesign ( ) [pure virtual]
```

Construct MOLS.

Returns

Constructed MOLS.

Implemented in [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPairConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.GeneralMOLSPairConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPairConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DifferenceMOLSPairConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DatastoreMOLSConstruction](#), and [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.ConstraintProgrammingMOLSPairConstruction](#).

7.4.3.2 ValidateOrder()

```
abstract bool TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.↔  
BaseMOLSCONSTRUCTION.ValidateOrder (  
    int order ) [protected], [pure virtual]
```

Method to validate the order in the constructor.

Parameters

<i>order</i>	The order to validate.
--------------	------------------------

Returns

True if the algorithm can construct MOLS of given order; otherwise false.

Implemented in [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPairConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.GeneralMOLSPairConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPairConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DifferenceMOLSPairConstruction](#), [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DatastoreMOLSCONSTRUCTION](#), and [TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.ConstraintProgrammingMOLSPairConstruction](#)

7.4.4 Property Documentation

7.4.4.1 ArgumentExceptionMessage

```
abstract string TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.↔  
BaseMOLSCONSTRUCTION.ArgumentExceptionMessage [get], [protected]
```

Gets the argument exception message of the MOLS construction algorithm.

7.4.4.2 Order

```
int TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.BaseMOLSCONSTRUCTION.↔  
Order [get], [protected]
```

Gets the order of the MOLS to construct.

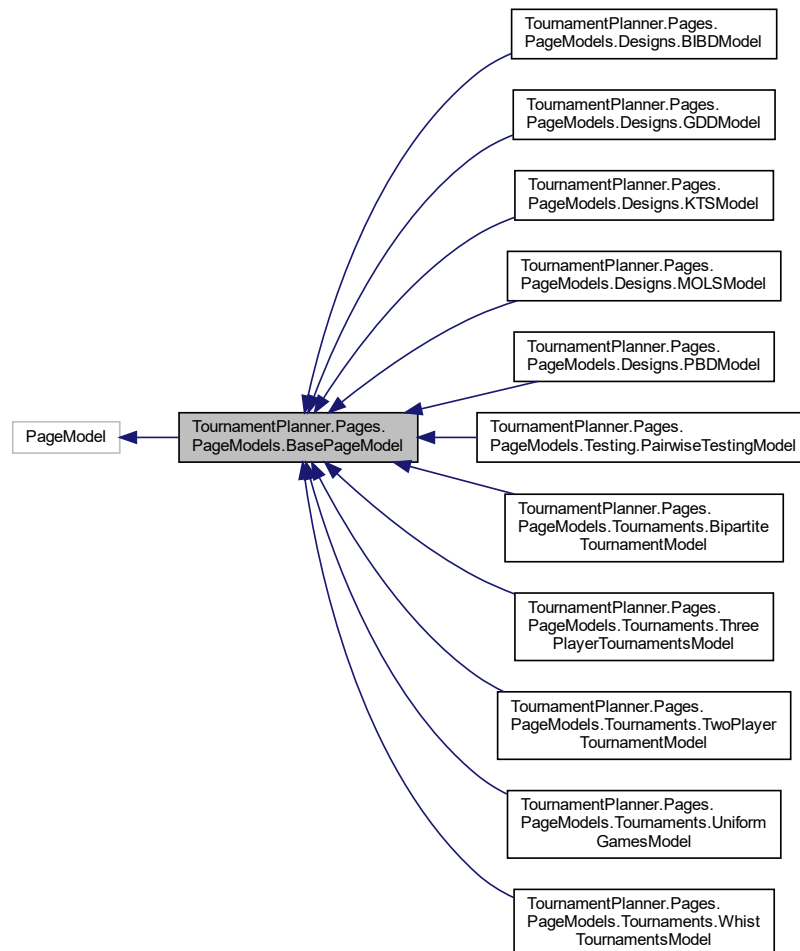
The documentation for this class was generated from the following file:

- BaseMOLSCONSTRUCTION.cs

7.5 TournamentPlanner.Pages.PageModels.BasePageModel Class Reference

The base class for the page models of designs and tournaments.

Inheritance diagram for TournamentPlanner.Pages.PageModels.BasePageModel:



Properties

- bool `DisplayResults` = false [get, protected set]
Gets or sets a value indicating whether to display the results.
- List< string > `ErrorMessage`s = new List<string>() [get]
Gets the error messages. If empty, there was no error. Otherwise, the error messages shall be displayed.

7.5.1 Detailed Description

The base class for the page models of designs and tournaments.

7.5.2 Property Documentation

7.5.2.1 DisplayResults

```
bool TournamentPlanner.Pages.PageModels.BasePageModel.DisplayResults = false [get], [protected set]
```

Gets or sets a value indicating whether to display the results.

7.5.2.2 ErrorMessages

```
List<string> TournamentPlanner.Pages.PageModels.BasePageModel.ErrorMessages = new List<string>() [get]
```

Gets the error messages. If empty, there was no error. Otherwise, the error messages shall be displayed.

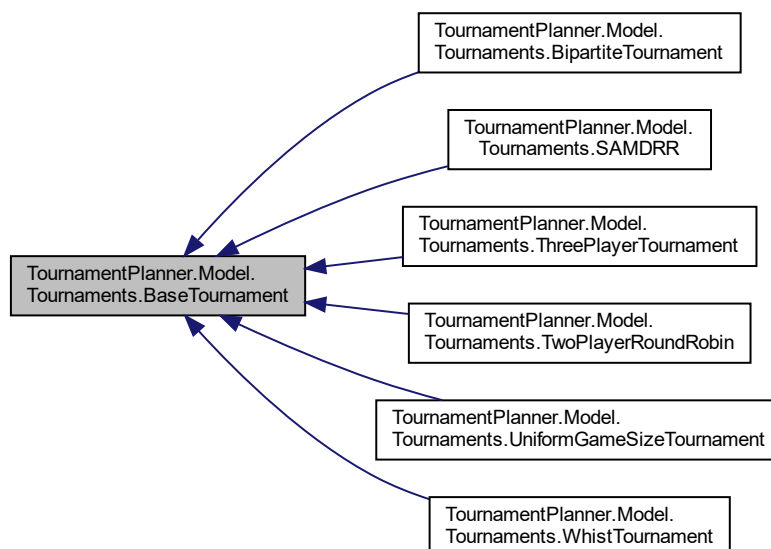
The documentation for this class was generated from the following file:

- BasePageModel.cs

7.6 TournamentPlanner.Model.Tournaments.BaseTournament Class Reference

Base abstract class for the tournaments.

Inheritance diagram for TournamentPlanner.Model.Tournaments.BaseTournament:



Public Member Functions

- void [AddRound](#) (List< List< int >> round)
Add round to the tournament.
- string [GameToString](#) (List< int > game)
Gets the string representation of the game.

Protected Member Functions

- [BaseTournament](#) (int order)
Initializes a new instance of the [BaseTournament](#) class.

Properties

- int [Order](#) [get, protected set]
Gets or sets the order of the tournament. The order is the number of players or 'teams'.
- List< List< List< int >> > [Rounds](#) = new List<List<List<int>>>() [get, protected set]
Gets or sets the round of the tournament.

7.6.1 Detailed Description

Base abstract class for the tournaments.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 BaseTournament()

```
TournamentPlanner.Model.Tournaments.BaseTournament.BaseTournament (
    int order ) [protected]
```

Initializes a new instance of the [BaseTournament](#) class.

Parameters

<i>order</i>	The order.
--------------	------------

7.6.3 Member Function Documentation

7.6.3.1 AddRound()

```
void TournamentPlanner.Model.Tournaments.BaseTournament.AddRound (
    List< List< int >> round )
```

Add round to the tournament.

Parameters

<i>round</i>	The round.
--------------	------------

7.6.3.2 GameToString()

```
string TournamentPlanner.Model.Tournaments.BaseTournament.GameToString (
    List< int > game )
```

Gets the string representation of the game.

Parameters

<i>game</i>	The game.
-------------	-----------

Returns

The string representation of the game.

7.6.4 Property Documentation

7.6.4.1 Order

```
int TournamentPlanner.Model.Tournaments.BaseTournament.Order [get], [protected set]
```

Gets or sets the order of the tournament. The order is the number of players or 'teams'.

7.6.4.2 Rounds

```
List<List<List<int> > > TournamentPlanner.Model.Tournaments.BaseTournament.Rounds = new
List<List<List<int>>>() [get], [protected set]
```

Gets or sets the round of the tournament.

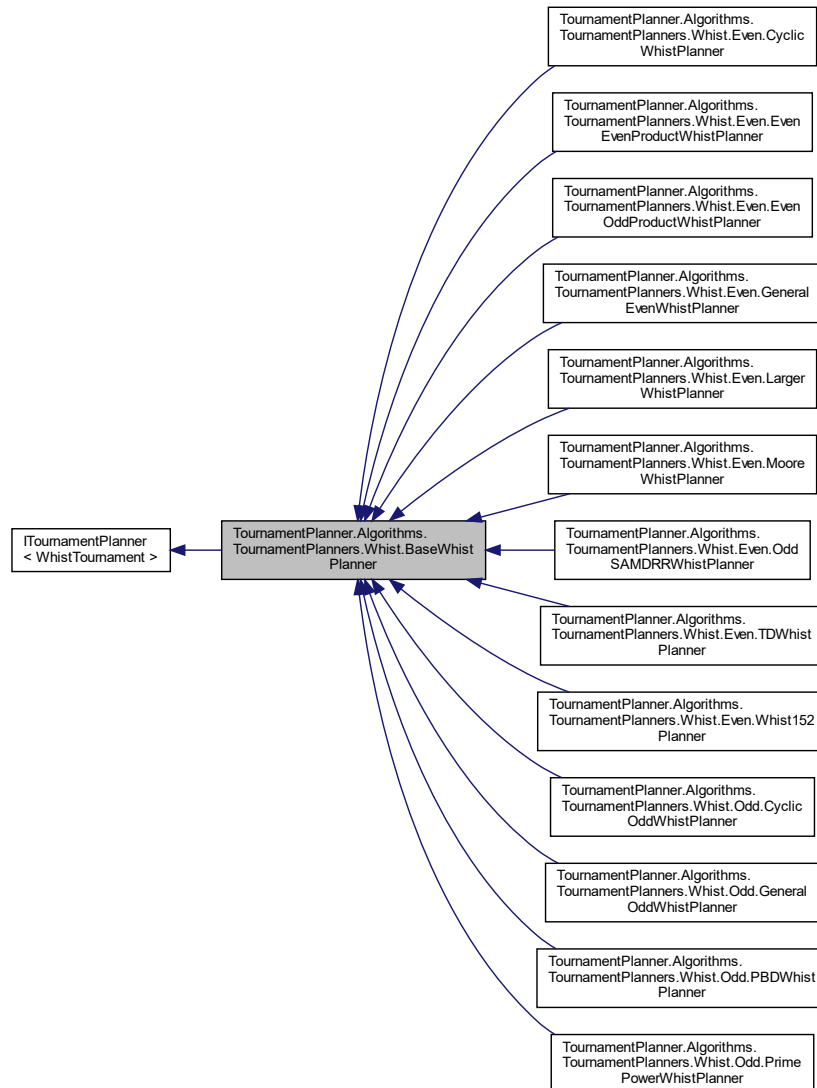
The documentation for this class was generated from the following file:

- BaseTournament.cs

7.7 TournamentPlanner.Algorithms.TournamentPlanners.Whist.BaseWhistPlanner Class Reference

The base class for whist tournament planners.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.BaseWhistPlanner:



Public Member Functions

- abstract [WhistTournament PlanTournament \(\)](#)
Plan the [Whist](#) tournament.

Protected Member Functions

- [BaseWhistPlanner \(int v\)](#)
Initializes a new instance of the [BaseWhistPlanner](#) class.

Properties

- `int V` [get, set]
Gets or sets the order of the tournament to be constructed.

7.7.1 Detailed Description

The base class for whist tournament planners.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 BaseWhistPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.Whist.BaseWhistPlanner.BaseWhistPlanner (
    int v ) [protected]
```

Initializes a new instance of the [BaseWhistPlanner](#) class.

Parameters

<code>v</code>	The order of the whist tournament.
----------------	------------------------------------

7.7.3 Member Function Documentation

7.7.3.1 PlanTournament()

```
abstract WhistTournament TournamentPlanner.Algorithms.TournamentPlanners.Whist.BaseWhistPlanner.PlanTournament ( ) [pure virtual]
```

Plan the [Whist](#) tournament.

Returns

The constructed whist tournament.

Implemented in [TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PrimePowerWhistPlanner](#), [TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlanner](#), [TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlanner](#), [TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.TDWhistPlanner](#), [TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlanner](#), [TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner](#), [TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlanner](#), and [TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlanner](#).

7.7.4 Property Documentation

7.7.4.1 V

```
int TournamentPlanner.Algorithms.TournamentPlanners.Whist.BaseWhistPlanner.V [get], [set],
[protected]
```

Gets or sets the order of the tournament to be constructed.

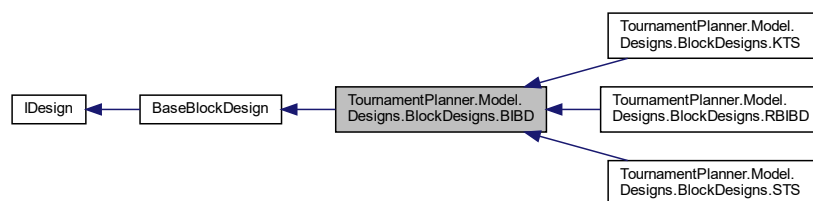
The documentation for this class was generated from the following file:

- BaseWhistPlanner.cs

7.8 TournamentPlanner.Model.Designs.BlockDesigns.BIBD Class Reference

Class representing a balanced incomplete block design.

Inheritance diagram for TournamentPlanner.Model.Designs.BlockDesigns.BIBD:



Public Member Functions

- [BIBD](#) (int v, int k, int lambda)
Initializes a new instance of the [BIBD](#) class.
- [BIBD](#) (int v, int k, int lambda, List< List< int >> blocks)
Initializes a new instance of the [BIBD](#) class.
- void [AddBlock](#) (List< int > block)
Add a single block to the list of blocks.
- void [AddBlocks](#) (List< List< int >> blocks)
Add multiple block at once.

Properties

- `int K` [get]
Gets the size of a block.
- `int Lambda` [get]
Gets the number of times every pair of elements occurs together in a block.
- `int R` [get]
Gets the number of occurrence of an element in a block. In a valid [BIBD](#), it has to be an integer. The check is done in the construction algorithm
- `int B` [get]
Gets the number of blocks. In a valid [BIBD](#), it has to be an integer. The check is done in the construction algorithm

Additional Inherited Members

7.8.1 Detailed Description

Class representing a balanced incomplete block design.

7.8.2 Constructor & Destructor Documentation

7.8.2.1 `BIBD()` [1/2]

```
TournamentPlanner.Model.Designs.BlockDesigns.BIBD.BIBD (
    int v,
    int k,
    int lambda )
```

Initializes a new instance of the [BIBD](#) class.

Parameters

<code>v</code>	Set size.
<code>k</code>	Block size.
<code>lambda</code>	Pair occurrences.

7.8.2.2 `BIBD()` [2/2]

```
TournamentPlanner.Model.Designs.BlockDesigns.BIBD.BIBD (
    int v,
    int k,
    int lambda,
    List< List< int >> blocks )
```

Initializes a new instance of the [BIBD](#) class.

Parameters

<i>v</i>	Set size.
<i>k</i>	Block size.
<i>lambda</i>	Pair occurrences.
<i>blocks</i>	Blocks of the design.

7.8.3 Member Function Documentation

7.8.3.1 AddBlock()

```
void TournamentPlanner.Model.Designs.BlockDesigns.BIBD.AddBlock (
    List< int > block )
```

Add a single block to the list of blocks.

Parameters

<i>block</i>	The block to add.
--------------	-------------------

7.8.3.2 AddBlocks()

```
void TournamentPlanner.Model.Designs.BlockDesigns.BIBD.AddBlocks (
    List< List< int >> blocks )
```

Add multiple block at once.

Parameters

<i>blocks</i>	The blocks to add.
---------------	--------------------

7.8.4 Property Documentation

7.8.4.1 B

```
int TournamentPlanner.Model.Designs.BlockDesigns.BIBD.B [get]
```

Gets the number of blocks. In a valid [BIBD](#), it has to be an integer. The check is done in the construction algorithm

7.8.4.2 K

```
int TournamentPlanner.Model.Designs.BlockDesigns.BIBD.K [get]
```

Gets the size of a block.

7.8.4.3 Lambda

```
int TournamentPlanner.Model.Designs.BlockDesigns.BIBD.Lambda [get]
```

Gets the number of times every pair of elements occurs together in a block.

7.8.4.4 R

```
int TournamentPlanner.Model.Designs.BlockDesigns.BIBD.R [get]
```

Gets the number of occurrence of an element in a block. In a valid [BIBD](#), it has to be an integer. The check is done in the construction algorithm

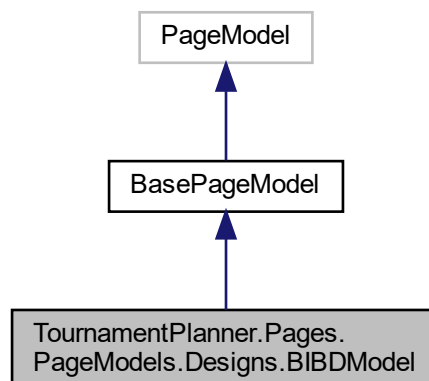
The documentation for this class was generated from the following file:

- BIBD.cs

7.9 TournamentPlanner.Pages.PageModels.Designs.BIBDModel Class Reference

The page model for BIBDs.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Designs.BIBDModel:



Public Member Functions

- void [OnPost](#) ()

Properties

- int [V](#) = 7 [get, set]
Gets or sets the size of the constructed BIBD.
- int [K](#) = 3 [get, set]
Gets or sets the size of a single block.
- int [L](#) = 1 [get, set]
Gets or sets the lambda of the BIBD.
- [Method BlockBuildingMethod](#) = [Method.IncrementalBlockBuilding](#) [get, set]
Gets or sets the method to use for the construction of BIBD.
- [BIBD BIBD](#) [get]
Gets the constructed BIBD.

7.9.1 Detailed Description

The page model for BIBDs.

7.9.2 Property Documentation

7.9.2.1 BIBD

[BIBD](#) `TournamentPlanner.Pages.PageModels.Designs.BIBDModel.BIBD` [get]

Gets the constructed BIBD.

7.9.2.2 BlockBuildingMethod

[Method](#) `TournamentPlanner.Pages.PageModels.Designs.BIBDModel.BlockBuildingMethod` = [Method.IncrementalBlockBuilding](#) [get], [set]

Gets or sets the method to use for the construction of BIBD.

7.9.2.3 K

`int` `TournamentPlanner.Pages.PageModels.Designs.BIBDModel.K` = 3 [get], [set]

Gets or sets the size of a single block.

7.9.2.4 L

```
int TournamentPlanner.Pages.PageModels.Designs.BIBDModel.L = 1 [get], [set]
```

Gets or sets the lambda of the BIBD.

7.9.2.5 V

```
int TournamentPlanner.Pages.PageModels.Designs.BIBDModel.V = 7 [get], [set]
```

Gets or sets the size of the constructed BIBD.

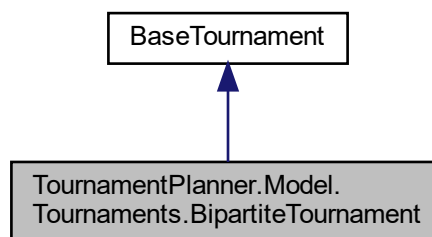
The documentation for this class was generated from the following file:

- BIBD.cshtml.cs

7.10 TournamentPlanner.Model.Tournaments.BipartiteTournament Class Reference

Class representing a bipartite tournament. In a bipartite tournament, there are 2 teams with the same number of players. The goal is, that every player from the first team should play a game with every player from the second team.

Inheritance diagram for TournamentPlanner.Model.Tournaments.BipartiteTournament:



Public Member Functions

- [BipartiteTournament](#) (int teamSize)
Initializes a new instance of the [BipartiteTournament](#) class.
- new string [GameToString](#) (List< int > game)

Additional Inherited Members

7.10.1 Detailed Description

Class representing a bipartite tournament. In a bipartite tournament, there are 2 teams with the same number of players. The goal is, that every player from the first team should play a game with every player from the second team.

7.10.2 Constructor & Destructor Documentation

7.10.2.1 BipartiteTournament()

```
TournamentPlanner.Model.Tournaments.BipartiteTournament.BipartiteTournament (
    int teamSize )
```

Initializes a new instance of the [BipartiteTournament](#) class.

Parameters

<i>teamSize</i>	The size of a team.
-----------------	---------------------

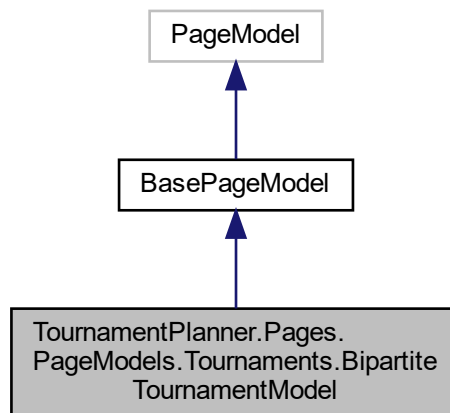
The documentation for this class was generated from the following file:

- BipartiteTournament.cs

7.11 TournamentPlanner.Pages.PageModels.Tournaments.Bipartite↔ TournamentModel Class Reference

Page model for the construction of bipartite tournaments.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournamentModel:



Public Member Functions

- void [OnPost](#) ()

Properties

- int [TeamSize](#) [get, set]
Gets or sets the size of teams.
- [BipartiteTournament Tournament](#) [get]
Gets the constructed tournament.

7.11.1 Detailed Description

Page model for the construction of bipartite tournaments.

7.11.2 Property Documentation

7.11.2.1 TeamSize

```
int TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournamentModel.TeamSize [get],  
[set]
```

Gets or sets the size of teams.

7.11.2.2 Tournament

`BipartiteTournament` `TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournamentModel`.↔
`Tournament` [get]

Gets the constructed tournament.

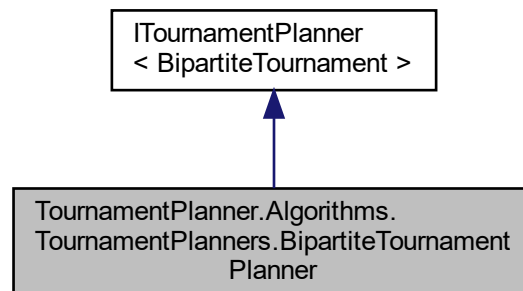
The documentation for this class was generated from the following file:

- `BipartiteTournament.cshtml.cs`

7.12 TournamentPlanner.Algorithms.TournamentPlanners.Bipartite↔ TournamentPlanner Class Reference

Bipartite tournament planner using Latin squares.

Inheritance diagram for `TournamentPlanner.Algorithms.TournamentPlanners.BipartiteTournamentPlanner`:



Public Member Functions

- `BipartiteTournamentPlanner` (`int teamSize`)
Initializes a new instance of the `BipartiteTournamentPlanner` class.
- `BipartiteTournament PlanTournament` ()
Plane a bipartite tournament.

7.12.1 Detailed Description

Bipartite tournament planner using Latin squares.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 BipartiteTournamentPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.BipartiteTournamentPlanner.BipartiteTournament↔  
Planner (   
     int teamSize )
```

Initializes a new instance of the [BipartiteTournamentPlanner](#) class.

Parameters

<i>teamSize</i>	Size of teams.
-----------------	----------------

7.12.3 Member Function Documentation

7.12.3.1 PlanTournament()

```
BipartiteTournament TournamentPlanner.Algorithms.TournamentPlanners.BipartiteTournament↔  
Planner.PlanTournament ( )
```

Plane a bipartite tournament.

Returns

A bipartite tournament.

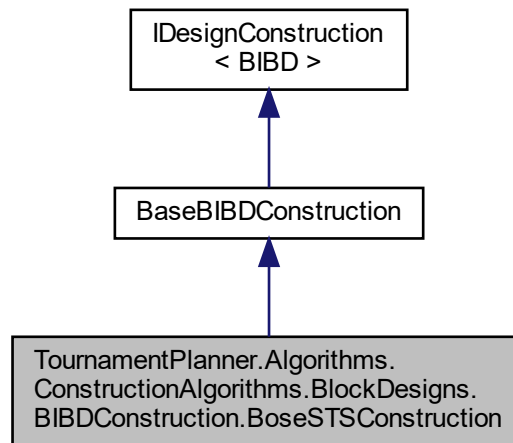
The documentation for this class was generated from the following file:

- [BipartiteTournamentPlanner.cs](#)

7.13 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block↔ Designs.BIBDConstruction.BoseSTSCONSTRUCTION Class Reference

Bose's algorithm for the construction of STS of size $V = 6N + 3$. <http://math.ucdenver.edu/~wcherowi/courses/m6406/sts.↔pdf>

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔
BoseSTSCONSTRUCTION:



Public Member Functions

- [BoseSTSCONSTRUCTION](#) (int v)
Initializes a new instance of the [BoseSTSCONSTRUCTION](#) class.
- override [STS ConstructDesign](#) ()
Construct STS of size $6N + 3$.

Additional Inherited Members

7.13.1 Detailed Description

Bose's algorithm for the construction of STS of size $V = 6N + 3$. <http://math.ucdenver.edu/~wcherowi/courses/m6406/sts.pdf>.↔

7.13.2 Constructor & Destructor Documentation

7.13.2.1 BoseSTSCONSTRUCTION()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BoseSTSCONSTRUCTION.↔
BoseSTSCONSTRUCTION (
    int v )
```

Initializes a new instance of the [BoseSTSCONSTRUCTION](#) class.

Parameters

v	Set size.
---	-----------

7.13.3 Member Function Documentation

7.13.3.1 ConstructDesign()

```
override STS TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔  
BoseSTSCONSTRUCTION.ConstructDesign ( ) [virtual]
```

Construct STS of size $6N + 3$.

Returns

The constructed STS.

Implements [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction](#).

The documentation for this class was generated from the following file:

- BoseSTSCONSTRUCTION.cs

7.14 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.[↔](#) BlockDesigns.BIBDConstruction.BoseSTSCONSTRUCTIONTests Class Reference

Tests for the BoseSTSCONSTRUCTION.

Public Member Functions

- void [BoseSTSCONSTRUCTIONTest](#) ()
Test for a few values of set size $6N + 3$.

7.14.1 Detailed Description

Tests for the BoseSTSCONSTRUCTION.

7.14.2 Member Function Documentation

7.14.2.1 BoseSTSCONSTRUCTIONTEST()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔
BoseSTSCONSTRUCTIONTESTS.BoseSTSCONSTRUCTIONTEST ( )
```

Test for a few values of set size $6N + 3$.

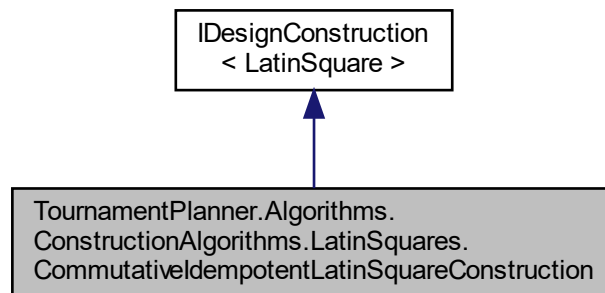
The documentation for this class was generated from the following file:

- BoseSTSCONSTRUCTIONTESTS.cs

7.15 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.CommutativeIdempotentLatinSquareConstruction Class Reference

Algorithm for the creation of commutative idempotent Latin squares of odd order

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.CommutativeIdempotentLatinSquareConstruction:



Public Member Functions

- [CommutativeIdempotentLatinSquareConstruction](#) (int order)
Initializes a new instance of the [CommutativeIdempotentLatinSquareConstruction](#) class.
- [LatinSquare ConstructDesign](#) ()
Construct a commutative idempotent Latin square of odd order.

7.15.1 Detailed Description

Algorithm for the creation of commutative idempotent Latin squares of odd order

7.15.2 Constructor & Destructor Documentation

7.15.2.1 CommutativeIdempotentLatinSquareConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.CommutativeIdempotentLatinSquareConstruction.CommutativeIdempotentLatinSquareConstruction (
    int order )
```

Initializes a new instance of the [CommutativeIdempotentLatinSquareConstruction](#) class.

Parameters

<i>order</i>	The LS order.
--------------	---------------

7.15.3 Member Function Documentation

7.15.3.1 ConstructDesign()

```
LatinSquare TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.CommutativeIdempotentLatinSquareConstruction.ConstructDesign ( )
```

Construct a commutative idempotent Latin square of odd order.

Returns

Latin square.

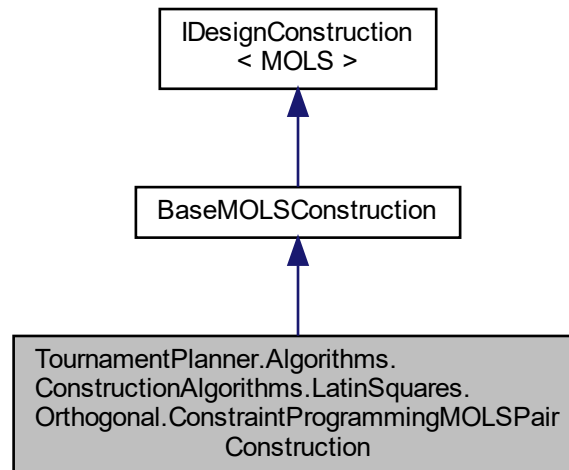
The documentation for this class was generated from the following file:

- CommutativeIdempotentLatinSquareConstruction.cs

7.16 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.ConstraintProgrammingMOLSPairConstruction Class Reference

Constraint a pair of MOLS using constraint programming approach with CP-SAT solver. This method only works for MOLS of small orders.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.ConstraintProgrammingMOLSPairConstruction:



Public Member Functions

- [ConstraintProgrammingMOLSPairConstruction](#) (int order)
- override [MOLS ConstructDesign](#) ()
Construct MOLS.

Protected Member Functions

- override bool [ValidateOrder](#) (int order)
Method to validate the order in the constructor.

Properties

- override string [ArgumentExceptionMessage](#) [get]

7.16.1 Detailed Description

Constraint a pair of MOLS using constraint programming approach with CP-SAT solver. This method only works for MOLS of small orders.

The documentation for this class was generated from the following file:

- `ConstraintProgrammingMOLSPairConstruction.cs`

7.17 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.ConstraintProgrammingMOLSPairConstructionTests Class Reference

Tests for the ConstraintProgrammingMOLSPairConstruction.

Public Member Functions

- void [InvalidOrderShouldThrowException](#) ()
Test that an exception is thrown when the order is 2 or 6.
- void [OrthogonalityTest](#) ()
Test the orthogonality of a few pairs of MOLS of low order.

7.17.1 Detailed Description

Tests for the ConstraintProgrammingMOLSPairConstruction.

7.17.2 Member Function Documentation

7.17.2.1 InvalidOrderShouldThrowException()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.ConstraintProgrammingMOLSPairConstructionTests.InvalidOrderShouldThrowException ( )
```

Test that an exception is thrown when the order is 2 or 6.

7.17.2.2 OrthogonalityTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.ConstraintProgrammingMOLSPairConstructionTests.OrthogonalityTest ( )
```

Test the orthogonality of a few pairs of MOLS of low order.

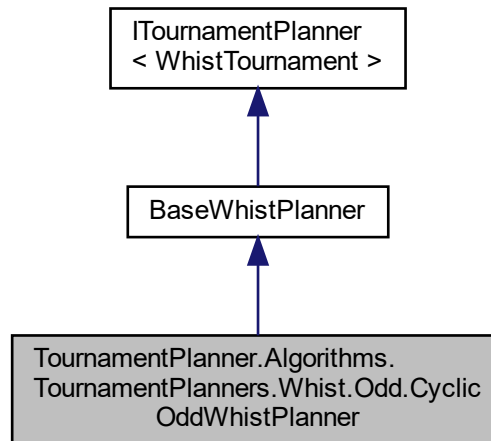
The documentation for this class was generated from the following file:

- ConstraintProgrammingMOLSPairConstructionTests.cs

7.18 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlanner Class Reference

Cyclic odd whist planner for $v = 33, 57$.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlanner:



Public Member Functions

- [CyclicOddWhistPlanner](#) (int v)
Initializes a new instance of the [CyclicOddWhistPlanner](#) class.
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.18.1 Detailed Description

Cyclic odd whist planner for $v = 33, 57$.

7.18.2 Constructor & Destructor Documentation

7.18.2.1 CyclicOddWhistPlanner()

```

TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlanner.CyclicOddWhistPlanner (
    int v )
  
```

Initializes a new instance of the [CyclicOddWhistPlanner](#) class.

Parameters

v	The order.
---	------------

The documentation for this class was generated from the following file:

- CyclicOddWhistPlanner.cs

7.19 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlannerTests Class Reference

Tests for the CyclicOddWhistPlanner.

Public Member Functions

- void [TestWhist33Construction](#) ()
Tests the construction of Whist(33).
- void [TestWhist57Construction](#) ()
Test the construction of Whist 57.

7.19.1 Detailed Description

Tests for the CyclicOddWhistPlanner.

7.19.2 Member Function Documentation

7.19.2.1 TestWhist33Construction()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlanner↔  
Tests.TestWhist33Construction ( )
```

Tests the construction of Whist(33).

7.19.2.2 TestWhist57Construction()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.CyclicOddWhistPlanner↔  
Tests.TestWhist57Construction ( )
```

Test the construction of Whist 57.

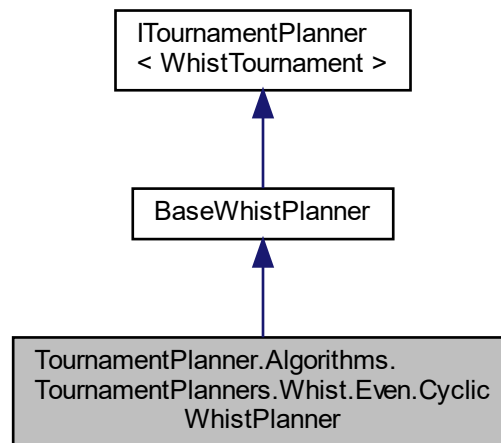
The documentation for this class was generated from the following file:

- CyclicOddWhistPlannerTests.cs

7.20 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlanner Class Reference

Cyclic construction method for several orders of [Whist](#)($V = 4N$).

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlanner:



Public Member Functions

- [CyclicWhistPlanner](#) (int v)
Initializes a new instance of the [CyclicWhistPlanner](#) class.
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.20.1 Detailed Description

Cyclic construction method for several orders of [Whist](#)($V = 4N$).

7.20.2 Constructor & Destructor Documentation

7.20.2.1 CyclicWhistPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlanner.CyclicWhistPlanner (
    int v )
```

Initializes a new instance of the [CyclicWhistPlanner](#) class.

Parameters

✓	The order of the whist tournament.
---	------------------------------------

The documentation for this class was generated from the following file:

- CyclicWhistPlanner.cs

7.21 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlannerTests Class Reference

Tests for the CyclicWhistPlanner.

Public Member Functions

- void [CyclicWhistConstructionsTest](#) ()
Tests the construction of the supported cyclic whist tournaments..

7.21.1 Detailed Description

Tests for the CyclicWhistPlanner.

7.21.2 Member Function Documentation

7.21.2.1 CyclicWhistConstructionsTest()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.CyclicWhistPlannerTests.CyclicWhistConstructionsTest ( )
```

Tests the construction of the supported cyclic whist tournaments..

The documentation for this class was generated from the following file:

- CyclicWhistPlannerTests.cs

7.22 TournamentPlannerTests.Datastore.DataReaderTests Class Reference

Tests for the DataReader.

Public Member Functions

- void **TestExistingNbMOLS** ()

7.22.1 Detailed Description

Tests for the DataReader.

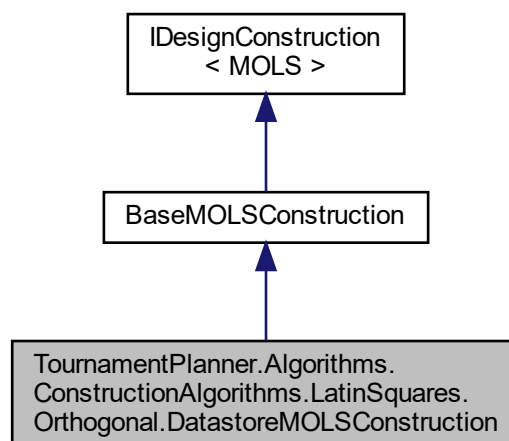
The documentation for this class was generated from the following file:

- DataReaderTests.cs

7.23 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DatastoreMOLSConstruction Class Reference

This class creates MOLS objects based on the ones in the data store. This is not a construction algorithm per se, but acts like one. It implements the same interface as the other MOLS construction methods so it is easier to use in other algorithms.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DatastoreMOLSConstruction:



Public Member Functions

- [DatastoreMOLSConstruction](#) (int order, int nbMols=2)
- override [MOLS ConstructDesign](#) ()

Construct MOLS.

Protected Member Functions

- override bool [ValidateOrder](#) (int order)
Method to validate the order in the constructor.

Properties

- override string [ArgumentExceptionMessage](#) [get]

7.23.1 Detailed Description

This class creates MOLS objects based on the ones in the data store. This is not a construction algorithm per se, but acts like one. It implements the same interface as the other MOLS construction methods so it is easier to use in other algorithms.

The documentation for this class was generated from the following file:

- DatastoreMOLSCONSTRUCTION.cs

7.24 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ LatinSquares.Orthogonal.DatastoreMOLSCONSTRUCTIONTESTS Class Reference

Tests for the DatastoreMOLSCONSTRUCTION.

Public Member Functions

- void [SeveralOrdersTest](#) ()
Test the retrieval of MOLS of several orders and their validity.

7.24.1 Detailed Description

Tests for the DatastoreMOLSCONSTRUCTION.

7.24.2 Member Function Documentation

7.24.2.1 SeveralOrdersTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.Datastore↔  
MOLSCONSTRUCTIONTESTS.SeveralOrdersTest ( )
```

Test the retrieval of MOLS of several orders and their validity.

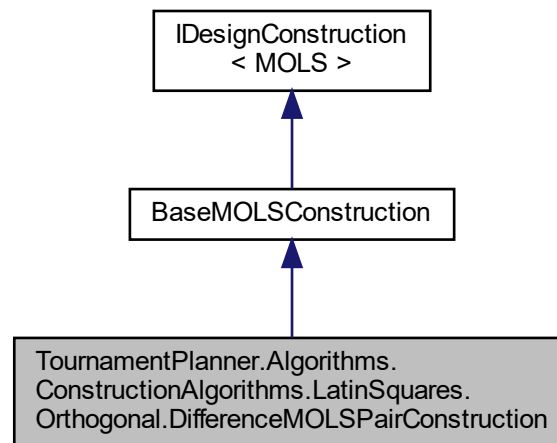
The documentation for this class was generated from the following file:

- DatastoreMOLSCONSTRUCTIONTESTS.cs

7.25 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DifferenceMOLSPairConstruction Class Reference

Construct a pair of MOLS of order $3M + 1$. This way, MOLS for all orders $12t + 10$ (which is a subset of $4k + 2$) can be constructed. This method was introduced in [https://www.math.uci.edu/~brusso/\[14\]BosShrParCJM1960.pdf](https://www.math.uci.edu/~brusso/[14]BosShrParCJM1960.pdf).

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DifferenceMOLSPairConstruction:



Public Member Functions

- [DifferenceMOLSPairConstruction](#) (int order)
- override [MOLS ConstructDesign](#) ()
Construct MOLS.

Protected Member Functions

- override bool [ValidateOrder](#) (int order)
Method to validate the order in the constructor.

Properties

- override string [ArgumentExceptionMessage](#) [get]

7.25.1 Detailed Description

Construct a pair of MOLS of order $3M + 1$. This way, MOLS for all orders $12t + 10$ (which is a subset of $4k + 2$) can be constructed. This method was introduced in [https://www.math.uci.edu/~brusso/\[14\]BosShrParCJM1960.pdf](https://www.math.uci.edu/~brusso/[14]BosShrParCJM1960.pdf).

The documentation for this class was generated from the following file:

- DifferenceMOLSPairConstruction.cs

7.26 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DifferenceMOLSPairConstructionTests Class Reference

Tests for the DifferenceMOLSPairConstruction.

Public Member Functions

- void [InvalidOrderTest](#) ()
Test that an exception is thrown when order is not $3m + 1$.
- void [OrthogonalityTest](#) ()
Test the orthogonality of a few pairs of MOLS of order $3M + 1$.

7.26.1 Detailed Description

Tests for the DifferenceMOLSPairConstruction.

7.26.2 Member Function Documentation

7.26.2.1 InvalidOrderTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DifferenceMOLSPairConstructionTests.InvalidOrderTest ( )
```

Test that an exception is thrown when order is not $3m + 1$.

7.26.2.2 OrthogonalityTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DifferenceMOLSPairConstructionTests.OrthogonalityTest ( )
```

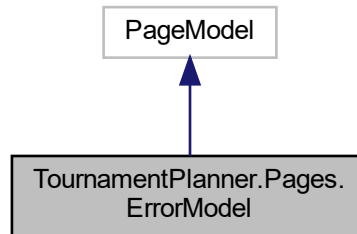
Test the orthogonality of a few pairs of MOLS of order $3M + 1$.

The documentation for this class was generated from the following file:

- DifferenceMOLSPairConstructionTests.cs

7.27 TournamentPlanner.Pages.ErrorModel Class Reference

Inheritance diagram for TournamentPlanner.Pages.ErrorModel:



Public Member Functions

- **ErrorModel** (ILogger< [ErrorModel](#) > logger)
- void **OnGet** ()

Properties

- string **RequestId** [get, set]
- bool **ShowRequestId** [get]

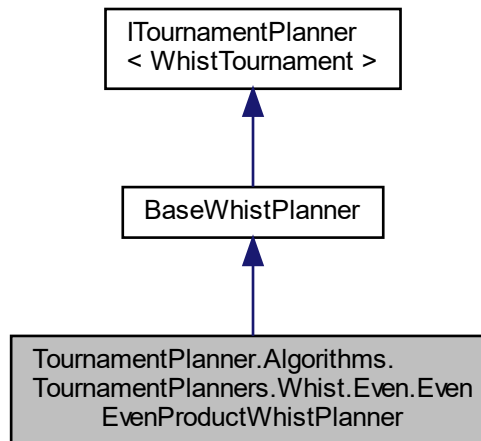
The documentation for this class was generated from the following file:

- Error.cshtml.cs

7.28 TournamentPlanner.Algorithms.TournamentPlanners.Whist.EvenEvenEvenProductWhistPlanner Class Reference

Construction of a [Whist](#) tournament as a product of 2 existing Wh(4n) tournaments.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlanner:



Public Member Functions

- `EvenEvenProductWhistPlanner` (int v, `WhistTournament` whistV, `WhistTournament` whistW)
Initializes a new instance of the `EvenEvenProductWhistPlanner` class.
- override `WhistTournament PlanTournament` ()
Plan the `Whist` tournament.

Additional Inherited Members

7.28.1 Detailed Description

Construction of a `Whist` tournament as a product of 2 existing `Wh(4n)` tournaments.

7.28.2 Constructor & Destructor Documentation

7.28.2.1 `EvenEvenProductWhistPlanner()`

```
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlanner.EvenEvenProductWhistPlanner (
    int v,
    WhistTournament whistV,
    WhistTournament whistW )
```

Initializes a new instance of the `EvenEvenProductWhistPlanner` class.

Parameters

v	Order of new whist.
$whistV$	First $Wh(v)$.
$whistW$	Second $Wh(w)$.

The documentation for this class was generated from the following file:

- `EvenEvenProductWhistPlanner.cs`

7.29 [TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlannerTests](#) Class Reference

Tests for the `ProductWhistPlanner`.

Public Member Functions

- `void TwoEvenProductTest ()`
- `void Whist32Test ()`
- `void Whit64Test ()`
- `void Whit80Test ()`
- `void Whit96Test ()`
- `void Whit144Test ()`

7.29.1 Detailed Description

Tests for the `ProductWhistPlanner`.

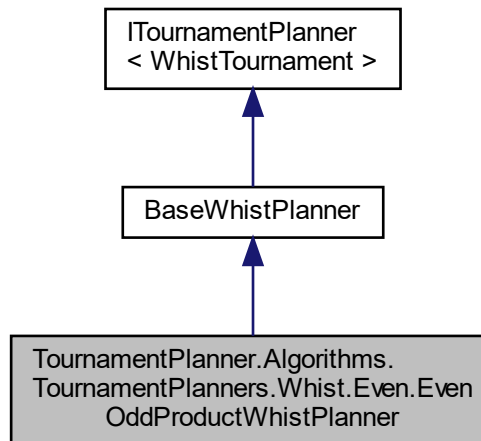
The documentation for this class was generated from the following file:

- `EvenEvenProductWhistPlannerTests.cs`

7.30 [TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenOddProductWhistPlanner](#) Class Reference

Construction of a [Whist](#) tournament as a product of a $Wh(4n)$ and $Wh(4n + 1)$.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenOddProductWhistPlanner:



Public Member Functions

- `EvenOddProductWhistPlanner` (int v, `WhistTournament` whistV, `WhistTournament` whistW)
Initializes a new instance of the `EvenOddProductWhistPlanner` class.
- override `WhistTournament PlanTournament` ()
Plan the `Whist` tournament.

Additional Inherited Members

7.30.1 Detailed Description

Construction of a `Whist` tournament as a product of a `Wh(4n)` and `Wh(4n + 1)`.

7.30.2 Constructor & Destructor Documentation

7.30.2.1 `EvenOddProductWhistPlanner()`

```
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenOddProductWhistPlanner.EvenOddProductWhistPlanner (
    int v,
    WhistTournament whistV,
    WhistTournament whistW )
```

Initializes a new instance of the `EvenOddProductWhistPlanner` class.

Parameters

v	Order of new whist.
$whistV$	First Wh(4n).
$whistW$	Second Wh(4n+1).

The documentation for this class was generated from the following file:

- EvenOddProductWhistPlanner.cs

7.31 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist. Even.EvenOddProductWhistPlannerTests Class Reference

Tests for the EvenOddProductWhistPlanner.

Public Member Functions

- void **Whist36ConstructionTest** ()
- void **Whist60ConstructionTest** ()

7.31.1 Detailed Description

Tests for the EvenOddProductWhistPlanner.

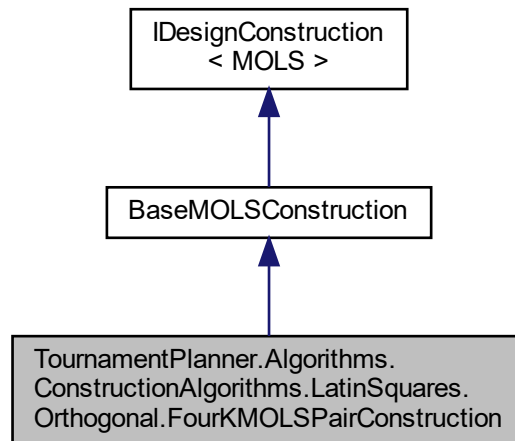
The documentation for this class was generated from the following file:

- EvenOddProductWhistPlannerTests.cs

7.32 TournamentPlanner.Algorithms.ConstructionAlgorithms.Latin Squares.Orthogonal.FourKMOLSPairConstruction Class Reference

Construction of MOLS of order $V = 4 * K$.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPairConstruction:



Public Member Functions

- [FourKMOLSPairConstruction](#) (int order)
- override [MOLS ConstructDesign](#) ()
Construct MOLS.

Protected Member Functions

- override bool [ValidateOrder](#) (int order)
Method to validate the order in the constructor.

Properties

- override string [ArgumentExceptionMessage](#) [get]

7.32.1 Detailed Description

Construction of MOLS of order $V = 4 * K$.

The documentation for this class was generated from the following file:

- FourKMOLSPairConstruction.cs

7.33 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPairConstructionTests Class Reference

Tests for the FourKMOLSPairConstruction.

Public Member Functions

- void [InvalidOrderTest](#) ()
Test that an exception is thrown when the order is not a multiple of 4.
- void [OrthogonalityTest](#) ()
Test the orthogonality of a few pairs of MOLS of order 4 K.

7.33.1 Detailed Description

Tests for the FourKMOLSPairConstruction.

7.33.2 Member Function Documentation

7.33.2.1 InvalidOrderTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPairConstructionTests.InvalidOrderTest ( )
```

Test that an exception is thrown when the order is not a multiple of 4.

7.33.2.2 OrthogonalityTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPairConstructionTests.OrthogonalityTest ( )
```

Test the orthogonality of a few pairs of MOLS of order 4 K.

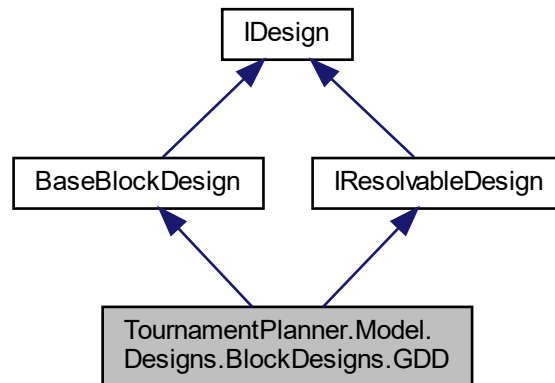
The documentation for this class was generated from the following file:

- FourKMOLSPairConstructionTests.cs

7.34 TournamentPlanner.Model.Designs.BlockDesigns.GDD Class Reference

Class representing a group divisible design.

Inheritance diagram for TournamentPlanner.Model.Designs.BlockDesigns.GDD:



Public Member Functions

- [GDD](#) (int v)
Initializes a new instance of the [GDD](#) class.
- [GDD](#) (int v, List< List< int >> groups)
Initializes a new instance of the [GDD](#) class.
- [GDD](#) (int v, int k, List< List< int >> groups)
Initializes a new instance of the [GDD](#) class.
- [GDD](#) (int v, int k, List< List< int >> groups, List< List< int >> blocks)
Initializes a new instance of the [GDD](#) class.
- [GDD](#) (int v, List< int > blockSizes, List< List< int >> groups, List< List< int >> blocks)
Initializes a new instance of the [GDD](#) class.
- void [AddParallelClassAndBlocks](#) (List< List< int >> parallelClass)
Add a parallel class and its blocks at the same time.
- void [AddBlock](#) (List< int > block)
Ass a single block to the blocks of the design.
- List< int > [GetGroupSizes](#) ()
Get the size(s) of the groups in the [GDD](#).

Properties

- `int K = 3` [get]
Gets the block size.
- `List< int > BlockSizes = new List<int> { 3 }` [get]
Gets the allowed block sizes.
- `List< List< int > > Groups = new ()` [get]
Gets the groups of the GDD.
- `List< List< List< int > > > ParallelClasses = new ()` [get]
Gets the parallel classes of the GDD.

Additional Inherited Members

7.34.1 Detailed Description

Class representing a group divisible design.

7.34.2 Constructor & Destructor Documentation

7.34.2.1 GDD() [1/5]

```
TournamentPlanner.Model.Designs.BlockDesigns.GDD.GDD (
    int v )
```

Initializes a new instance of the [GDD](#) class.

Parameters

<code>v</code>	Size of the GDD .
----------------	-----------------------------------

7.34.2.2 GDD() [2/5]

```
TournamentPlanner.Model.Designs.BlockDesigns.GDD.GDD (
    int v,
    List< List< int >> groups )
```

Initializes a new instance of the [GDD](#) class.

Parameters

<code>v</code>	Size of the GDD .
<code>groups</code>	The groups of the GDD .

7.34.2.3 GDD() [3/5]

```
TournamentPlanner.Model.Designs.BlockDesigns.GDD.GDD (
    int v,
    int k,
    List< List< int >> groups )
```

Initializes a new instance of the [GDD](#) class.

Parameters

<i>v</i>	Size of the GDD .
<i>k</i>	Size of the blocks.
<i>groups</i>	The groups of the GDD .

7.34.2.4 GDD() [4/5]

```
TournamentPlanner.Model.Designs.BlockDesigns.GDD.GDD (
    int v,
    int k,
    List< List< int >> groups,
    List< List< int >> blocks )
```

Initializes a new instance of the [GDD](#) class.

Parameters

<i>v</i>	Size of the GDD .
<i>k</i>	Size of the blocks.
<i>groups</i>	The groups of the GDD .
<i>blocks</i>	The blocks of the GDD .

7.34.2.5 GDD() [5/5]

```
TournamentPlanner.Model.Designs.BlockDesigns.GDD.GDD (
    int v,
    List< int > blockSizes,
    List< List< int >> groups,
    List< List< int >> blocks )
```

Initializes a new instance of the [GDD](#) class.

Parameters

<i>v</i>	Size of the GDD .
<i>blockSizes</i>	Sizes of the blocks.
<i>groups</i>	The groups of the GDD .
<i>blocks</i>	The blocks of the GDD .

7.34.3 Member Function Documentation

7.34.3.1 AddBlock()

```
void TournamentPlanner.Model.Designs.BlockDesigns.GDD.AddBlock (
    List< int > block )
```

Ass a single block to the blocks of the design.

Parameters

<i>block</i>	The block to add.
--------------	-------------------

7.34.3.2 AddParallelClassAndBlocks()

```
void TournamentPlanner.Model.Designs.BlockDesigns.GDD.AddParallelClassAndBlocks (
    List< List< int >> parallelClass )
```

Add a parallel class and its blocks at the same time.

Parameters

<i>parallelClass</i>	The parallel class to add.
----------------------	----------------------------

7.34.3.3 GetGroupSizes()

```
List<int> TournamentPlanner.Model.Designs.BlockDesigns.GDD.GetGroupSizes ( )
```

Get the size(s) of the groups in the [GDD](#).

Returns

List of sizes.

7.34.4 Property Documentation

7.34.4.1 BlockSizes

```
List<int> TournamentPlanner.Model.Designs.BlockDesigns.GDD.BlockSizes = new List<int> { 3 }  
[get]
```

Gets the allowed block sizes.

7.34.4.2 Groups

```
List<List<int> > TournamentPlanner.Model.Designs.BlockDesigns.GDD.Groups = new () [get]
```

Gets the groups of the [GDD](#).

7.34.4.3 K

```
int TournamentPlanner.Model.Designs.BlockDesigns.GDD.K = 3 [get]
```

Gets the block size.

7.34.4.4 ParallelClasses

```
List<List<List<int> > > TournamentPlanner.Model.Designs.BlockDesigns.GDD.ParallelClasses =  
new () [get]
```

Gets the parallel classes of the [GDD](#).

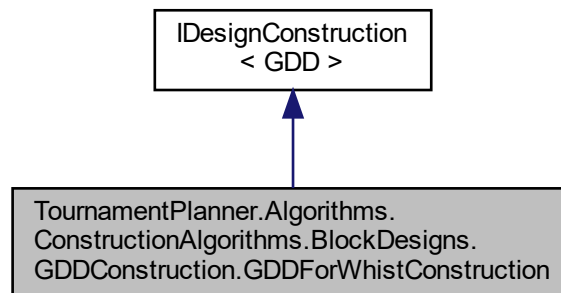
The documentation for this class was generated from the following file:

- GDD.cs

7.35 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.GDDForWhistConstruction Class Reference

Construction of GDD($17g, + 4u + 4v, \{5, 17\}, \{g, g + 4u, g + 4v\}$). These are used to construct whist tournaments.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.GDDForWhistConstruction:



Public Member Functions

- [GDDForWhistConstruction](#) (int g, int u, int v)
Initializes a new instance of the [GDDForWhistConstruction](#) class.
- [GDD ConstructDesign](#) ()

7.35.1 Detailed Description

Construction of GDD($17g, + 4u + 4v, \{5, 17\}, \{g, g + 4u, g + 4v\}$). These are used to construct whist tournaments.

7.35.2 Constructor & Destructor Documentation

7.35.2.1 GDDForWhistConstruction()

```

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.GDDForWhistConstruction.GDDForWhistConstruction (
    int g,
    int u,
    int v )
  
```

Initializes a new instance of the [GDDForWhistConstruction](#) class.

Parameters

g	Initial size of groups.
u	Number of elements to replace in the first group.
v	Number of elements to replace in the second group.

The documentation for this class was generated from the following file:

- GDDForWhistConstruction.cs

7.36 TournamentPlannerTests.Algorithms.ConstructionAlgorithms. BlockDesigns.GDDConstruction.GDDForWhistConstructionTests Class Reference

Tests for the GDDForWhistConstruction.

Public Member Functions

- void **GDDConstructionTest** ()
- void **GDDConstructionWithReplacementsIn2GroupsTest** ()

7.36.1 Detailed Description

Tests for the GDDForWhistConstruction.

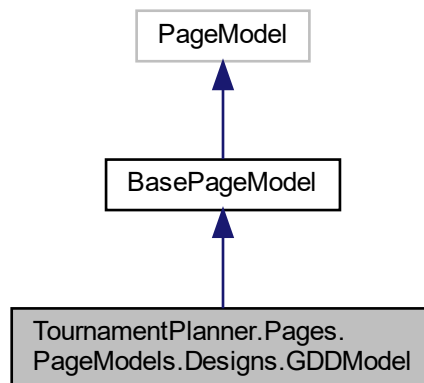
The documentation for this class was generated from the following file:

- GDDForWhistConstructionTests.cs

7.37 TournamentPlanner.Pages.PageModels.Designs.GDDModel Class Reference

Page model for resolvable group divisible designs with block size 3.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Designs.GDDModel:



Public Member Functions

- void [OnPost](#) ()

Properties

- int [M](#) = 4 [get, set]
Gets or sets the M which is the number of groups and the order of MOLS used for the construction.
- int [V](#) = 12 [get, set]
*Gets or sets the size of the GDD. $V = 3 * M$.*
- [GDD](#) [GDD](#) [get]
Gets the constructed GDD.

7.37.1 Detailed Description

Page model for resolvable group divisible designs with block size 3.

7.37.2 Property Documentation

7.37.2.1 GDD

[GDD](#) `TournamentPlanner.Pages.PageModels.Designs.GDDModel.GDD` [get]

Gets the constructed GDD.

7.37.2.2 M

```
int TournamentPlanner.Pages.PageModels.Designs.GDDModel.M = 4 [get], [set]
```

Gets or sets the M which is the number of groups and the order of MOLS used for the construction.

7.37.2.3 V

```
int TournamentPlanner.Pages.PageModels.Designs.GDDModel.V = 12 [get], [set]
```

Gets or sets the size of the GDD. $V = 3 * M$.

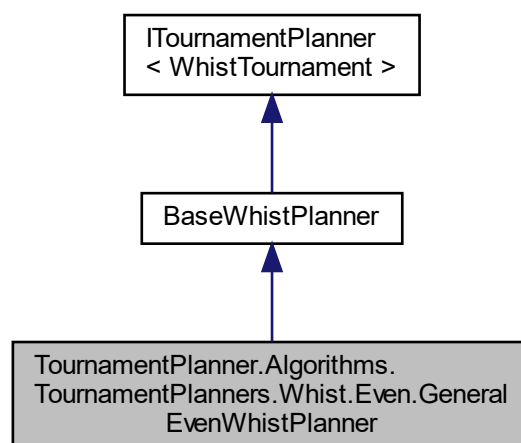
The documentation for this class was generated from the following file:

- GDD.cshtml.cs

7.38 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner Class Reference

Plan a whist tournament of order $4n$ or $4n + 1$ using some of the implemented approaches. This class basically just selects the method to use for the construction.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner:



Public Member Functions

- [GeneralEvenWhistPlanner](#) (int v)
Initializes a new instance of the [GeneralEvenWhistPlanner](#) class.
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.38.1 Detailed Description

Plan a whist tournament of order $4n$ or $4n + 1$ using some of the implemented approaches. This class basically just selects the method to use for the construction.

7.38.2 Constructor & Destructor Documentation

7.38.2.1 GeneralEvenWhistPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner.GeneralEvenWhistPlanner (
    int v )
```

Initializes a new instance of the [GeneralEvenWhistPlanner](#) class.

Parameters

v	The order of the tournament.
---	------------------------------

The documentation for this class was generated from the following file:

- [GeneralEvenWhistPlanner.cs](#)

7.39 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlannerTests Class Reference

Tests for the [General4NWhistPlanner](#).

Public Member Functions

- void [SmallerWhistConstructionTest](#) ()
Validate constructed whist tournaments up to 81.
- void [LargerWhistConstructionTest](#) ()
Test the construction of larger whist tournaments. Commented because it takes a long time but passes.
- void [LargerConstructionWithoutValidationTest](#) ()
Construction without validation. Takes a long ass time too.

7.39.1 Detailed Description

Tests for the General4NWhistPlanner.

7.39.2 Member Function Documentation

7.39.2.1 LargerConstructionWithoutValidationTest()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner←  
Tests.LargerConstructionWithoutValidationTest ( )
```

Construction without validation. Takes a long ass time too.

[Test]

7.39.2.2 LargerWhistConstructionTest()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner←  
Tests.LargerWhistConstructionTest ( )
```

Test the construction of larger whist tournaments. Commented because it takes a long time but passes.

[Test]

7.39.2.3 SmallerWhistConstructionTest()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner←  
Tests.SmallerWhistConstructionTest ( )
```

Validate constructed whist tournaments up to 81.

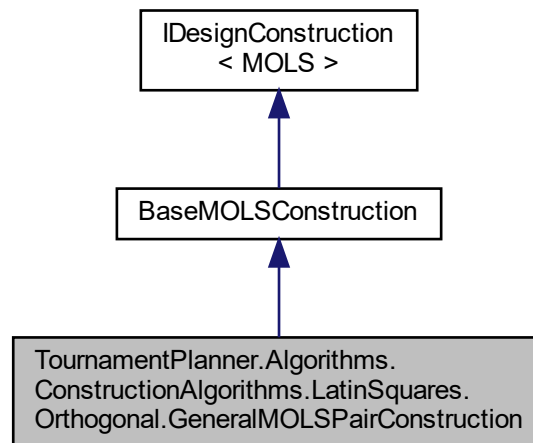
The documentation for this class was generated from the following file:

- GeneralEvenWhistPlannerTests.cs

7.40 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.GeneralMOLSPairConstruction Class Reference

Class facilitating the construction of MOLS of all the supported orders. Uses other construction algorithms.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.GeneralMOLSPairConstruction:



Public Member Functions

- [GeneralMOLSPairConstruction](#) (int order)
- override [MOLS ConstructDesign](#) ()
Construct MOLS.

Protected Member Functions

- override bool [ValidateOrder](#) (int order)
Method to validate the order in the constructor.

Properties

- override string [ArgumentExceptionMessage](#) [get]

7.40.1 Detailed Description

Class facilitating the construction of MOLS of all the supported orders. Uses other construction algorithms.

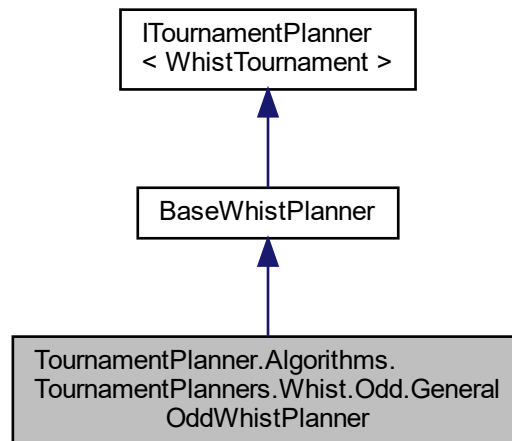
The documentation for this class was generated from the following file:

- `GeneralMOLSPairConstruction.cs`

7.41 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.GeneralOddWhistPlanner Class Reference

Construct some [Whist](#) tournaments of order $4n + 1$. Not all orders are supported unfortunately.

Inheritance diagram for [TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.GeneralOddWhistPlanner](#):



Public Member Functions

- [GeneralOddWhistPlanner](#) (int v)
Initializes a new instance of the [GeneralOddWhistPlanner](#) class.
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.41.1 Detailed Description

Construct some [Whist](#) tournaments of order $4n + 1$. Not all orders are supported unfortunately.

7.41.2 Constructor & Destructor Documentation

7.41.2.1 GeneralOddWhistPlanner()

```

TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.GeneralOddWhistPlanner.GeneralOddWhistPlanner (
    int v )
  
```

Initializes a new instance of the [GeneralOddWhistPlanner](#) class.

Parameters

v	Number of players.
---	--------------------

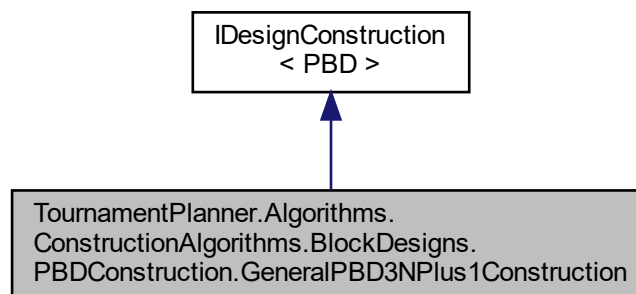
The documentation for this class was generated from the following file:

- GeneralOddWhistPlanner.cs

7.42 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block↔ Designs.PBDConstruction.GeneralPBD3NPlus1Construction Class Reference

Class that is used for construction of any PBD(3n+1). It uses the required depending on the order. It is also used in the recursive step of the [LargerOrderPBDConstruction](#).

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.↔
GeneralPBD3NPlus1Construction:



Public Member Functions

- [GeneralPBD3NPlus1Construction](#) (int v)
Initializes a new instance of the [GeneralPBD3NPlus1Construction](#) class.
- [PBD ConstructDesign](#) ()

7.42.1 Detailed Description

Class that is used for construction of any PBD(3n+1). It uses the required depending on the order. It is also used in the recursive step of the [LargerOrderPBDConstruction](#).

7.42.2 Constructor & Destructor Documentation

7.42.2.1 GeneralPBD3NPlus1Construction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.GeneralPBD3↔  
NPlus1Construction.GeneralPBD3NPlus1Construction (   
    int v )
```

Initializes a new instance of the [GeneralPBD3NPlus1Construction](#) class.

Parameters

v	The PBD order.
---	----------------

The documentation for this class was generated from the following file:

- [GeneralPBD3NPlus1Construction.cs](#)

7.43 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ BlockDesigns.BIBDConstruction.GurobiIncremental↔ BIBDConstructionTests Class Reference

Tests for the GurobiIncrementalBIBDConstruction.

Public Member Functions

- void [IncrementalBIBDConstructionTest](#) ()
Test the algorithm on a few BIBDs.

7.43.1 Detailed Description

Tests for the GurobiIncrementalBIBDConstruction.

7.43.2 Member Function Documentation

7.43.2.1 IncrementalBIBDConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔
GurobiIncrementalBIBDConstructionTests.IncrementalBIBDConstructionTest ( )
```

Test the algorithm on a few BIBDs.

The documentation for this class was generated from the following file:

- GurobiIncrementalBIBDConstructionTests.cs

7.44 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ BlockDesigns.BIBDConstruction.GurobiQuadraticProgramming↔ BIBDConstructionTests Class Reference

Tests for the QuadraticProgrammingBIBDConstruction.

Public Member Functions

- void **QuadraticConstructionTest** ()

7.44.1 Detailed Description

Tests for the QuadraticProgrammingBIBDConstruction.

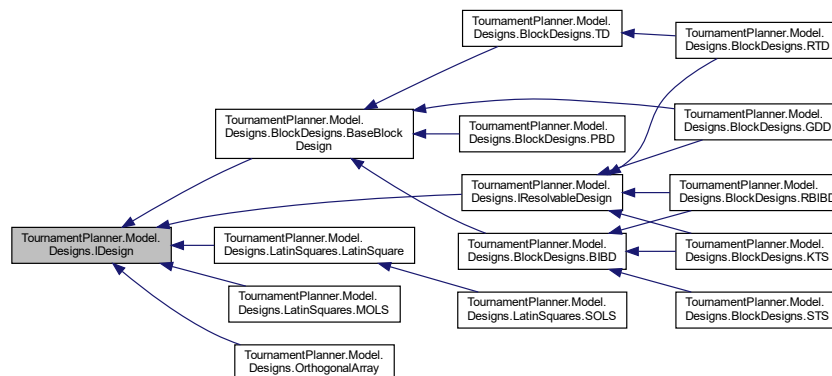
The documentation for this class was generated from the following file:

- GurobiQuadraticProgrammingBIBDConstructionTests.cs

7.45 TournamentPlanner.Model.Designs.IDesign Interface Reference

Common interface for all the design structures.

Inheritance diagram for TournamentPlanner.Model.Designs.IDesign:



Properties

- int `V` [get]

Gets the size of the design. In literature, V denotes the size of the design - the number of distinct elements in the design. It is probably the only common parameter of all the designs. From the tournament POV, it denotes the number of players. In Latin squares, V is the order of the square. In the context of orthogonal arrays, V denotes the level of the OA.

7.45.1 Detailed Description

Common interface for all the design structures.

7.45.2 Property Documentation

7.45.2.1 `V`

```
int TournamentPlanner.Model.Designs.IDesign.V [get]
```

Gets the size of the design. In literature, V denotes the size of the design - the number of distinct elements in the design. It is probably the only common parameter of all the designs. From the tournament POV, it denotes the number of players. In Latin squares, V is the order of the square. In the context of orthogonal arrays, V denotes the level of the OA.

The documentation for this interface was generated from the following file:

- IDesign.cs

7.46 TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< IDesign > Interface Template Reference

Interface for the design construction algorithms.

Public Member Functions

- IDesign `ConstructDesign` ()

Construct the design.

7.46.1 Detailed Description

Interface for the design construction algorithms.

Template Parameters

<i>IDesign</i>	The type of IDesign.
----------------	----------------------

7.46.2 Member Function Documentation

7.46.2.1 ConstructDesign()

```
IDesign TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< IDesign >.ConstructDesign ( )
```

Construct the design.

Returns

Concrete design.

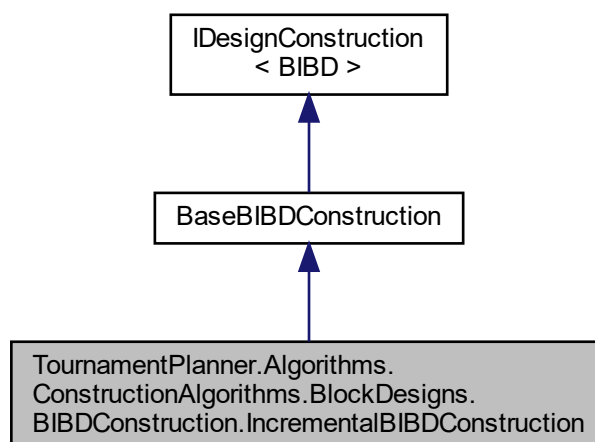
The documentation for this interface was generated from the following file:

- IDesignConstruction.cs

7.47 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.IncrementalBIBDConstruction Class Reference

Incremental method that constructs BIBDs block by block. Implementation of the algorithm described in https://www.researchgate.net/publication/220660995_A_mathematical_programming_approach_to_the_construction_of_BIBDs. Uses CP-SAT as solver.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.IncrementalBIBDConstruction:



Public Member Functions

- [IncrementalBIBDConstruction](#) (int v, int k, int lambda, int timeLimit=10)
- override [BIBD ConstructDesign](#) ()
Construct the BIBD.

Additional Inherited Members

7.47.1 Detailed Description

Incremental method that constructs BIBDs block by block. Implementation of the algorithm described in https://www.researchgate.net/publication/220660995_A_mathematical_programming_approach_to_the_construction_of_BIBDs. Uses CP-SAT as solver.

The documentation for this class was generated from the following file:

- IncrementalBIBDConstruction.cs

7.48 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.IncrementalBIBDConstructionTests Class Reference

Test for the IncrementalBIBDConstruction.

Public Member Functions

- void [IncrementalBIBDConstructionTest](#) ()
Test the algorithm on a few BIBDs.

7.48.1 Detailed Description

Test for the IncrementalBIBDConstruction.

7.48.2 Member Function Documentation

7.48.2.1 IncrementalBIBDConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.IncrementalBIBDConstructionTests.IncrementalBIBDConstructionTest ( )
```

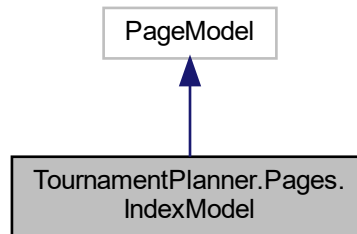
Test the algorithm on a few BIBDs.

The documentation for this class was generated from the following file:

- IncrementalBIBDConstructionTests.cs

7.49 TournamentPlanner.Pages.IndexModel Class Reference

Inheritance diagram for TournamentPlanner.Pages.IndexModel:



Public Member Functions

- **IndexModel** (ILogger< [IndexModel](#) > logger)
- void **OnGet** ()

Properties

- string **Message** = "PageModel in C#" [get]

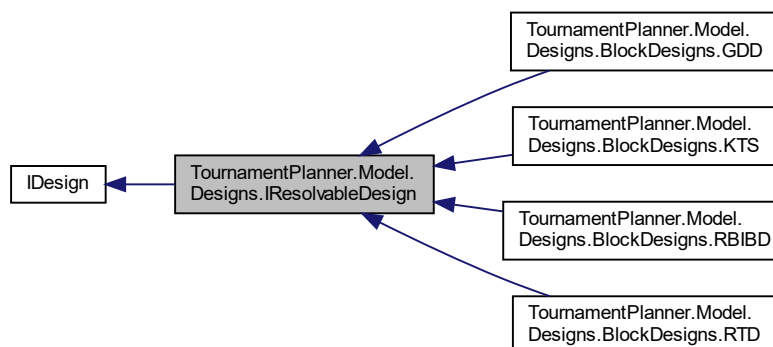
The documentation for this class was generated from the following file:

- Index.cshhtml.cs

7.50 TournamentPlanner.Model.Designs.IResolvableDesign Interface Reference

Interface for resolvable designs.

Inheritance diagram for TournamentPlanner.Model.Designs.IResolvableDesign:



Properties

- [List< List< List< int > > > ParallelClasses](#) [get]
Gets the parallel classes of the resolvable design.

7.50.1 Detailed Description

Interface for resolvable designs.

7.50.2 Property Documentation

7.50.2.1 ParallelClasses

```
List<List<List<int> > > TournamentPlanner.Model.Designs.IResolvableDesign.ParallelClasses  
[get]
```

Gets the parallel classes of the resolvable design.

The documentation for this interface was generated from the following file:

- IResolvableDesign.cs

7.51 TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< BaseTournament > Interface Template Reference

Interface for the tournament planners.

Public Member Functions

- [BaseTournament PlanTournament](#) ()
Plan the given tournament.

7.51.1 Detailed Description

Interface for the tournament planners.

Template Parameters

<i>BaseTournament</i>	Type of the tournament to plan.
-----------------------	---------------------------------

7.51.2 Member Function Documentation

7.51.2.1 PlanTournament()

```
BaseTournament TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner< BaseTournament
>.PlanTournament ( )
```

Plan the given tournament.

Returns

The planned tournament.

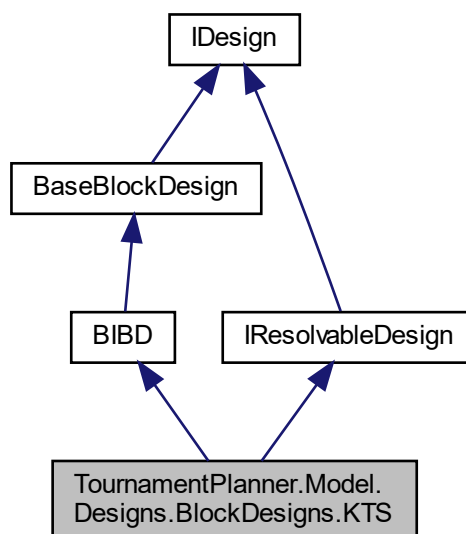
The documentation for this interface was generated from the following file:

- ITournamentPlanner.cs

7.52 TournamentPlanner.Model.Designs.BlockDesigns.KTS Class Reference

A class representing a Kirkman triple system. A [KTS](#) is a resolvable BIBD($V, 3, 1$), where $V = 6N + 3$. Basically, a resolvable [STS](#) is a [KTS](#).

Inheritance diagram for TournamentPlanner.Model.Designs.BlockDesigns.KTS:



Public Member Functions

- [KTS](#) (int v)
Initializes a new instance of the [KTS](#) class.
- [KTS](#) (int v, List< List< List< int >>> parallelClasses)
Initializes a new instance of the [KTS](#) class.
- void [AddParallelClass](#) (List< List< int >> parallelClass)
Add a parallel class to the list of parallel classes.
- void [AddParallelClassAndBlocks](#) (List< List< int >> parallelClass)
Add a parallel class to the list of parallel classes. At the same time, it adds the blocks of the parallel class to the list of blocks.

Properties

- List< List< List< int >>> [ParallelClasses](#) = new List<List<List<int>>>() [get]

Additional Inherited Members

7.52.1 Detailed Description

A class representing a Kirkman triple system. A [KTS](#) is a resolvable BIBD($V, 3, 1$), where $V = 6N + 3$. Basically, a resolvable [STS](#) is a [KTS](#).

7.52.2 Constructor & Destructor Documentation

7.52.2.1 [KTS\(\)](#) [1/2]

```
TournamentPlanner.Model.Designs.BlockDesigns.KTS.KTS (
    int v )
```

Initializes a new instance of the [KTS](#) class.

Parameters

v	Set size.
---	-----------

7.52.2.2 [KTS\(\)](#) [2/2]

```
TournamentPlanner.Model.Designs.BlockDesigns.KTS.KTS (
    int v,
    List< List< List< int >>> parallelClasses )
```

Initializes a new instance of the [KTS](#) class.

Parameters

<i>v</i>	Set size.
<i>parallelClasses</i>	The parallel classes.

7.52.3 Member Function Documentation

7.52.3.1 AddParallelClass()

```
void TournamentPlanner.Model.Designs.BlockDesigns.KTS.AddParallelClass (
    List< List< int >> parallelClass )
```

Add a parallel class to the list of parallel classes.

Parameters

<i>parallelClass</i>	Parallel class to add.
----------------------	------------------------

7.52.3.2 AddParallelClassAndBlocks()

```
void TournamentPlanner.Model.Designs.BlockDesigns.KTS.AddParallelClassAndBlocks (
    List< List< int >> parallelClass )
```

Add a parallel class to the list of parallel classes. At the same time, it adds the blocks of the parallel class to the list of blocks.

Parameters

<i>parallelClass</i>	Parallel class to add.
----------------------	------------------------

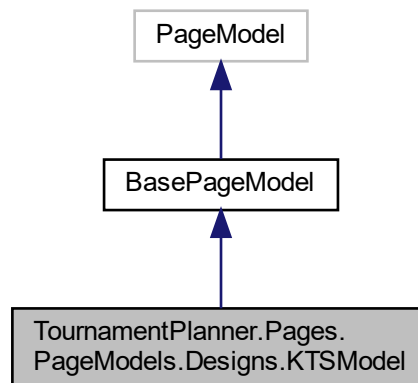
The documentation for this class was generated from the following file:

- KTS.cs

7.53 TournamentPlanner.Pages.PageModels.Designs.KTSModel Class Reference

Page model for the Kirkman triple system creation.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Designs.KTSMModel:



Public Member Functions

- void [OnPost](#) ()

Properties

- int [Multiplier](#) = 4 [get, set]
Gets or sets the multiplier.
- int [V](#) = 27 [get, set]
*Gets or sets the V - the order of the KTS to construct. Should be a multiple equal to $6 * N + 3$.*
- [KTS](#) [KTS](#) [get]
Gets the constructed KTS.

7.53.1 Detailed Description

Page model for the Kirkman triple system creation.

7.53.2 Property Documentation

7.53.2.1 KTS

[KTS](#) `TournamentPlanner.Pages.PageModels.Designs.KTSMModel.KTS` [get]

Gets the constructed KTS.

7.53.2.2 Multiplier

```
int TournamentPlanner.Pages.PageModels.Designs.KTSModel.Multiplier = 4 [get], [set]
```

Gets or sets the multiplier.

7.53.2.3 V

```
int TournamentPlanner.Pages.PageModels.Designs.KTSModel.V = 27 [get], [set]
```

Gets or sets the V - the order of the KTS to construct. Should be a multiple equal to $6 * N + 3$.

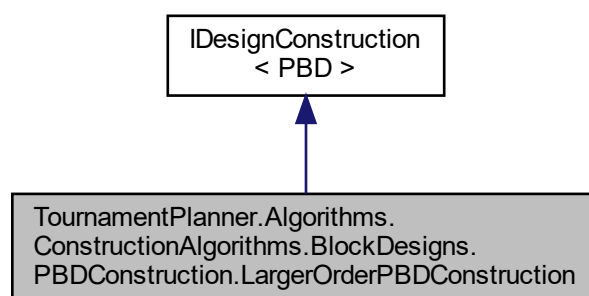
The documentation for this class was generated from the following file:

- KTS.cshtml.cs

7.54 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block Designs.PBDConstruction.LargerOrderPBDConstruction Class Reference

Construction of larger order PBDs, where the order is $V = 3N + 1$.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.Block Designs.PBDConstruction.LargerOrderPBDConstruction:



Public Member Functions

- [LargerOrderPBDConstruction](#) (int v)
Initializes a new instance of the [LargerOrderPBDConstruction](#) class.
- [PBD ConstructDesign](#) ()

7.54.1 Detailed Description

Construction of larger order PBDs, where the order is $V = 3N + 1$.

7.54.2 Constructor & Destructor Documentation

7.54.2.1 LargerOrderPBDConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.LargerOrder↔  
PBDConstruction.LargerOrderPBDConstruction (   
    int v )
```

Initializes a new instance of the [LargerOrderPBDConstruction](#) class.

Parameters

v	The order of the PBD to construct.
---	------------------------------------

The documentation for this class was generated from the following file:

- LargerOrderPBDConstruction.cs

7.55 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ BlockDesigns.PBDConstruction.LargerOrderPBDConstructionTests Class Reference

Tests for the LargerOrderPBDConstruction.

Public Member Functions

- void [TestPBD49To76Construction](#) ()
Test the construction of PBD(3n+1) from order 49 up to order 76.
- void [PBD178ConstructionTest](#) ()
Tests the construction of PBD(178).

7.55.1 Detailed Description

Tests for the LargerOrderPBDConstruction.

7.55.2 Member Function Documentation

7.55.2.1 PBD178ConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.↔
LargerOrderPBDConstructionTests.PBD178ConstructionTest ( )
```

Tests the construction of PBD(178).

7.55.2.2 TestPBD49To76Construction()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.↔
LargerOrderPBDConstructionTests.TestPBD49To76Construction ( )
```

Test the construction of PBD(3n+1) from order 49 up to order 76.

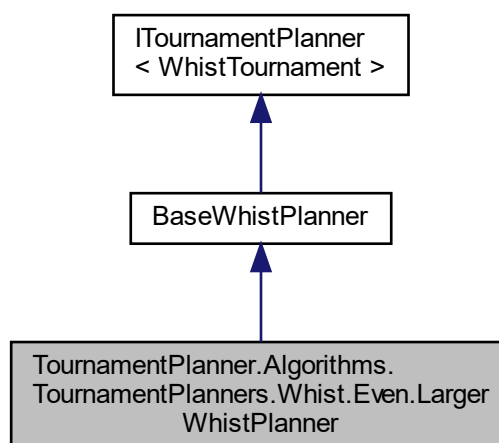
The documentation for this class was generated from the following file:

- LargerOrderPBDConstructionTests.cs

7.56 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.↔ LargerWhistPlanner Class Reference

Construct Wh(4n), where $n > 80$.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.LargerWhistPlanner:



Public Member Functions

- [LargerWhistPlanner](#) (int v)
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.56.1 Detailed Description

Construct $Wh(4n)$, where $n > 80$.

The documentation for this class was generated from the following file:

- LargerWhistPlanner.cs

7.57 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist. Even.LargerWhistPlannerTests Class Reference

Tests for the LargerWhistPlanner.

Public Member Functions

- void [Whist324ConstructionTest](#) ()

7.57.1 Detailed Description

Tests for the LargerWhistPlanner.

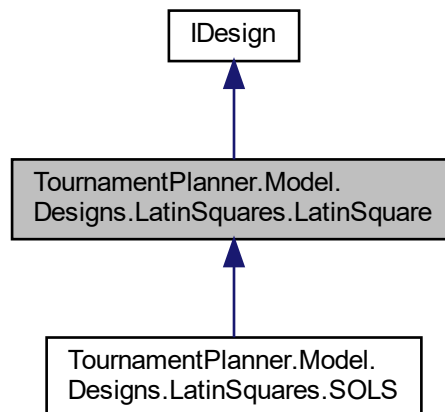
The documentation for this class was generated from the following file:

- LargerWhistPlannerTests.cs

7.58 TournamentPlanner.Model.Designs.LatinSquares.LatinSquare Class Reference

The Latin square.

Inheritance diagram for TournamentPlanner.Model.Designs.LatinSquares.LatinSquare:



Public Member Functions

- [LatinSquare](#) (int v, int[,] ls)
Initializes a new instance of the [LatinSquare](#) class.

Properties

- int **V** [get, protected set]
Gets or sets the order of the Latin square.
- int[,] **Matrix** [get, protected set]
Gets or sets the Latin square matrix. It should be a V times V matrix. Only V distinct elements should appear in it.

7.58.1 Detailed Description

The Latin square.

7.58.2 Constructor & Destructor Documentation

7.58.2.1 LatinSquare()

```

TournamentPlanner.Model.Designs.LatinSquares.LatinSquare.LatinSquare (
    int v,
    int ls[, ] )
  
```

Initializes a new instance of the [LatinSquare](#) class.

Parameters

<i>v</i>	The order od the LS.
<i>/s</i>	The LS matrix.

7.58.3 Property Documentation

7.58.3.1 Matrix

```
int [,] TournamentPlanner.Model.Designs.LatinSquares.LatinSquare.Matrix [get], [protected set]
```

Gets or sets the Latin square matrix. It should be a V times V matrix. Only V distinct elements should appear in it.

7.58.3.2 V

```
int TournamentPlanner.Model.Designs.LatinSquares.LatinSquare.V [get], [protected set]
```

Gets or sets the order of the Latin square.

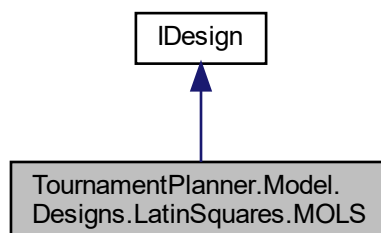
The documentation for this class was generated from the following file:

- LatinSquare.cs

7.59 TournamentPlanner.Model.Designs.LatinSquares.MOLS Class Reference

Set of mutually orthogonal Latin squares. For simplicity, it is allowed for the set of [MOLS](#) to contain only a single Latin square.

Inheritance diagram for TournamentPlanner.Model.Designs.LatinSquares.MOLS:



Public Member Functions

- [MOLS](#) (int order)
Initializes a new instance of the [MOLS](#) class.
- [MOLS](#) (int order, [LatinSquare](#) lsA, [LatinSquare](#) lsB)
Initializes a new instance of the [MOLS](#) class. Often, only 2 [MOLS](#) are given, not the whole set.
- [MOLS](#) (int order, List< [LatinSquare](#) > latinSquares)
Initializes a new instance of the [MOLS](#) class.
- [MOLS](#) (int order, [LatinSquare](#) latinSquare)
Initializes a new instance of the [MOLS](#) class.
- List< int[,]> [GetMOLSMatrices](#) ()
Retrieve the matrices of the mutually orthogonal Latin squares.
- int [GetNumberOfMOLS](#) ()
Retrieve the number of mutually orthogonal Latin squares.
- void [AddOrthogonalLatinSquare](#) ([LatinSquare](#) ls)
Add a [MOLS](#).

Properties

- int [V](#) [get]
Gets the order of the mutually orthogonal Latin squares.
- List< [LatinSquare](#) > [LatinSquares](#) [get]
Gets the list of mutually orthogonal Latin squares

7.59.1 Detailed Description

Set of mutually orthogonal Latin squares. For simplicity, it is allowed for the set of [MOLS](#) to contain only a single Latin square.

7.59.2 Constructor & Destructor Documentation

7.59.2.1 [MOLS\(\)](#) [1/4]

```
TournamentPlanner.Model.Designs.LatinSquares.MOLS.MOLS (
    int order )
```

Initializes a new instance of the [MOLS](#) class.

Parameters

<i>order</i>	The order of MOLS .
--------------	-------------------------------------

7.59.2.2 MOLS() [2/4]

```
TournamentPlanner.Model.Designs.LatinSquares.MOLS.MOLS (
    int order,
    LatinSquare lsA,
    LatinSquare lsB )
```

Initializes a new instance of the [MOLS](#) class. Often, only 2 [MOLS](#) are given, not the whole set.

Parameters

<i>order</i>	The order of MOLS .
<i>lsA</i>	First Latin square.
<i>lsB</i>	Second Latin square.

7.59.2.3 MOLS() [3/4]

```
TournamentPlanner.Model.Designs.LatinSquares.MOLS.MOLS (
    int order,
    List< LatinSquare > latinSquares )
```

Initializes a new instance of the [MOLS](#) class.

Parameters

<i>order</i>	The order of MOLS .
<i>latinSquares</i>	A list of mutually orthogonal Latin squares.

7.59.2.4 MOLS() [4/4]

```
TournamentPlanner.Model.Designs.LatinSquares.MOLS.MOLS (
    int order,
    LatinSquare latinSquare )
```

Initializes a new instance of the [MOLS](#) class.

Parameters

<i>order</i>	The order of MOLS .
<i>latinSquare</i>	A single Latin square.

7.59.3 Member Function Documentation

7.59.3.1 AddOrthogonalLatinSquare()

```
void TournamentPlanner.Model.Designs.LatinSquares.MOLS.AddOrthogonalLatinSquare (
    LatinSquare ls )
```

Add a [MOLS](#).

Parameters

<i>ls</i>	Latin square that should be orthogonal to the others in the list.
-----------	---

7.59.3.2 GetMOLSMatrices()

```
List<int[,]> TournamentPlanner.Model.Designs.LatinSquares.MOLS.GetMOLSMatrices ( )
```

Retrieve the matrices of the mutually orthogonal Latin squares.

Returns

List of [MOLS](#) matrices.

7.59.3.3 GetNumberOfMOLS()

```
int TournamentPlanner.Model.Designs.LatinSquares.MOLS.GetNumberOfMOLS ( )
```

Retrieve the number of mutually orthogonal Latin squares.

Returns

Number of [MOLS](#).

7.59.4 Property Documentation

7.59.4.1 LatinSquares

```
List<LatinSquare> TournamentPlanner.Model.Designs.LatinSquares.MOLS.LatinSquares [get]
```

Gets the list of mutually orthogonal Latin squares

7.59.4.2 V

```
int TournamentPlanner.Model.Designs.LatinSquares.MOLS.V [get]
```

Gets the order of the mutually orthogonal Latin squares.

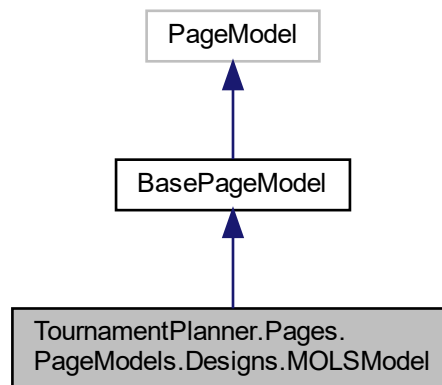
The documentation for this class was generated from the following file:

- MOLS.cs

7.60 TournamentPlanner.Pages.PageModels.Designs.MOLSModel Class Reference

Page model for mutually orthogonal Latin squares.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Designs.MOLSModel:



Public Member Functions

- void [OnPost](#) ()

Properties

- [MOLS](#) [MOLS](#) [get]
Gets the constructed MOLS.
- [OrthogonalArray](#) [OA](#) [get]
Gets the orthogonal array equivalent to the constructed MOLS.
- int [Order](#) = 4 [get, set]
Gets or sets the order of the Latin square.
- [MOLSMethod](#) [MOLSConstructionMethod](#) = [MOLSMethod](#).Automatic [get, set]
Gets or sets the construction method for the Latin square.

7.60.1 Detailed Description

Page model for mutually orthogonal Latin squares.

7.60.2 Property Documentation

7.60.2.1 MOLS

`MOLS` `TournamentPlanner.Pages.PageModels.Designs.MOLSModel.MOLS` [get]

Gets the constructed MOLS.

7.60.2.2 MOLSConstructionMethod

`MOLSMethod` `TournamentPlanner.Pages.PageModels.Designs.MOLSModel.MOLSConstructionMethod` = `MOLSMethod.Automatic` [get], [set]

Gets or sets the construction method for the Latin square.

7.60.2.3 OA

`OrthogonalArray` `TournamentPlanner.Pages.PageModels.Designs.MOLSModel.OA` [get]

Gets the orthogonal array equivalent to the constructed MOLS.

7.60.2.4 Order

`int` `TournamentPlanner.Pages.PageModels.Designs.MOLSModel.Order` = 4 [get], [set]

Gets or sets the order of the Latin square.

The documentation for this class was generated from the following file:

- MOLS.cshtml.cs

7.61 TournamentPlannerTests.Designs.MOLSSetValidatorTests Class Reference

Tests for the MOLSSetValidator.

Public Member Functions

- void [ValidMOLSTest](#) ()
Test where the MOLS are valid.
- void [InvalidMOLSTest](#) ()
Test where some of the MOLS are not valid.

7.61.1 Detailed Description

Tests for the MOLSSetValidator.

7.61.2 Member Function Documentation

7.61.2.1 InvalidMOLSTest()

```
void TournamentPlannerTests.Designs.MOLSSetValidatorTests.InvalidMOLSTest ( )
```

Test where some of the MOLS are not valid.

7.61.2.2 ValidMOLSTest()

```
void TournamentPlannerTests.Designs.MOLSSetValidatorTests.ValidMOLSTest ( )
```

Test where the MOLS are valid.

The documentation for this class was generated from the following file:

- MOLSSetValidatorTests.cs

7.62 TournamentPlanner.Datastore.MongoMOLS Class Reference

Class representing a MOLS in mongo DB.

Properties

- int [Order](#) [get, set]
Gets or sets the order of MOLS.
- int [NbMOLS](#) [get, set]
Gets or sets the number of MOLS.
- List< int[,]> [Matrices](#) [get, set]
Gets or sets the Latin square matrices of the MOLS.

7.62.1 Detailed Description

Class representing a MOLS in mongo DB.

7.62.2 Property Documentation

7.62.2.1 Matrices

```
List<int[,]> TournamentPlanner.Datastore.MongoMOLS.Matrices [get], [set]
```

Gets or sets the Latin square matrices of the MOLS.

7.62.2.2 NbMOLS

```
int TournamentPlanner.Datastore.MongoMOLS.NbMOLS [get], [set]
```

Gets or sets the number of MOLS.

7.62.2.3 Order

```
int TournamentPlanner.Datastore.MongoMOLS.Order [get], [set]
```

Gets or sets the order of MOLS.

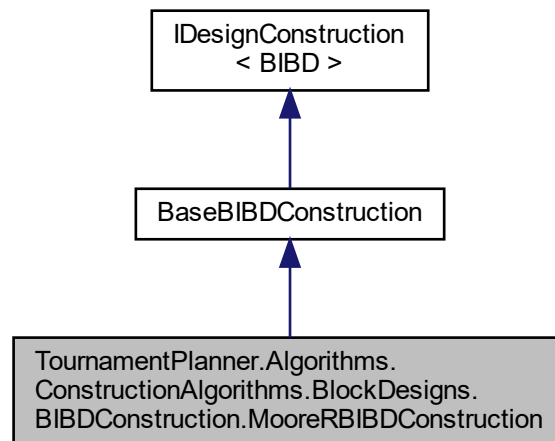
The documentation for this class was generated from the following file:

- MongoMOLS.cs

7.63 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.MooreRBIBDConstruction Class Reference

Construction of RBIBD($3q+1, 4, 1$) where q is a prime power and can be written as $q = 4m + 1$. These RBIBD are used to construct Whist tournaments.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.MooreRBIBDConstruction:



Public Member Functions

- [MooreRBIBDConstruction](#) (int v)
Initializes a new instance of the [MooreRBIBDConstruction](#) class.
- override [RBIBD ConstructDesign](#) ()
Construct a resolvable BIBD.

Additional Inherited Members

7.63.1 Detailed Description

Construction of RBIBD($3q+1, 4, 1$) where q is a prime power and can be written as $q = 4m + 1$. These RBIBD are used to construct Whist tournaments.

7.63.2 Constructor & Destructor Documentation

7.63.2.1 MooreRBIBDConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.MooreRBIBDConstruction.MooreRBIBDConstruction (
    int v )
```

Initializes a new instance of the [MooreRBIBDConstruction](#) class.

Parameters

v	Order of the RBIBD to construct.
---	----------------------------------

7.63.3 Member Function Documentation

7.63.3.1 ConstructDesign()

```
override RBIBD TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.MooreRBIBDConstruction.ConstructDesign ( ) [virtual]
```

Construct a resolvable BIBD.

Returns

Constructed RBIBD.

Implements [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction](#).

The documentation for this class was generated from the following file:

- MooreRBIBDConstruction.cs

7.64 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.MooreRBIBDConstructionTests Class Reference

Tests for the MooreRBIBDConstruction.

Public Member Functions

- void [ConstructionTestForSeveralOrdersOfRBIBD](#) ()
Test the PBIBD construction for all the important orders.

7.64.1 Detailed Description

Tests for the MooreRBIBDConstruction.

7.64.2 Member Function Documentation

7.64.2.1 ConstructionTestForSeveralOrdersOfRBIBD()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.  
MooreRBIBDConstructionTests.ConstructionTestForSeveralOrdersOfRBIBD ( )
```

Test the PBIBD construction for all the important orders.

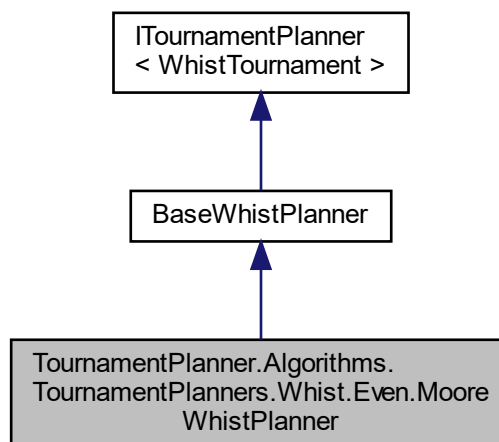
The documentation for this class was generated from the following file:

- MooreRBIBDConstructionTests.cs

7.65 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlanner Class Reference

This class implements the construction of $Wh(v = 4n)$, where $v = 3q + 1$ and $q = 1 \pmod{4} = 4m + 1$ and is a prime power. $v = 28, 40, 52, 76, 88 \dots$ https://www.jstor.org/stable/2369860?seq=1::metadata_info_tab_contents.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlanner:



Public Member Functions

- [MooreWhistPlanner](#) (int v)
Initializes a new instance of the [MooreWhistPlanner](#) class.
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.65.1 Detailed Description

This class implements the construction of $Wh(v = 4n)$, where $v = 3q + 1$ and $q = 1 \pmod{4} = 4m + 1$ and is a prime power. $v = 28, 40, 52, 76, 88 \dots$ https://www.jstor.org/stable/2369860?seq=1::metadata_info_tab_contents.

7.65.2 Constructor & Destructor Documentation

7.65.2.1 MooreWhistPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlanner.MooreWhistPlanner
(
    int v )
```

Initializes a new instance of the [MooreWhistPlanner](#) class.

Parameters

v	Number of players.
---	--------------------

The documentation for this class was generated from the following file:

- [MooreWhistPlanner.cs](#)

7.66 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlannerTests Class Reference

Tests for the MooreWhistPlanner.

Public Member Functions

- void [TestValidOrdersOfMooreConstruction](#) ()
Test with the valid orders of whist tournaments for the Moore construction.

7.66.1 Detailed Description

Tests for the MooreWhistPlanner.

7.66.2 Member Function Documentation

7.66.2.1 TestValidOrdersOfMooreConstruction()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.MooreWhistPlannerTests.TestValidOrdersOfMooreConstruction ( )
```

Test with the valid orders of whist tournaments for the Moore construction.

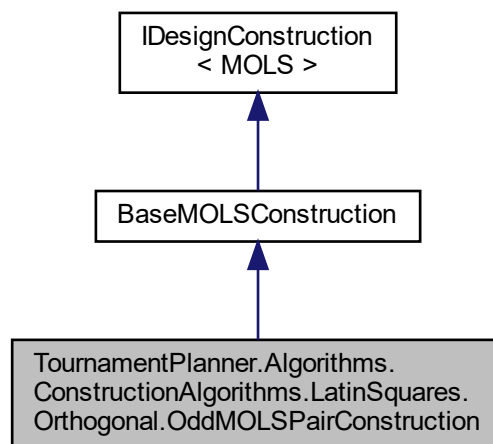
The documentation for this class was generated from the following file:

- MooreWhistPlannerTests.cs

7.67 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPairConstruction Class Reference

Algorithm to construct pairs of mutually orthogonal Latin squares of odd order. https://www.whitman.edu/mathematics/cgt_online/book/section04.03.html

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPairConstruction:



Public Member Functions

- [OddMOLSPairConstruction](#) (int order)
- override [MOLS ConstructDesign](#) ()

Construct MOLS.

Protected Member Functions

- override bool [ValidateOrder](#) (int order)

Method to validate the order in the constructor.

Properties

- override string [ArgumentExceptionMessage](#) [get]

7.67.1 Detailed Description

Algorithm to construct pairs of mutually orthogonal Latin squares of odd order. https://www.whitman.edu/mathematics/cgt_online/book/section04.03.html

The documentation for this class was generated from the following file:

- OddMOLSPairConstruction.cs

7.68 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPairConstructionTests Class Reference

Tests for the OddMOLSPairConstruction.

Public Member Functions

- void [EvenOrderShouldThrowExceptionTest](#) ()
Test that an exception is thrown when the order is even.
- void [OrthogonalityTest](#) ()
Test the orthogonality of a few pairs of MOLS of odd orders.

7.68.1 Detailed Description

Tests for the OddMOLSPairConstruction.

7.68.2 Member Function Documentation

7.68.2.1 EvenOrderShouldThrowExceptionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPairConstructionTests.EvenOrderShouldThrowExceptionTest ( )
```

Test that an exception is thrown when the order is even.

7.68.2.2 OrthogonalityTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.OddMOLSPairConstructionTests.OrthogonalityTest ( )
```

Test the orthogonality of a few pairs of MOLS of odd orders.

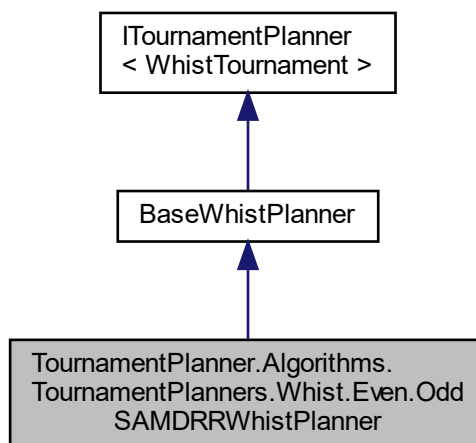
The documentation for this class was generated from the following file:

- OddMOLSPairConstructionTests.cs

7.69 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.OddSAMDRRWhistPlanner Class Reference

Use odd order SAMMDR to construct several orders od Whist(4n).

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.OddSAMDRRWhistPlanner:



Public Member Functions

- [OddSAMDRRWhistPlanner](#) (int v)
Initializes a new instance of the [OddSAMDRRWhistPlanner](#) class.
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.69.1 Detailed Description

Use odd order SAMMDR to construct several orders od Whist(4n).

7.69.2 Constructor & Destructor Documentation

7.69.2.1 OddSAMDRRWhistPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.OddSAMDRRWhistPlanner.OddSAMDRRWhist↔
Planner (
    int v )
```

Initializes a new instance of the [OddSAMDRRWhistPlanner](#) class.

Parameters

v	The order.
---	------------

The documentation for this class was generated from the following file:

- [OddSAMDRRWhistPlanner.cs](#)

7.70 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.↔ Even.OddSAMDRRWhistPlannerTests Class Reference

Tests for the SAMDRRWhistPlanner.

Public Member Functions

- void [TestWhistConstructionUsingSAMDRR](#) ()
Test the construction of Whist(44).

7.70.1 Detailed Description

Tests for the SAMDRRWhistPlanner.

7.70.2 Member Function Documentation

7.70.2.1 TestWhistConstructionUsingSAMDRR()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.OddSAMDRRWhistPlanner←
Tests.TestWhistConstructionUsingSAMDRR ( )
```

Test the construction of Whist(44).

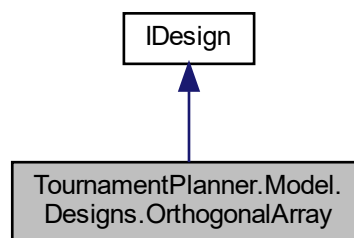
The documentation for this class was generated from the following file:

- OddSAMDRRWhistPlannerTests.cs

7.71 TournamentPlanner.Model.Designs.OrthogonalArray Class Reference

Class representing the Orthogonal array structure. In some literature, this representation is actually the transpose of an orthogonal array.

Inheritance diagram for TournamentPlanner.Model.Designs.OrthogonalArray:



Public Member Functions

- [OrthogonalArray](#) (int v, int[,] matrix)
Initializes a new instance of the [OrthogonalArray](#) class.
- int [Width](#) ()
Gets the width of the orthogonal array.
- int [Height](#) ()
Gets the height of the orthogonal array.
- int [RepresentedMOLSCount](#) ()
Get the number of MOLS this orthogonal array might represent.

Properties

- `int V` [get]
Gets the level of the orthogonal array.
- `int[,] Matrix` [get]
Gets the matrix representation of the OA.

7.71.1 Detailed Description

Class representing the Orthogonal array structure. In some literature, this representation is actually the transpose of an orthogonal array.

7.71.2 Constructor & Destructor Documentation

7.71.2.1 OrthogonalArray()

```
TournamentPlanner.Model.Designs.OrthogonalArray.OrthogonalArray (
    int v,
    int matrix[,])
```

Initializes a new instance of the [OrthogonalArray](#) class.

Parameters

<code>v</code>	The level of the orthogonal array.
<code>matrix</code>	The matrix representing the orthogonal array.

7.71.3 Member Function Documentation

7.71.3.1 Height()

```
int TournamentPlanner.Model.Designs.OrthogonalArray.Height ( )
```

Gets the height of the orthogonal array.

Returns

Height of the OA matrix.

7.71.3.2 RepresentedMOLSCount()

```
int TournamentPlanner.Model.Designs.OrthogonalArray.RepresentedMOLSCount ( )
```

Get the number of MOLS this orthogonal array might represent.

Returns

Number of MOLS.

7.71.3.3 Width()

```
int TournamentPlanner.Model.Designs.OrthogonalArray.Width ( )
```

Gets the width of the orthogonal array.

Returns

Width of the OA matrix.

7.71.4 Property Documentation

7.71.4.1 Matrix

```
int [,] TournamentPlanner.Model.Designs.OrthogonalArray.Matrix [get]
```

Gets the matrix representation of the OA.

7.71.4.2 V

```
int TournamentPlanner.Model.Designs.OrthogonalArray.V [get]
```

Gets the level of the orthogonal array.

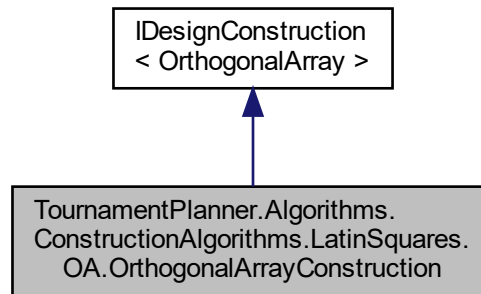
The documentation for this class was generated from the following file:

- OrthogonalArray.cs

7.72 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.OrthogonalArrayConstruction Class Reference

Construction method for OAs using MOLS.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.OrthogonalArrayConstruction:



Public Member Functions

- [OrthogonalArrayConstruction](#) (int k, int m)
Initializes a new instance of the [OrthogonalArrayConstruction](#) class.
- [OrthogonalArray ConstructDesign](#) ()
Construct the orthogonal array.

7.72.1 Detailed Description

Construction method for OAs using MOLS.

7.72.2 Constructor & Destructor Documentation

7.72.2.1 OrthogonalArrayConstruction()

```

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.OrthogonalArrayConstruction.OrthogonalArrayConstruction (
    int k,
    int m )
  
```

Initializes a new instance of the [OrthogonalArrayConstruction](#) class.

Parameters

k	The factor.
m	The level.

7.72.3 Member Function Documentation

7.72.3.1 ConstructDesign()

[OrthogonalArray](#) TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.Orthogonal↔
ArrayConstruction.ConstructDesign ()

Construct the orthogonal array.

Returns

Constructed orthogonal array.

The documentation for this class was generated from the following file:

- OrthogonalArrayConstruction.cs

7.73 TournamentPlannerTests.Utils.OrthogonalArrayMOLSCConverter↔ Tests Class Reference

Tests for the OrthogonalArrayMOLSCConverter.

Public Member Functions

- void [MOLS2OATest](#) ()
Test the conversion of MOLS to OA.
- void [OA2MOLSTest](#) ()
Test the conversion of an OA into MOLS.

7.73.1 Detailed Description

Tests for the OrthogonalArrayMOLSCConverter.

7.73.2 Member Function Documentation

7.73.2.1 MOLS2OATest()

```
void TournamentPlannerTests.Utils.OrthogonalArrayMOLSConverterTests.MOLS2OATest ( )
```

Test the conversion of MOLS to OA.

7.73.2.2 OA2MOLSTest()

```
void TournamentPlannerTests.Utils.OrthogonalArrayMOLSConverterTests.OA2MOLSTest ( )
```

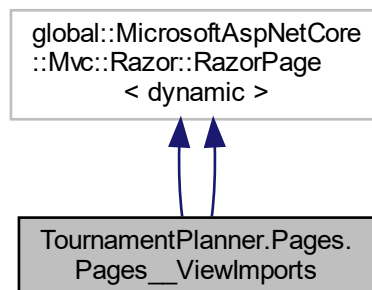
Test the conversion of an OA into MOLS.

The documentation for this class was generated from the following file:

- OrthogonalArrayMOLSConverterTests.cs

7.74 TournamentPlanner.Pages.Pages__ViewImports Class Reference

Inheritance diagram for TournamentPlanner.Pages.Pages__ViewImports:



Public Member Functions

- async override `global::System.Threading.Tasks.Task ExecuteAsync ()`
- async override `global::System.Threading.Tasks.Task ExecuteAsync ()`

Properties

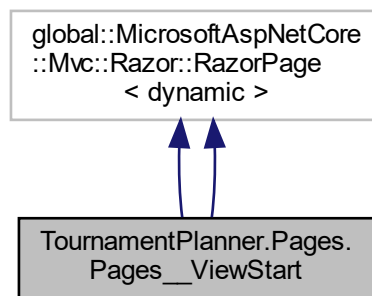
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< dynamic > **Html** [get]

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/_ViewImports.cshtml.g.cs

7.75 TournamentPlanner.Pages.Pages__ViewStart Class Reference

Inheritance diagram for TournamentPlanner.Pages.Pages__ViewStart:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

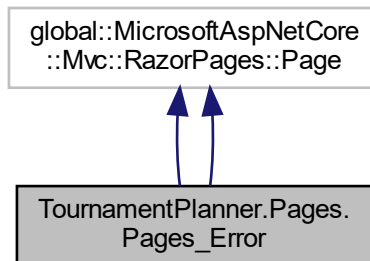
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< dynamic > **Html** [get]

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/_ViewStart.cshtml.g.cs

7.76 TournamentPlanner.Pages.Pages_Error Class Reference

Inheritance diagram for TournamentPlanner.Pages.Pages_Error:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< **ErrorModel** > **Html** [get]
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< **ErrorModel** > **ViewData** [get]
- **ErrorModel Model** [get]

7.76.1 Property Documentation

7.76.1.1 ModelExpressionProvider

```
Microsoft.AspNetCore.Mvc.ViewFeatures.global.IModelExpressionProvider TournamentPlanner.Pages.Pages_Error.ModelExpressionProvider [get]
```

Initial value:

```
<h3>Development Mode</h3>
```

```
<p>
```

```
Swapping to the <strong>Development</strong> environment displays detailed information about the error that occurred.
```

```
</p>
```

```
<p>
```

```

<strong>The Development environment shouldn't be enabled for deployed applications.</strong>
It can result in displaying sensitive information from exceptions to end users.
For local debugging, enable the <strong>Development</strong> environment by setting the
<strong>ASPNETCORE_ENVIRONMENT</strong> environment variable to <strong>Development</strong>
and restarting the app.
</p>
"

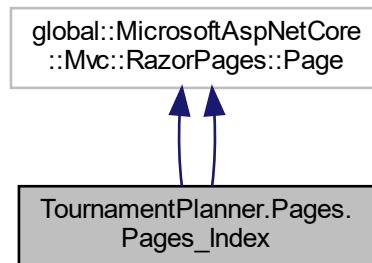
```

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/Error.cshtml.g.cs

7.77 TournamentPlanner.Pages.Pages_Index Class Reference

Inheritance diagram for TournamentPlanner.Pages.Pages_Index:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< [IndexModel](#) > **Html** [get]
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< [IndexModel](#) > **ViewData** [get]
- [IndexModel](#) **Model** [get]

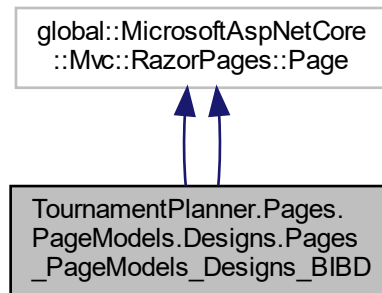
The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/Index.cshtml.g.cs

7.78 TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_BIBD Class Reference ↔

Models_Designs_BIBD Class Reference

Inheritance diagram for TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_BIBD:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

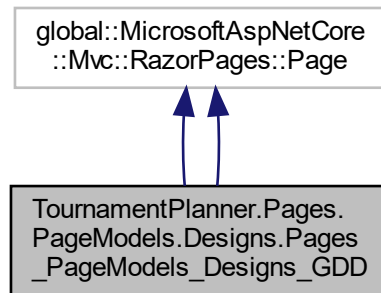
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< [TournamentPlanner.Pages.PageModels.Designs.BIBDModel](#) > **Html** [get]
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< [TournamentPlanner.Pages.PageModels.Designs.BIBDModel](#) > **ViewData** [get]
- [TournamentPlanner.Pages.PageModels.Designs.BIBDModel](#) **Model** [get]

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/PageModels/Designs/BIBD.cshtml.g.cs

7.79 TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_GDD Class Reference

Inheritance diagram for TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_GDD:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< [TournamentPlanner.Pages.PageModels.Designs.GDDModel](#) > **Html** [get]
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< [TournamentPlanner.Pages.PageModels.Designs.GDDModel](#) > **ViewData** [get]
- [TournamentPlanner.Pages.PageModels.Designs.GDDModel](#) **Model** [get]

7.79.1 Property Documentation

7.79.1.1 ModelExpressionProvider

Microsoft.AspNetCore.Mvc.ViewFeatures.global.IModelExpressionProvider TournamentPlanner.↔
 Pages.PageModels.Designs.Pages_PageModels_Designs_GDD.ModelExpressionProvider [get]

Initial value:

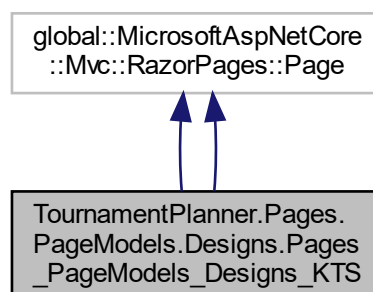
```
<div class="content-section">
  <div class="content-section-header">
    <h4>Group Divisible Designs</h4>
  </div>
  <div class="content-section-body">
    A <i>Group divisible design</i> or <b>GDD</b> of order  $\{v\}$  consists of a  $\{v\}$ -sized set  $\{S\}$ 
    of elements,
    a collection of subsets  $\{G\}$  called <b>groups</b> and a collection of blocks  $\{B\}$  such that
    <ul>
      <li>the groups form a partition of  $\{S\}$ ;</li>
      <li>each pair of elements from the same group do not occur together in any of the
        blocks;</li>
      <li>each pair of elements from different groups occur together in exactly one of the
        blocks.</li>
    </ul>
    Resolvable GDDs can be constructed using MOLES. This construction is possible for any  $\{v = 3m\}$ 
    except  $\{m = 2, 6\}$ . They are
    then used to schedule partially balanced tournaments with three player games.
  </div>
</div>
<div class="c" <div class="content-section">
  <div class="content-section-header">
    <h4>Constructed resolvable GDD with block size 3</h4>
  </div>
  <div class="content-section-body">
    <table class="result-table">
      <tr>
        <td>V</td>
        <td>"
```

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/PageModels/Designs/GDD.cshtml.g.cs

7.80 TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_KTS Class Reference

Inheritance diagram for TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_KTS:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

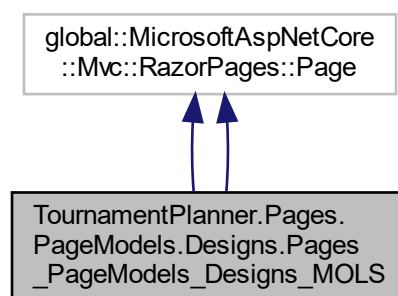
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.UrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< TournamentPlanner.Pages.PageModels.Designs.KTSMModel > **Html** [get]
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< TournamentPlanner.Pages.PageModels.Designs.KTSMModel > **ViewData** [get]
- TournamentPlanner.Pages.PageModels.Designs.KTSMModel **Model** [get]

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/PageModels/Designs/KTS.cshtml.g.cs

7.81 TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_MOLS Class Reference

Inheritance diagram for TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_MOLS:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

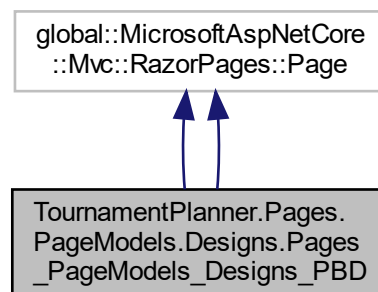
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider` **ModelExpressionProvider** [get]
- `global::Microsoft.AspNetCore.Mvc.IUrlHelper` **Url** [get]
- `global::Microsoft.AspNetCore.Mvc.IViewComponentHelper` **Component** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper` **Json** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< TournamentPlanner.Pages.PageModels.Designs.MOLSMModel >` **Html** [get]
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< TournamentPlanner.Pages.PageModels.Designs.MOLSMModel >` **ViewData** [get]
- `TournamentPlanner.Pages.PageModels.Designs.MOLSMModel` **Model** [get]

The documentation for this class was generated from the following file:

- `Debug/net5.0/Razor/Pages/PageModels/Designs/MOLS.cshtml.g.cs`

7.82 TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_PBD Class Reference

Inheritance diagram for `TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_PBD`:



Public Member Functions

- async override `global::System.Threading.Tasks.Task` **ExecuteAsync** ()
- async override `global::System.Threading.Tasks.Task` **ExecuteAsync** ()

Properties

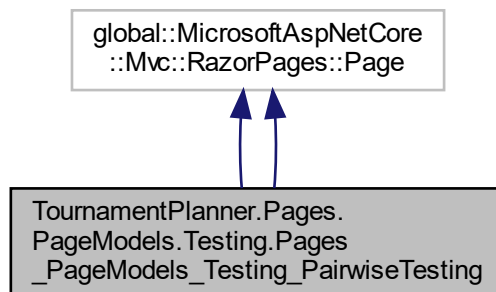
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.UrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< [TournamentPlanner.Pages.PageModels.Designs.PBDModel](#) > **Html** [get]
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< [TournamentPlanner.Pages.PageModels.Designs.PBDModel](#) > **ViewData** [get]
- [TournamentPlanner.Pages.PageModels.Designs.PBDModel](#) **Model** [get]

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/PageModels/Designs/PBD.cshtml.g.cs

7.83 TournamentPlanner.Pages.PageModels.Testing.Pages_PageModels_Testing_PairwiseTesting Class Reference

Inheritance diagram for TournamentPlanner.Pages.PageModels.Testing.Pages_PageModels_Testing_PairwiseTesting:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

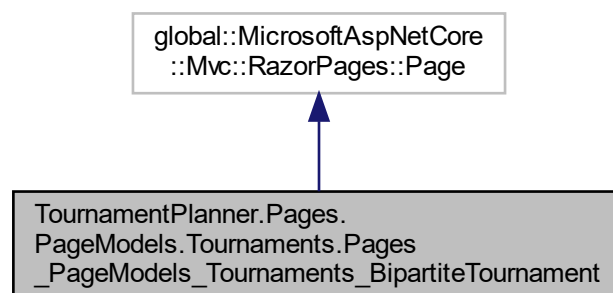
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider` **ModelExpressionProvider** [get]
- `global::Microsoft.AspNetCore.Mvc.IUrlHelper` **Url** [get]
- `global::Microsoft.AspNetCore.Mvc.IViewComponentHelper` **Component** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper` **Json** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel >` **Html** [get]
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel >` **ViewData** [get]
- `TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel` **Model** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< TournamentPlanner.Pages.PageModels.Testing.PairwiseTesting2Model >` **Html** [get]
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< TournamentPlanner.Pages.PageModels.Testing.PairwiseTesting2Model >` **ViewData** [get]
- `TournamentPlanner.Pages.PageModels.Testing.PairwiseTesting2Model` **Model** [get]

The documentation for this class was generated from the following file:

- `Debug/net5.0/Razor/Pages/PageModels/Testing/PairwiseTesting.cshtml.g.cs`

7.84 TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_BipartiteTournament Class Reference

Inheritance diagram for `TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_BipartiteTournament`:



Public Member Functions

- async override `global::System.Threading.Tasks.Task` **ExecuteAsync** ()

Properties

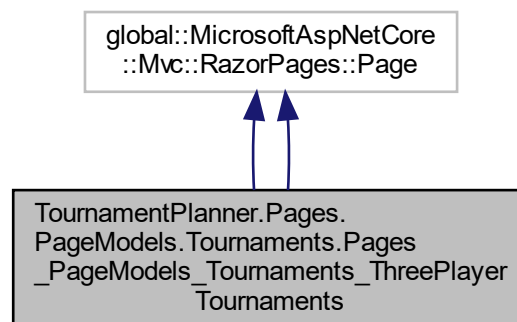
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider` **ModelExpressionProvider** [get]
- `global::Microsoft.AspNetCore.Mvc.IUrlHelper` **Url** [get]
- `global::Microsoft.AspNetCore.Mvc.IViewComponentHelper` **Component** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper` **Json** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournamentModel >` **Html** [get]
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournamentModel >` **ViewData** [get]
- `TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournamentModel` **Model** [get]

The documentation for this class was generated from the following file:

- `Debug/net5.0/Razor/Pages/PageModels/Tournaments/BipartiteTournament.cshtml.g.cs`

7.85 TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_ThreePlayerTournaments Class Reference

Inheritance diagram for TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_ThreePlayerTournaments:



Public Member Functions

- async override `global::System.Threading.Tasks.Task` **ExecuteAsync** ()
- async override `global::System.Threading.Tasks.Task` **ExecuteAsync** ()

Properties

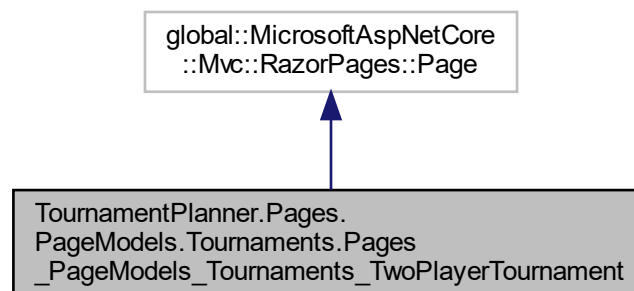
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider` **ModelExpressionProvider** [get]
- `global::Microsoft.AspNetCore.Mvc.IUrlHelper` **Url** [get]
- `global::Microsoft.AspNetCore.Mvc.IViewComponentHelper` **Component** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper` **Json** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournamentsModel >` **Html** [get]
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournamentsModel >` **ViewData** [get]
- `TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournamentsModel` **Model** [get]

The documentation for this class was generated from the following file:

- `Debug/net5.0/Razor/Pages/PageModels/Tournaments/ThreePlayerTournaments.cshtml.g.cs`

7.86 TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_TwoPlayerTournament Class Reference

Inheritance diagram for `TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_TwoPlayerTournament`:



Public Member Functions

- `async override global::System.Threading.Tasks.Task ExecuteAsync ()`

Properties

- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider` **ModelExpressionProvider** [get]
- `global::Microsoft.AspNetCore.Mvc.IUrlHelper` **Url** [get]
- `global::Microsoft.AspNetCore.Mvc.IViewComponentHelper` **Component** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper` **Json** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< PageModels.Tournaments.TwoPlayerTournamentModel >` **Html** [get]
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< PageModels.Tournaments.TwoPlayerTournamentModel >` **ViewData** [get]
- `PageModels.Tournaments.TwoPlayerTournamentModel` **Model** [get]

7.86.1 Property Documentation

7.86.1.1 ModelExpressionProvider

```
global.Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider TournamentPlanner.↔  
Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_TwoPlayerTournament.ModelExpression↔  
Provider [get]
```

Initial value:

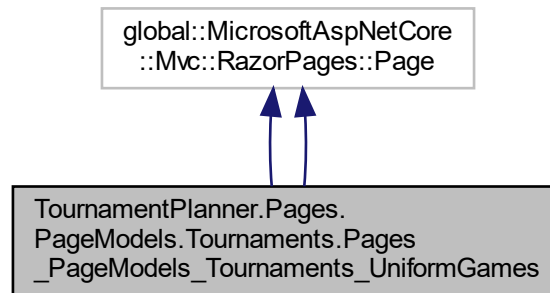
```
<div class="content">  
  <div class="content-section">  
    <div class="content-section-header">  
      <h4>Round Robin Tournament</h4>  
    </div>  
    <div class="content-section-body">  
      <p>  
        A round-robin tournament (or all-play-all tournament) is a competition in which each  
        contestant meets every other participant,  
        usually in turn. A round-robin contrasts with an elimination tournament, in which  
        participants are eliminated after  
        a certain number of losses. <a  
        href="https://en.wikipedia.org/wiki/Round-robin_tournament">wiki</a>  
      </p>  
    </div>  
  </div>  
  <div class="content-section">  
    <div class="content-section-header">  
      <h4>Construct Tournament</h4>  
    </div>  
    <div class="content-section-body">  
      <div class="row">  
        <div class="col-md-4">  
          <div class="content-section">  
            <div class="content-section-header">  
              <h4>Resulting Tournament</h4>  
            </div>  
            <div class="content-section-body">  
              <table class="result-table">  
                "
```

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/PageModels/Tournaments/TwoPlayerTournament.cshtml.g.cs

7.87 TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_UniformGames Class Reference

Inheritance diagram for TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_UniformGames:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

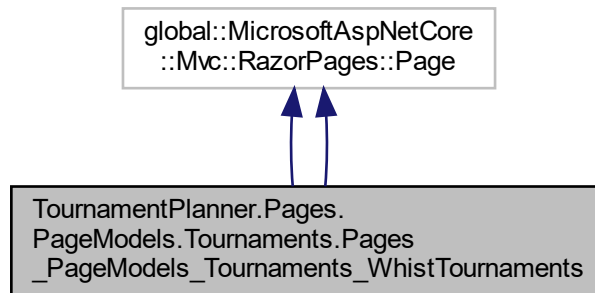
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< [TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel](#) > **Html** [get]
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< [TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel](#) > **ViewData** [get]
- [TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel](#) **Model** [get]

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/PageModels/Tournaments/UniformGames.cshtml.g.cs

7.88 TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_WhistTournaments Class Reference

Inheritance diagram for TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_WhistTournaments:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

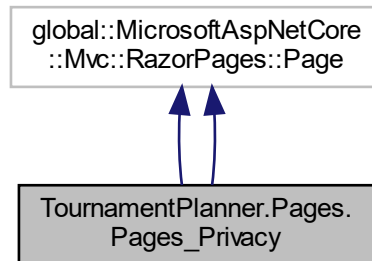
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel > **Html** [get]
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel > **ViewData** [get]
- TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel **Model** [get]

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/PageModels/Tournaments/WhistTournaments.cshtml.g.cs

7.89 TournamentPlanner.Pages.Pages_Privacy Class Reference

Inheritance diagram for TournamentPlanner.Pages.Pages_Privacy:



Public Member Functions

- async override `global::System.Threading.Tasks.Task ExecuteAsync ()`
- async override `global::System.Threading.Tasks.Task ExecuteAsync ()`

Properties

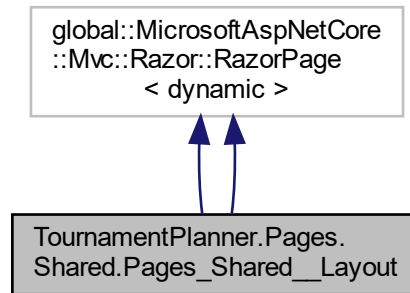
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider` **ModelExpressionProvider** [get]
- `global::Microsoft.AspNetCore.Mvc.IUrlHelper` **Url** [get]
- `global::Microsoft.AspNetCore.Mvc.IViewComponentHelper` **Component** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper` **Json** [get]
- `global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< PrivacyModel >` **Html** [get]
- `global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< PrivacyModel >` **ViewData** [get]
- `PrivacyModel` **Model** [get]

The documentation for this class was generated from the following file:

- `Debug/net5.0/Razor/Pages/Privacy.cshtml.g.cs`

7.90 TournamentPlanner.Pages.Shared.Pages_Shared__Layout Class Reference

Inheritance diagram for TournamentPlanner.Pages.Shared.Pages_Shared__Layout:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.UrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.ViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper<dynamic> **Html** [get]

7.90.1 Property Documentation

7.90.1.1 ModelExpressionProvider

Microsoft.AspNetCore.Mvc.ViewFeatures.global.IModelExpressionProvider TournamentPlanner.Pages.Shared.Pages_Shared__Layout.ModelExpressionProvider [get]

Initial value:

```

<button class="navbar-toggler" type="button" data-toggle="collapse"
data-target=".navbar-collapse" aria-controls="navbarSupportedContent"
aria-expanded="false" aria-label="Toggle navigation">
  <span class="navbar-toggler-icon"></span>
</button>

```

```

</nav>
<div class="all">
  <nav class="left">
    <ul class="content">
      <li class="nav-head"><b>Designs</b></li>
      <li class="nav-item">
        "
    </ul>
  </nav>
  <script>
    var sandwichToggler = document.querySelector("#sandwich");
    sandwichToggler.addEventListener('click', toggleContentMenu, false);
    function toggleContentMenu() {
      let leftMenu = document.querySelector('.left');
      leftMenu.classList.toggle('toggled');
    }
  </script>
  "

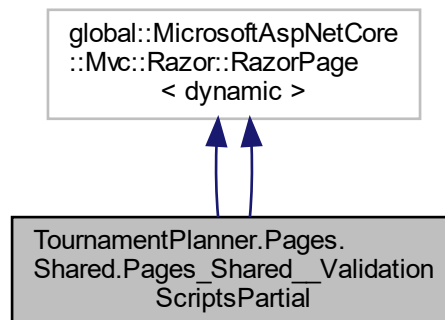
```

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/Shared/_Layout.cshtml.g.cs

7.91 TournamentPlanner.Pages.Shared.Pages_Shared__Validation< ScriptsPartial Class Reference

Inheritance diagram for TournamentPlanner.Pages.Shared.Pages_Shared__ValidationScriptsPartial:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()
- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

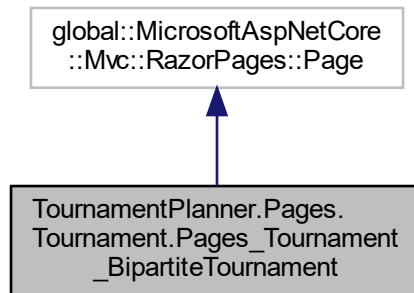
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< dynamic > **Html** [get]

The documentation for this class was generated from the following file:

- Debug/net5.0/Razor/Pages/Shared/_ValidationScriptsPartial.cshtml.g.cs

7.92 TournamentPlanner.Pages.Tournament.Pages_Tournament_↔ BipartiteTournament Class Reference

Inheritance diagram for TournamentPlanner.Pages.Tournament.Pages_Tournament_BipartiteTournament:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

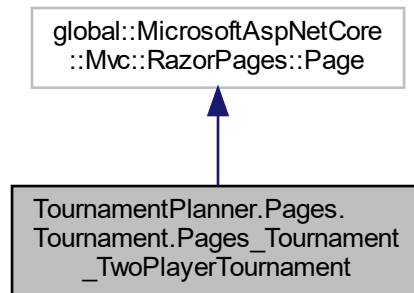
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< TournamentPlanner.Pages.Tournament_↔ BipartiteTournamentModel > **Html** [get]
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< TournamentPlanner.Pages_↔ Tournament.BipartiteTournamentModel > **ViewData** [get]
- TournamentPlanner.Pages.Tournament.BipartiteTournamentModel **Model** [get]

The documentation for this class was generated from the following file:

- Release/net5.0/Razor/Pages/Tournament/BipartiteTournament.cshtml.g.cs

7.93 TournamentPlanner.Pages.Tournament.Pages_Tournament_TwoPlayerTournament Class Reference

Inheritance diagram for TournamentPlanner.Pages.Tournament.Pages_Tournament_TwoPlayerTournament:



Public Member Functions

- async override global::System.Threading.Tasks.Task **ExecuteAsync** ()

Properties

- global::Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider **ModelExpressionProvider** [get]
- global::Microsoft.AspNetCore.Mvc.IUrlHelper **Url** [get]
- global::Microsoft.AspNetCore.Mvc.IViewComponentHelper **Component** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IJsonHelper **Json** [get]
- global::Microsoft.AspNetCore.Mvc.Rendering.IHtmlHelper< TournamentPlanner.Pages.Tournament.TwoPlayerTournamentModel > **Html** [get]
- global::Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary< TournamentPlanner.Pages.Tournament.TwoPlayerTournamentModel > **ViewData** [get]
- TournamentPlanner.Pages.Tournament.TwoPlayerTournamentModel **Model** [get]

7.93.1 Property Documentation

7.93.1.1 ModelExpressionProvider

global.Microsoft.AspNetCore.Mvc.ViewFeatures.IModelExpressionProvider TournamentPlanner.↔
Pages.Tournament.Pages_Tournament_TwoPlayerTournament.ModelExpressionProvider [get]

Initial value:

```
<div class="content">
  <div class="content-section">
    <div class="content-section-header">
      <h4>Round Robin Tournament</h4>
    </div>
    <div class="content-section-body">
      Etiam commodo dui eget wisi. Vivamus luctus egestas leo. Nullam sit amet magna in magna gravida
      vehicula.
      Sed vel lectus. Donec odio tempus molestie, porttitor ut, iaculis quis, sem.
      Aliquam ante. In laoreet, magna id viverra tincidunt, sem odio bibendum justo, vel imperdiet
      sapien wisi sed libero.
      Vivamus ac leo pretium faucibus. Fusce nibh. Praesent in mauris eu tortor porttitor accumsan.
      Integer malesuada.
    </div>
  </div>
  <div class="content-section">
    <div class="content-section-header">
      <h4>Construct Tournament</h4>
    </div>
    <div class="content-section-body">
      <div class="row">
        <div class="col-md-4">
          "
```

The documentation for this class was generated from the following file:

- Release/net5.0/Razor/Pages/Tournament/TwoPlayerTournament.cshtml.g.cs

7.94 TournamentPlanner.Model.Testing.PairwiseTestCases Class Reference

A class representing the pairwise test cases created using orthogonal arrays a some post processing methods.

Public Member Functions

- [PairwiseTestCases](#) (List< string > variableNames, Dictionary< int, List< string >> variableValues, List< List< int >> testCases)
Initializes a new instance of the [PairwiseTestCases](#) class.
- List< List< string >> [TranslatedTestCases](#) ()
Test cases as list of strings.

Properties

- List< string > [VariableNames](#) [get]
Gets the list of variable names.
- Dictionary< int, List< string >> [VariableValues](#) [get]
Gets the dictionary of variable values. The key is the index of the variable name in its list.
- List< List< int >> [TestCases](#) [get]
Gets the test cases.

7.94.1 Detailed Description

A class representing the pairwise test cases created using orthogonal arrays a some post processing methods.

7.94.2 Constructor & Destructor Documentation

7.94.2.1 PairwiseTestCases()

```
TournamentPlanner.Model.Testing.PairwiseTestCases.PairwiseTestCases (
    List< string > variableNames,
    Dictionary< int, List< string >> variableValues,
    List< List< int >> testCases )
```

Initializes a new instance of the [PairwiseTestCases](#) class.

Parameters

<i>variableNames</i>	Variable names.
<i>variableValues</i>	Variable values.
<i>testCases</i>	Test cases.

7.94.3 Member Function Documentation

7.94.3.1 TranslatedTestCases()

```
List<List<string> > TournamentPlanner.Model.Testing.PairwiseTestCases.TranslatedTestCases ( )
```

Test cases as list of strings.

Returns

Translated test cases.

7.94.4 Property Documentation

7.94.4.1 TestCases

```
List<List<int> > TournamentPlanner.Model.Testing.PairwiseTestCases.TestCases [get]
```

Gets the test cases.

7.94.4.2 VariableNames

```
List<string> TournamentPlanner.Model.Testing.PairwiseTestCases.VariableNames [get]
```

Gets the list of variable names.

7.94.4.3 VariableValues

```
Dictionary<int, List<string> > TournamentPlanner.Model.Testing.PairwiseTestCases.Variable↵  
Values [get]
```

Gets the dictionary of variable values. The key is the index of the variable name in its list.

The documentation for this class was generated from the following file:

- PairwiseTestCases.cs

7.95 TournamentPlanner.Algorithms.Testing.PairwiseTestCases↵ Construction Class Reference

Class implementing the construction of the pairwise test cases using orthogonal arrays.

Public Member Functions

- [PairwiseTestCasesConstruction](#) (List< string > variableNames, Dictionary< int, List< string >> variable↵
Values)
Initializes a new instance of the [PairwiseTestCasesConstruction](#) class.
- [PairwiseTestCases ConstructPairwiseTestCases](#) ()
Construct the pairwise test cases.

7.95.1 Detailed Description

Class implementing the construction of the pairwise test cases using orthogonal arrays.

7.95.2 Constructor & Destructor Documentation

7.95.2.1 PairwiseTestCasesConstruction()

```
TournamentPlanner.Algorithms.Testing.PairwiseTestCasesConstruction.PairwiseTestCasesConstruction  
(  
    List< string > variableNames,  
    Dictionary< int, List< string >> variableValues )
```

Initializes a new instance of the [PairwiseTestCasesConstruction](#) class.

Parameters

<i>variableNames</i>	Variable names.
<i>variableValues</i>	Variable values.

7.95.3 Member Function Documentation

7.95.3.1 ConstructPairwiseTestCases()

`PairwiseTestCases` TournamentPlanner.Algorithms.Testing.PairwiseTestCasesConstruction.Construct↔
`PairwiseTestCases` ()

Construct the pairwise test cases.

Returns

Constructed test cases.

The documentation for this class was generated from the following file:

- PairwiseTestCasesConstruction.cs

7.96 TournamentPlannerTests.Algorithms.Testing.PairwiseTestCases↔ ConstructionTests Class Reference

Public Member Functions

- void `TariffFairsPairwiseTestsConstructionTest` ()

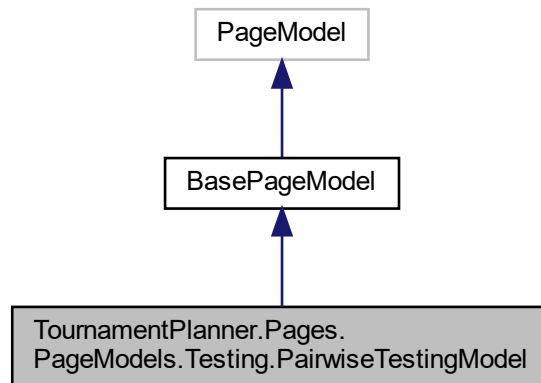
The documentation for this class was generated from the following file:

- PairwiseTestCasesConstructionTests.cs

7.97 TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel Class Reference

Page model for the construction of pairwise tests.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel:



Public Member Functions

- [PairwiseTestingModel \(\)](#)
Initializes a new instance of the [PairwiseTestingModel](#) class.
- void [OnPost](#) (IFormCollection data)

Properties

- Dictionary< string, List< string > > [VariableValues](#) [get]
Gets the variable values.
- [PairwiseTestCases TestCases](#) [get]
Gets the constructed test cases.

7.97.1 Detailed Description

Page model for the construction of pairwise tests.

7.97.2 Constructor & Destructor Documentation

7.97.2.1 PairwiseTestingModel()

`TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel.PairwiseTestingModel ()`

Initializes a new instance of the [PairwiseTestingModel](#) class.

7.97.3 Property Documentation

7.97.3.1 TestCases

[PairwiseTestCases](#) `TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel.TestCases`
[get]

Gets the constructed test cases.

7.97.3.2 VariableValues

`Dictionary<string, List<string> > TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel.VariableValues` [get]

Gets the variable values.

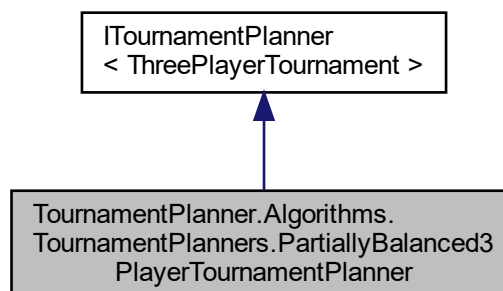
The documentation for this class was generated from the following file:

- `PairwiseTesting.cshtml.cs`

7.98 TournamentPlanner.Algorithms.TournamentPlanners.PartiallyBalanced3PlayerTournamentPlanner Class Reference

This class facilitates the construction of partially balanced three player tournaments. The number of players in the tournament has to be divisible by 3. For the construction of the tournament, GDD are used.

Inheritance diagram for `TournamentPlanner.Algorithms.TournamentPlanners.PartiallyBalanced3PlayerTournamentPlanner`:



Public Member Functions

- [PartiallyBalanced3PlayerTournamentPlanner](#) (int v)
Initializes a new instance of the [PartiallyBalanced3PlayerTournamentPlanner](#) class.
- [ThreePlayerTournament PlanTournament](#) ()

7.98.1 Detailed Description

This class facilitates the construction of partially balanced three player tournaments. The number of players in the tournament has to be divisible by 3. For the construction of the tournament, GDD are used.

7.98.2 Constructor & Destructor Documentation

7.98.2.1 PartiallyBalanced3PlayerTournamentPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.PartiallyBalanced3PlayerTournamentPlanner.↔
PartiallyBalanced3PlayerTournamentPlanner (
    int v )
```

Initializes a new instance of the [PartiallyBalanced3PlayerTournamentPlanner](#) class.

Parameters

v	Number of players.
---	--------------------

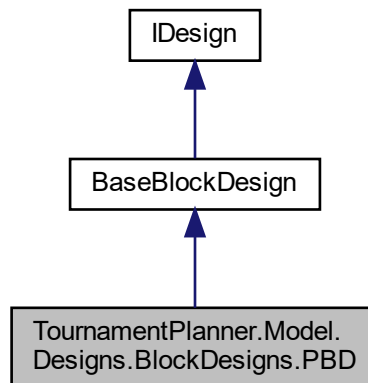
The documentation for this class was generated from the following file:

- [PartiallyBalanced3PlayerTournamentPlanner.cs](#)

7.99 TournamentPlanner.Model.Designs.BlockDesigns.PBD Class Reference

Class representing a Pairwise balanced design. [PBD](#) is similar to [BIBD](#), but the block sizes are from K - they don't have to be uniform. We only consider PBDs with lambda = 1.

Inheritance diagram for TournamentPlanner.Model.Designs.BlockDesigns.PBD:



Public Member Functions

- [PBD](#) (int v, List< int > k)
Initializes a new instance of the [PBD](#) class.
- [PBD](#) (int v, List< int > k, List< List< int >> blocks)
Initializes a new instance of the [PBD](#) class.
- [PBD](#) (int v, List< List< int >> blocks)
Initializes a new instance of the [PBD](#) class.
- void [AddBlock](#) (List< int > block)
Add a block to the list of blocks.

Properties

- List< int > [K](#) [get.]
Gets the block sizes of the [PBD](#).

Additional Inherited Members

7.99.1 Detailed Description

Class representing a Pairwise balanced design. [PBD](#) is similar to [BIBD](#), but the block sizes are from K - they don't have to be uniform. We only consider PBDs with lambda = 1.

7.99.2 Constructor & Destructor Documentation

7.99.2.1 PBD() [1/3]

```
TournamentPlanner.Model.Designs.BlockDesigns.PBD.PBD (
    int v,
    List< int > k )
```

Initializes a new instance of the [PBD](#) class.

Parameters

<i>v</i>	The order of the PBD .
<i>k</i>	The block sizes of the PBD .

7.99.2.2 PBD() [2/3]

```
TournamentPlanner.Model.Designs.BlockDesigns.PBD.PBD (
    int v,
    List< int > k,
    List< List< int >> blocks )
```

Initializes a new instance of the [PBD](#) class.

Parameters

<i>v</i>	The order of the PBD .
<i>k</i>	The block sizes of the PBD .
<i>blocks</i>	The blocks of the PBD .

7.99.2.3 PBD() [3/3]

```
TournamentPlanner.Model.Designs.BlockDesigns.PBD.PBD (
    int v,
    List< List< int >> blocks )
```

Initializes a new instance of the [PBD](#) class.

Parameters

<i>v</i>	The order of the PBD .
<i>blocks</i>	The blocks of the PBD .

7.99.3 Member Function Documentation

7.99.3.1 AddBlock()

```
void TournamentPlanner.Model.Designs.BlockDesigns.PBD.AddBlock (
    List< int > block )
```

Add a block to the list of blocks.

Parameters

<i>block</i>	The block to add.
--------------	-------------------

7.99.4 Property Documentation

7.99.4.1 K

```
List<int> TournamentPlanner.Model.Designs.BlockDesigns.PBD.K [get]
```

Gets the block sizes of the [PBD](#).

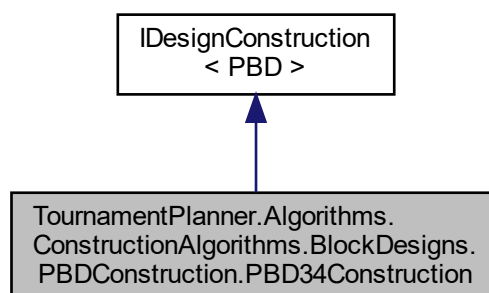
The documentation for this class was generated from the following file:

- PBD.cs

7.100 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block↔ Designs.PBDConstruction.PBD34Construction Class Reference

The construction of PBD 34 was supposed to be in the [SmallOrderPBDConstruction](#), but was a bit too complicated so i extracted it to its custom class. For the construction of PBD34 a PBD27 needs to be constructed first. Then it is modified and new blocks are added to get a PBD34.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.↔
PBD34Construction:



Public Member Functions

- [PBD ConstructDesign \(\)](#)
Construct PBD of order 34.

7.100.1 Detailed Description

The construction of PBD 34 was supposed to be in the [SmallOrderPBDConstruction](#), but was a bit too complicated so i extracted it to its custom class. For the construction of PBD34 a PBD27 needs to be constructed first. Then it is modified and new blocks are added to get a PBD34.

7.100.2 Member Function Documentation

7.100.2.1 ConstructDesign()

`PBD TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.PBD34↔
Construction.ConstructDesign ()`

Construct PBD of order 34.

Returns

Constructed PBD.

The documentation for this class was generated from the following file:

- PBD34Construction.cs

7.101 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ BlockDesigns.PBDConstruction.PBD34ConstructionTest Class Reference

Public Member Functions

- void `TestPBD34Construction ()`

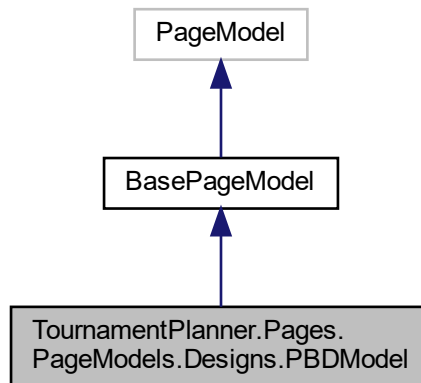
The documentation for this class was generated from the following file:

- PBD34ConstructionTest.cs

7.102 TournamentPlanner.Pages.PageModels.Designs.PBDModel Class Reference

Page model for the pairwise balanced design. Construction of PBD of orders $3M + 1$ is implemented.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Designs.PBDModel:



Public Member Functions

- void [OnPost](#) ()

Properties

- int [Multiplier](#) = 4 [get, set]
Gets or sets the multiplier.
- int [V](#) = 13 [get, set]
Gets or sets the order of the PBD to construct.
- [PBD PBD](#) [get]
Gets the constructed PBD.

7.102.1 Detailed Description

Page model for the pairwise balanced design. Construction of PBD of orders $3M + 1$ is implemented.

7.102.2 Property Documentation

7.102.2.1 Multiplier

```
int TournamentPlanner.Pages.PageModels.Designs.PBDModel.Multiplier = 4 [get], [set]
```

Gets or sets the multiplier.

7.102.2.2 PBD

```
PBD TournamentPlanner.Pages.PageModels.Designs.PBDModel.PBD [get]
```

Gets the constructed PBD.

7.102.2.3 V

```
int TournamentPlanner.Pages.PageModels.Designs.PBDModel.V = 13 [get], [set]
```

Gets or sets the order of the PBD to construct.

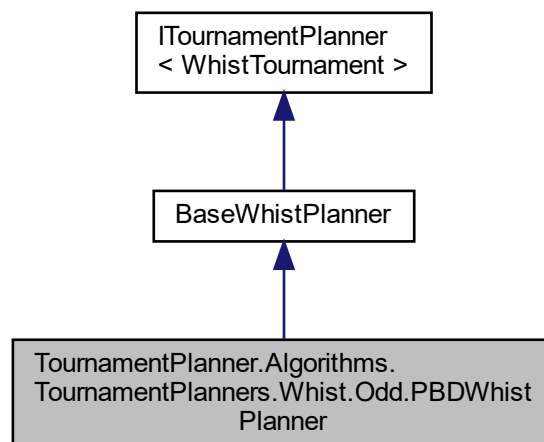
The documentation for this class was generated from the following file:

- PBD.cshtml.cs

7.103 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlanner Class Reference

Class facilitating the construction of Whist(21) and Whist(69) using PBDs.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlanner:



Public Member Functions

- [PBDWhistPlanner](#) (int v)
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.103.1 Detailed Description

Class facilitating the construction of Whist(21) and Whist(69) using PBDs.

The documentation for this class was generated from the following file:

- PBDWhistPlanner.cs

7.104 TournamentPlannerTests.Algorithms.TournamentPlanners.↔ Whist.Odd.PBDWhistPlannerTests Class Reference

Tests for the [PBDWhistPlannerTests](#).

Public Member Functions

- void [Whist21ConstructionTest](#) ()
Test the construction of whist 21.
- void [Whist69ConstructionTest](#) ()

7.104.1 Detailed Description

Tests for the [PBDWhistPlannerTests](#).

7.104.2 Member Function Documentation

7.104.2.1 Whist21ConstructionTest()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlannerTests.↔  
Whist21ConstructionTest ( )
```

Test the construction of whist 21.

The documentation for this class was generated from the following file:

- PBDWhistPlannerTests.cs

7.105 TournamentPlannerTests.Utils.PrimeFactorizationTests Class Reference

Tests for the PrimeFactorization static class.

Public Member Functions

- void [IsPrimeTest](#) ()
Test the recognition of primes.
- void [IsPrimeOrPrimePowerTest](#) ()
Test the recognition of prime powers.

7.105.1 Detailed Description

Tests for the PrimeFactorization static class.

7.105.2 Member Function Documentation

7.105.2.1 IsPrimeOrPrimePowerTest()

```
void TournamentPlannerTests.Utils.PrimeFactorizationTests.IsPrimeOrPrimePowerTest ( )
```

Test the recognition of prime powers.

7.105.2.2 IsPrimeTest()

```
void TournamentPlannerTests.Utils.PrimeFactorizationTests.IsPrimeTest ( )
```

Test the recognition of primes.

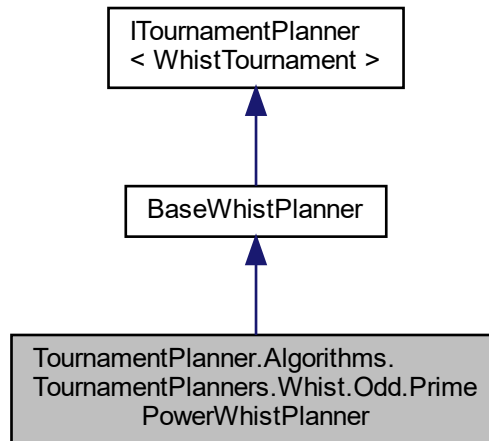
The documentation for this class was generated from the following file:

- PrimeFactorizationTests.cs

7.106 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.↔ PrimePowerWhistPlanner Class Reference

[Whist](#) tournament planner for prime power number of players in the form of $v = 4n + 1$. It uses finite fields for the tournament construction.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PrimePowerWhist↔Planner:



Public Member Functions

- [PrimePowerWhistPlanner](#) (int v)
Initializes a new instance of the [PrimePowerWhistPlanner](#) class.
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.106.1 Detailed Description

[Whist](#) tournament planner for prime power number of players in the form of $v = 4n + 1$. It uses finite fields for the tournament construction.

7.106.2 Constructor & Destructor Documentation

7.106.2.1 PrimePowerWhistPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.PrimePowerWhistPlanner.PrimePower↔
WhistPlanner (
    int v )
```

Initializes a new instance of the [PrimePowerWhistPlanner](#) class.

v	The order od the whist tournament.
---	------------------------------------

The documentation for this class was generated from the following file:

- PrimePowerWhistPlanner.cs

7.107 TournamentPlannerTests.Algorithms.TournamentPlanners.↔ Whist.Odd.PrimePowerWhistPlannerTests Class Reference

Tests for the PrimeNWhistPlanner.

Public Member Functions

- void **TestGFWhist5** ()
- void **TestSeveralValidInputs** ()
Test for several $V = 4N + 1$ where V is prime.

7.107.1 Detailed Description

Tests for the PrimeNWhistPlanner.

7.107.2 Member Function Documentation

7.107.2.1 TestSeveralValidInputs()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.PrimePowerWhistPlanner↔
Tests.TestSeveralValidInputs ( )
```

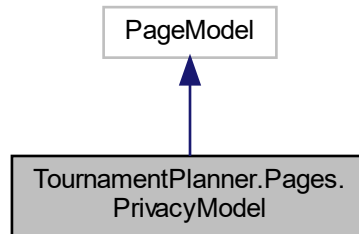
Test for several $V = 4N + 1$ where V is prime.

The documentation for this class was generated from the following file:

- PrimePowerWhistPlannerTests.cs

7.108 TournamentPlanner.Pages.PrivacyModel Class Reference

Inheritance diagram for TournamentPlanner.Pages.PrivacyModel:



Public Member Functions

- **PrivacyModel** (ILogger< [PrivacyModel](#) > logger)
- void **OnGet** ()

The documentation for this class was generated from the following file:

- Privacy.cshtml.cs

7.109 TournamentPlanner.Program Class Reference

Static Public Member Functions

- static void **Main** (string[] args)
- static IHostBuilder **CreateHostBuilder** (string[] args)

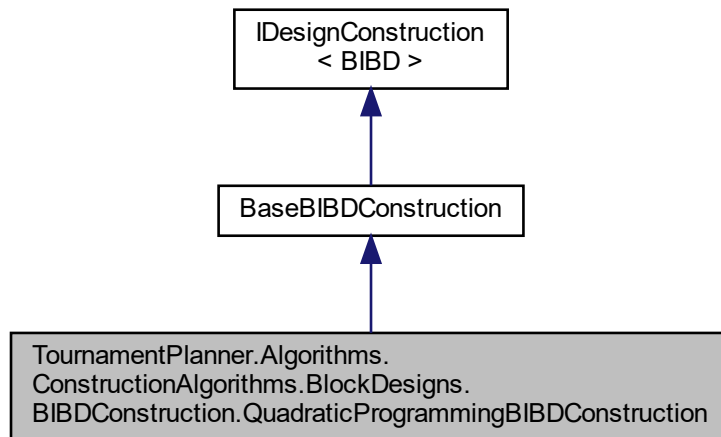
The documentation for this class was generated from the following file:

- Program.cs

7.110 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block↔ Designs.BIBDConstruction.QuadraticProgramming↔ BIBDConstruction Class Reference

A quadratic programming approach to the construction of BIBDs as described in the thesis. Uses CP-SAT as solver.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔
QuadraticProgrammingBIBDConstruction:



Public Member Functions

- [QuadraticProgrammingBIBDConstruction](#) (int v, int k, int lambda, int timeLimit=30)
- override [BIBD ConstructDesign](#) ()
Construct the BIBD.

Additional Inherited Members

7.110.1 Detailed Description

A quadratic programming approach to the construction of BIBDs as described in the thesis. Uses CP-SAT as solver.

The documentation for this class was generated from the following file:

- QuadraticProgrammingBIBDConstruction.cs

7.111 [TournamentPlannerTests.Algorithms.ConstructionAlgorithms.](#) [BlockDesigns.BIBDConstruction.QuadraticProgramming](#) **BIBDConstructionTests Class Reference**

Tests for the QuadraticProgrammingBIBDConstruction.

Public Member Functions

- void [QuadraticConstructionTest](#) ()
Simple BIBD construction test.

7.111.1 Detailed Description

Tests for the QuadraticProgrammingBIBDConstruction.

7.111.2 Member Function Documentation

7.111.2.1 QuadraticConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.  
QuadraticProgrammingBIBDConstructionTests.QuadraticConstructionTest ( )
```

Simple BIBD construction test.

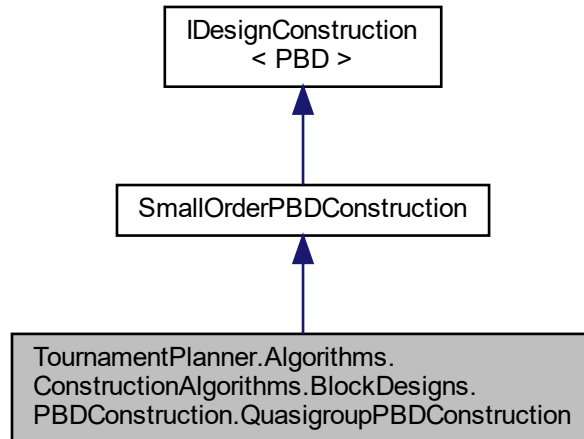
The documentation for this class was generated from the following file:

- QuadraticProgrammingBIBDConstructionTests.cs

7.112 [TournamentPlanner.Algorithms.ConstructionAlgorithms.Block](#) [Designs.PBDConstruction.QuasigroupPBDConstruction](#) Class **Reference**

Method for construction of PBDs of orders 13, 16 and 28. This method uses pairs of MOLS as quasigroups.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.↔
QuasigroupPBDConstruction:



Public Member Functions

- [QuasigroupPBDConstruction](#) (int v)
Initializes a new instance of the [QuasigroupPBDConstruction](#) class.
- new [PBD ConstructDesign](#) ()

Additional Inherited Members

7.112.1 Detailed Description

Method for construction of PBDs of orders 13, 16 and 28. This method uses pairs of MOLS as quasigroups.

7.112.2 Constructor & Destructor Documentation

7.112.2.1 QuasigroupPBDConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Quasigroup↔  
PBDConstruction.QuasigroupPBDConstruction (   
    int v )
```

Initializes a new instance of the [QuasigroupPBDConstruction](#) class.

Parameters

v	The order of PBD.
---	-------------------

The documentation for this class was generated from the following file:

- QuasigroupPBDConstruction.cs

7.113 TournamentPlannerTests.Algorithms.ConstructionAlgorithms. BlockDesigns.PBDConstruction.QuasigroupPBDConstruction Tests Class Reference

Tests for the QuasigroupPBDConstruction.

Public Member Functions

- void [QuasigroupConstructionTest](#) ()
Tests the construction of PBDs of orders 13, 16, 28.

7.113.1 Detailed Description

Tests for the QuasigroupPBDConstruction.

7.113.2 Member Function Documentation

7.113.2.1 QuasigroupConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.  
QuasigroupPBDConstructionTests.QuasigroupConstructionTest ( )
```

Tests the construction of PBDs of orders 13, 16, 28.

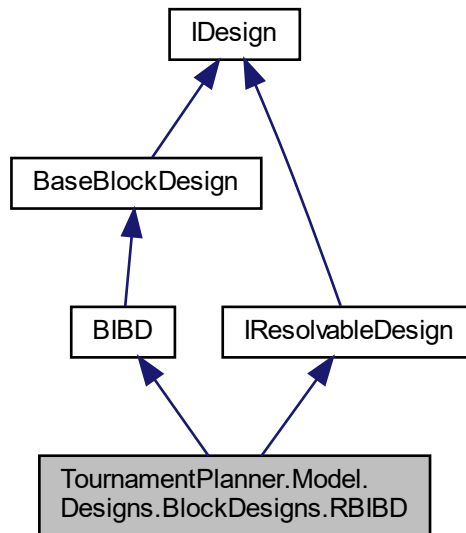
The documentation for this class was generated from the following file:

- QuasigroupPBDConstructionTests.cs

7.114 TournamentPlanner.Model.Designs.BlockDesigns.RBIBD Class Reference

Class representing a resolvable balanced incomplete blocks design.

Inheritance diagram for TournamentPlanner.Model.Designs.BlockDesigns.RBIBD:



Public Member Functions

- [RBIBD](#) (int v, int k, int lambda)
Initializes a new instance of the [RBIBD](#) class.
- void [AddParallelClass](#) (List< List< int >> parallelClass)
Add a parallel class. Also add its block to the list of blocks.

Properties

- List< List< List< int >>> [ParallelClasses](#) = new () [get, protected set]

Additional Inherited Members

7.114.1 Detailed Description

Class representing a resolvable balanced incomplete blocks design.

7.114.2 Constructor & Destructor Documentation

7.114.2.1 RBIBD()

```
TournamentPlanner.Model.Designs.BlockDesigns.RBIBD.RBIBD (
    int v,
    int k,
    int lambda )
```

Initializes a new instance of the [RBIBD](#) class.

Parameters

<i>v</i>	Order of the RBIBD .
<i>k</i>	Size of blocks.
<i>lambda</i>	Pair occurrences.

7.114.3 Member Function Documentation

7.114.3.1 AddParallelClass()

```
void TournamentPlanner.Model.Designs.BlockDesigns.RBIBD.AddParallelClass (
    List< List< int >> parallelClass )
```

Add a parallel class. Also add its block to the list of blocks.

Parameters

<i>parallelClass</i>	The parallel class to add.
----------------------	----------------------------

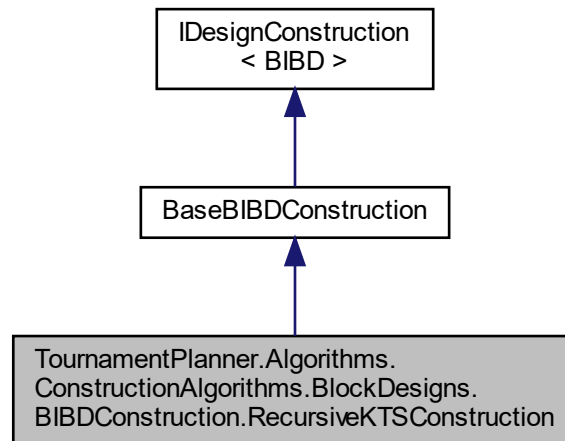
The documentation for this class was generated from the following file:

- RBIBD.cs

7.115 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block↔ Designs.BIBDConstruction.RecursiveKTSConstruction Class Reference

Algorithm for the recursive construction of KTS of order $6N + 3$. Method described in <https://books.google.com/books?id=OoRFisqn-aIC>. KTSs of orders 9, 15, 21 and 39 are constructed statically, all the others are constructed in a recursive manner.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔
RecursiveKTSConstruction:



Public Member Functions

- [RecursiveKTSConstruction](#) (int v)
Initializes a new instance of the [RecursiveKTSConstruction](#) class.
- override [KTS ConstructDesign](#) ()
Construct a KTS.
- List< List< List< Tuple< int, int > > > > [ConstructBlockInducedParallelClasses](#) (List< int > block, [KTS](#) kts)
Construct block induced parallel classes of tuples (before translation).

Additional Inherited Members

7.115.1 Detailed Description

Algorithm for the recursive construction of KTS of order $6N + 3$. Method described in <https://books.google.com/books?id=OoRFisqn-aIC>. KTSs of orders 9, 15, 21 and 39 are constructed statically, all the others are constructed in a recursive manner.

7.115.2 Constructor & Destructor Documentation

7.115.2.1 RecursiveKTSConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Recursive↔  
KTSConstruction.RecursiveKTSConstruction (  
    int v )
```

Initializes a new instance of the [RecursiveKTSConstruction](#) class.

Parameters

v	Set size.
---	-----------

7.115.3 Member Function Documentation

7.115.3.1 ConstructBlockInducedParallelClasses()

```
List<List<List<Tuple<int, int> > > > TournamentPlanner.Algorithms.ConstructionAlgorithms.↔
BlockDesigns.BIBDConstruction.RecursiveKTSConstruction.ConstructBlockInducedParallelClasses (
    List< int > block,
    KTS kts )
```

Construct block induced parallel classes of tuples (before translation).

Parameters

<i>block</i>	The block of PBD.
<i>kts</i>	The initial KTS.

Returns

Parallel classes of tuples.

7.115.3.2 ConstructDesign()

```
override KTS TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔
RecursiveKTSConstruction.ConstructDesign ( ) [virtual]
```

Construct a KTS.

Returns

Constructed KTS.

Implements [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction](#).

The documentation for this class was generated from the following file:

- RecursiveKTSConstruction.cs

7.116 TournamentPlannerTests.Algorithms.ConstructionAlgorithms. BlockDesigns.BIBDConstruction.RecursiveKTSConstructionTests Class Reference

Tests for the RecursiveKTSConstruction.

Public Member Functions

- void **KTS357ConstructionTest** ()
- void [RecursiveKTSConstructionTest](#) ()
Test the construction of KTS of orders $6N + 3$.

7.116.1 Detailed Description

Tests for the RecursiveKTSConstruction.

7.116.2 Member Function Documentation

7.116.2.1 RecursiveKTSConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.  
RecursiveKTSConstructionTests.RecursiveKTSConstructionTest ( )
```

Test the construction of KTS of orders $6N + 3$.

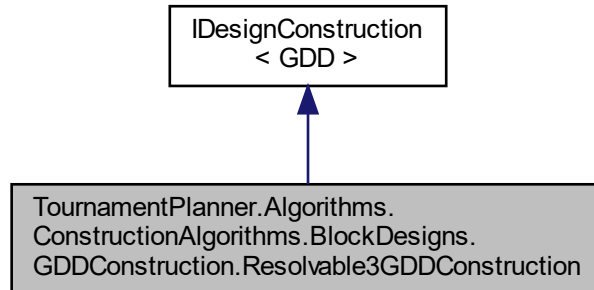
The documentation for this class was generated from the following file:

- RecursiveKTSConstructionTests.cs

7.117 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block Designs.GDDConstruction.Resolvable3GDDConstruction Class Reference

Construct resolvable group divisible designs with block size 3 using MOLS. Should be able to construct GDDs where the order is a multiple of 3 except 6 and 18.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Resolvable3GDDConstruction:



Public Member Functions

- [Resolvable3GDDConstruction](#) (int v)
Initializes a new instance of the [Resolvable3GDDConstruction](#) class.
- [GDD ConstructDesign](#) ()
Construct a GDD with block size 3.

7.117.1 Detailed Description

Construct resolvable group divisible designs with block size 3 using MOLS. Should be able to construct GDDs where the order is a multiple of 3 except 6 and 18.

7.117.2 Constructor & Destructor Documentation

7.117.2.1 Resolvable3GDDConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Resolvable3GDDConstruction.Resolvable3GDDConstruction (
    int v )
```

Initializes a new instance of the [Resolvable3GDDConstruction](#) class.

Parameters

v	Number of elements in GDD.
---	----------------------------

7.117.3 Member Function Documentation

7.117.3.1 ConstructDesign()

`GDD TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Resolvable3GDDConstruction.ConstructDesign ()`

Construct a GDD with block size 3.

Returns

Group divisible design.

The documentation for this class was generated from the following file:

- Resolvable3GDDConstruction.cs

7.118 TournamentPlannerTests.Algorithms.ConstructionAlgorithms. BlockDesigns.GDDConstruction.Resolvable3GDDConstruction Tests Class Reference

Tests the creation of resolvable GDDs by Resolvable3GDDConstruction.

Public Member Functions

- void `GDDConstructionTest ()`
Test the construction, the subsequent validity and the parallel classes of GDDs of several orders. Valid orders are

7.118.1 Detailed Description

Tests the creation of resolvable GDDs by Resolvable3GDDConstruction.

7.118.2 Member Function Documentation

7.118.2.1 GDDConstructionTest()

`void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Resolvable3GDDConstructionTests.GDDConstructionTest ()`

Test the construction, the subsequent validity and the parallel classes of GDDs of several orders. Valid orders are

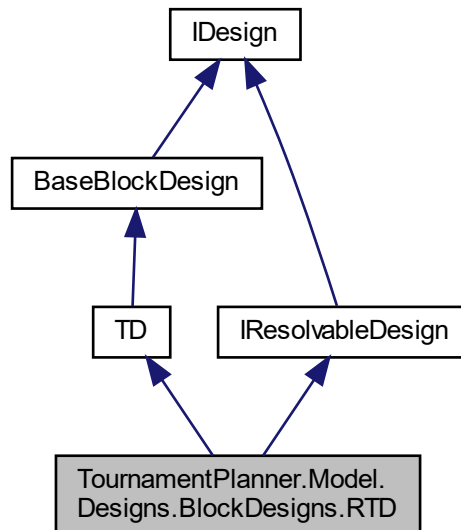
The documentation for this class was generated from the following file:

- Resolvable3GDDConstructionTests.cs

7.119 TournamentPlanner.Model.Designs.BlockDesigns.RTD Class Reference

Resolvable transversal design.

Inheritance diagram for TournamentPlanner.Model.Designs.BlockDesigns.RTD:



Public Member Functions

- [RTD](#) (int k, int m, List< List< int >> groups)
Initializes a new instance of the [RTD](#) class.
- void [AddParallelClass](#) (List< List< int >> parallelClass)
Add parallel class and its blocks.

Properties

- List< List< List< int >> > [ParallelClasses](#) = new () [get]

Additional Inherited Members

7.119.1 Detailed Description

Resolvable transversal design.

7.119.2 Constructor & Destructor Documentation**7.119.2.1 RTD()**

```
TournamentPlanner.Model.Designs.BlockDesigns.RTD.RTD (
    int k,
    int m,
    List< List< int >> groups )
```

Initializes a new instance of the [RTD](#) class.

Parameters

<i>k</i>	Block sizes.
<i>m</i>	Group sizes.
<i>groups</i>	Groups of the design.

7.119.3 Member Function Documentation**7.119.3.1 AddParallelClass()**

```
void TournamentPlanner.Model.Designs.BlockDesigns.RTD.AddParallelClass (
    List< List< int >> parallelClass )
```

Add parallel class and its blocks.

Parameters

<i>parallelClass</i>	The parallel class to add.
----------------------	----------------------------

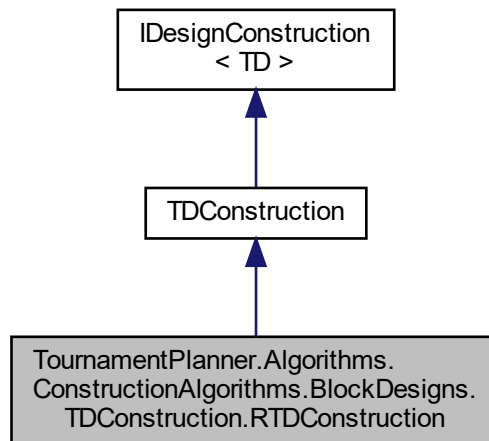
The documentation for this class was generated from the following file:

- RTD.cs

7.120 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.RTDConstruction Class Reference

Construction method for resolvable transversal designs.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.↔
RTDConstruction:



Public Member Functions

- [RTDConstruction](#) (int *k*, int *m*)
Initializes a new instance of the [RTDConstruction](#) class.
- new [RTD ConstructDesign](#) ()
Construct the resolvable transversal design.

Additional Inherited Members

7.120.1 Detailed Description

Construction method for resolvable transversal designs.

7.120.2 Constructor & Destructor Documentation

7.120.2.1 RTDConstruction()

```

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.RTDConstruction.↔
RTDConstruction (
    int k,
    int m )
  
```

Initializes a new instance of the [RTDConstruction](#) class.

Parameters

k	Block sizes.
m	Group sizes.

7.120.3 Member Function Documentation

7.120.3.1 ConstructDesign()

```
new RTD TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.↔  
RTDConstruction.ConstructDesign ( )
```

Construct the resolvable transversal design.

Returns

Resolvable transversal design.

The documentation for this class was generated from the following file:

- RTDConstruction.cs

7.121 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ BlockDesigns.TDConstruction.RTDConstructionTests Class Reference

Tests for the RTDConstruction.

Public Member Functions

- void [RTDConstructionTest](#) ()
Test the construction of resolvable TD(18, 19).

7.121.1 Detailed Description

Tests for the RTDConstruction.

7.121.2 Member Function Documentation

7.121.2.1 RTDConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.↔
RTDConstructionTests.RTDConstructionTest ( )
```

Test the construction of resolvable TD(18, 19).

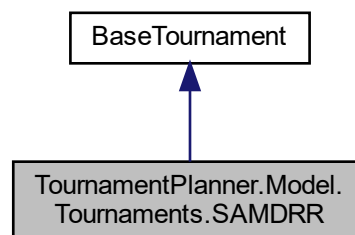
The documentation for this class was generated from the following file:

- RTDConstructionTests.cs

7.122 TournamentPlanner.Model.Tournaments.SAMDRR Class Reference

Spouse-avoiding mixed doubles round robin tournament. Games are represented as 4 integers, (i, l, j, k) - H_iW_lvH_jW_k

Inheritance diagram for TournamentPlanner.Model.Tournaments.SAMDRR:



Public Member Functions

- [SAMDRR](#) (int order, List< List< List< int >>> rounds)
Initializes a new instance of the [SAMDRR](#) class.

Additional Inherited Members

7.122.1 Detailed Description

Spouse-avoiding mixed doubles round robin tournament. Games are represented as 4 integers, (i, l, j, k) - H_iW_lvH_jW_k

7.122.2 Constructor & Destructor Documentation

7.122.2.1 SAMDRR()

```
TournamentPlanner.Model.Tournaments.SAMDRR.SAMDRR (
    int order,
    List< List< List< int >>> rounds )
```

Initializes a new instance of the [SAMDRR](#) class.

Parameters

<i>order</i>	The order.
<i>rounds</i>	The rounds.

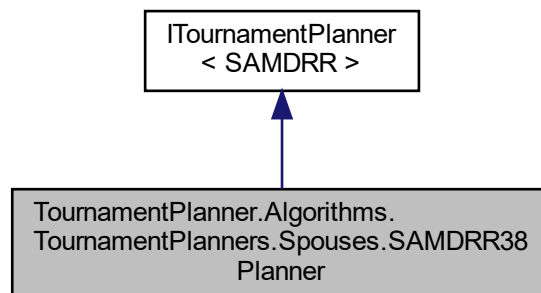
The documentation for this class was generated from the following file:

- SAMDRR.cs

7.123 TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR38Planner Class Reference

SAMDRR 38 planner. $\text{GCD}(38, 6) \neq 1$ so a special construction is required. SAMDRR 38 is used to construct Whist(152).

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR38Planner:



Public Member Functions

- [SAMDRR PlanTournament \(\)](#)
Plan a SAMDRR 38.

7.123.1 Detailed Description

SAMDRR 38 planner. $\text{GCD}(38, 6) \neq 1$ so a special construction is required. SAMDRR 38 is used to construct `Whist(152)`.

7.123.2 Member Function Documentation

7.123.2.1 `PlanTournament()`

`SAMDRR TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR38Planner.PlanTournament()`

Plan a SAMDRR 38.

Returns

SAMDRR 38.

The documentation for this class was generated from the following file:

- SAMDRR38Planner.cs

7.124 TournamentPlannerTests.Algorithms.TournamentPlanners.↔ Spouses.SAMDRR38PlannerTests Class Reference

Tests for the SAMDRR38Planner.

Public Member Functions

- void `SAMDRR38ConstructionTest()`

7.124.1 Detailed Description

Tests for the SAMDRR38Planner.

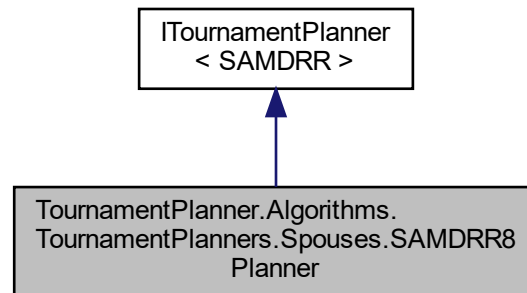
The documentation for this class was generated from the following file:

- SAMDRR38PlannerTests.cs

7.125 TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR8Planner Class Reference

SAMDRR 8 construction class.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR8Planner:



Public Member Functions

- [SAMDRR PlanTournament \(\)](#)
Plan SAMDRR 8.

7.125.1 Detailed Description

SAMDRR 8 construction class.

7.125.2 Member Function Documentation

7.125.2.1 PlanTournament()

`SAMDRR TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR8Planner.PlanTournament ()`

Plan SAMDRR 8.

Returns

SAMDRR 8.

The documentation for this class was generated from the following file:

- SAMDRR8Planner.cs

7.126 TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SAMDRR8PlannerTests Class Reference

Tests for the SAMDRR8Planner.

Public Member Functions

- void **SAMDRR8ConstructionTest** ()

7.126.1 Detailed Description

Tests for the SAMDRR8Planner.

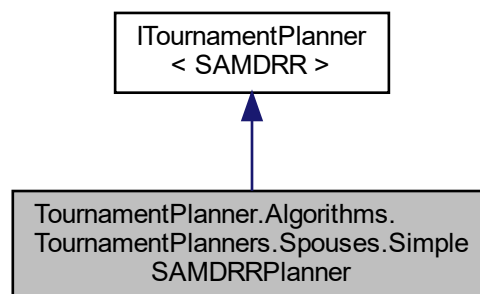
The documentation for this class was generated from the following file:

- SAMDRR8PlannerTests.cs

7.127 TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlanner Class Reference

SAMDRR planner using cyclic SOLS such that for the SOLS order n , $\text{GCD}(n, 6) = 1$.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlanner:



Public Member Functions

- [SimpleSAMDRRPlanner](#) (int order)
Initializes a new instance of the [SimpleSAMDRRPlanner](#) class.
- [SAMDRR PlanTournament](#) ()
Plan a SAMDRR.

7.127.1 Detailed Description

SAMDRR planner using cyclic SOLS such that for the SOLS order n , $\text{GCD}(n, 6) = 1$.

7.127.2 Constructor & Destructor Documentation

7.127.2.1 SimpleSAMDRRPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlanner.SimpleSAMDRRPlanner  
(  
    int order )
```

Initializes a new instance of the [SimpleSAMDRRPlanner](#) class.

Parameters

<i>order</i>	The order.
--------------	------------

7.127.3 Member Function Documentation

7.127.3.1 PlanTournament()

```
SAMDRR TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlanner.Plan↔  
Tournament ( )
```

Plan a SAMDRR.

Returns

The scheduled SAMDRR.

The documentation for this class was generated from the following file:

- SimpleSAMDRRPlanner.cs

7.128 TournamentPlannerTests.Algorithms.TournamentPlanners.↔ Spouses.SimpleSAMDRRPlannerTests Class Reference

Tests for the SimpleSAMDRRPlanner.

Public Member Functions

- void [TestSAMDRR11Construction](#) ()
Test the construction of SAMDRR 11.

7.128.1 Detailed Description

Tests for the SimpleSAMDRRPlanner.

7.128.2 Member Function Documentation

7.128.2.1 TestSAMDRR11Construction()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlannerTests.↔
TestSAMDRR11Construction ( )
```

Test the construction of SAMDRR 11.

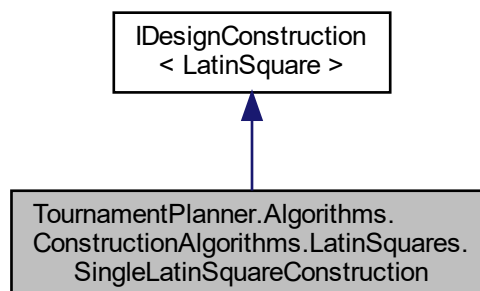
The documentation for this class was generated from the following file:

- SimpleSAMDRRPlannerTests.cs

7.129 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstruction Class Reference

Recursive construction of a single Latin square of any order.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstruction:↔



Public Member Functions

- [SingleLatinSquareConstruction](#) (int order)
Initializes a new instance of the [SingleLatinSquareConstruction](#) class.
- [LatinSquare ConstructDesign](#) ()
Construct the Latin square.

7.129.1 Detailed Description

Recursive construction of a single Latin square of any order.

7.129.2 Constructor & Destructor Documentation

7.129.2.1 SingleLatinSquareConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstruction.↔  
SingleLatinSquareConstruction (  
    int order )
```

Initializes a new instance of the [SingleLatinSquareConstruction](#) class.

Parameters

<i>order</i>	The LS order.
--------------	---------------

7.129.3 Member Function Documentation

7.129.3.1 ConstructDesign()

```
LatinSquare TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatin↔  
SquareConstruction.ConstructDesign ( )
```

Construct the Latin square.

Returns

Constructed Latin square.

The documentation for this class was generated from the following file:

- [SingleLatinSquareConstruction.cs](#)

7.130 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstructionTests Class Reference

Tests for the SingleLatinSquareConstruction.

Public Member Functions

- void [SingleLatinSquareOfOrder10Test](#) ()
Test the creation and validate a Latin square of order 10.
- void [SeveralOrdersTest](#) ()
Test Latin squares or orders 2 to 10.

7.130.1 Detailed Description

Tests for the SingleLatinSquareConstruction.

7.130.2 Member Function Documentation

7.130.2.1 SeveralOrdersTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstructionTests.SeveralOrdersTest ( )
```

Test Latin squares or orders 2 to 10.

7.130.2.2 SingleLatinSquareOfOrder10Test()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstructionTests.SingleLatinSquareOfOrder10Test ( )
```

Test the creation and validate a Latin square of order 10.

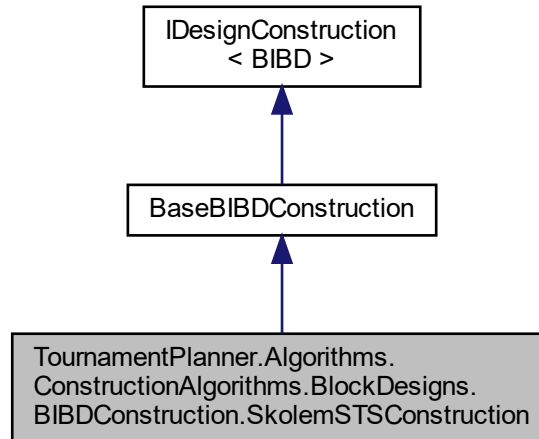
The documentation for this class was generated from the following file:

- SingleLatinSquareConstructionTests.cs

7.131 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SkolemSTSTSCConstruction Class Reference

Skolem construction algorithm for STS of size $6N + 1$. <http://math.ucdenver.edu/~wcherowi/courses/m6406/sts.pdf>

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SkolemSTSTSCConstruction:



Public Member Functions

- [SkolemSTSTSCConstruction](#) (int v)
Initializes a new instance of the [SkolemSTSTSCConstruction](#) class.
- override [STS ConstructDesign](#) ()
Construct STS of size $6N + 1$.

Additional Inherited Members

7.131.1 Detailed Description

Skolem construction algorithm for STS of size $6N + 1$. <http://math.ucdenver.edu/~wcherowi/courses/m6406/sts.pdf>

7.131.2 Constructor & Destructor Documentation

7.131.2.1 SkolemSTSTSCConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SkolemSTSTSCConstruction.SkolemSTSTSCConstruction (
    int v )
```

Initializes a new instance of the [SkolemSTSTSCConstruction](#) class.

Parameters

v	Set size.
---	-----------

7.131.3 Member Function Documentation

7.131.3.1 ConstructDesign()

```
override STS TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔
SkolemSTSTConstruction.ConstructDesign ( ) [virtual]
```

Construct STS of size $6N + 1$.

Returns

The constructed STS.

Implements [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction](#).

The documentation for this class was generated from the following file:

- SkolemSTSTConstruction.cs

7.132 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ BlockDesigns.BIBDConstruction.SkolemSTSTConstructionTests Class Reference

Tests for the SkolemSTSTConstruction.

Public Member Functions

- void [SkolemSTSTConstructionTest](#) ()
Test the creation of a few STS of order $6N + 1$.
- void [SkolemSTSTExceptionTest](#) ()
Test the exception if the order is not valid.

7.132.1 Detailed Description

Tests for the SkolemSTSTConstruction.

7.132.2 Member Function Documentation

7.132.2.1 SkolemSTSConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.  
SkolemSTSConstructionTests.SkolemSTSConstructionTest ( )
```

Test the creation of a few STS of order $6N + 1$.

7.132.2.2 SkolemSTSExceptionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.  
SkolemSTSConstructionTests.SkolemSTSExceptionTest ( )
```

Test the exception if the order is not valid.

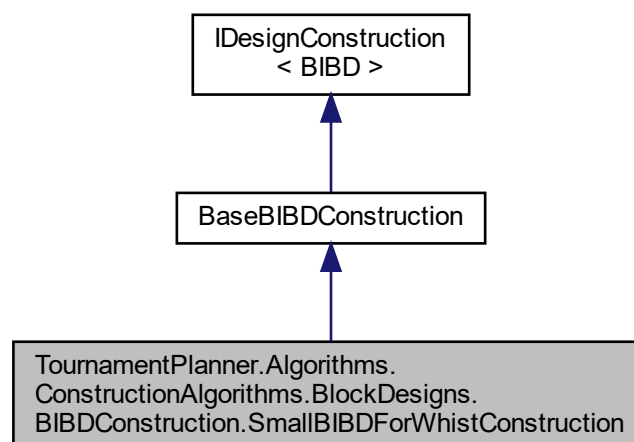
The documentation for this class was generated from the following file:

- SkolemSTSConstructionTests.cs

7.133 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmallBIBDForWhistConstruction Class Reference

Static construction of BIBD(21, 5, 1) and BIBD(25, 5, 1). These designs have been constructed by the incremental / quadratic methods. Those methods are, however, unreliable (license and memory issues) so the constructed designs are stored here for faster and more reliable access.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmallBIBDForWhistConstruction:



Public Member Functions

- [SmallBIBDForWhistConstruction](#) (int v)
Initializes a new instance of the [SmallBIBDForWhistConstruction](#) class.
- override [BIBD ConstructDesign](#) ()
Construct the BIBD.

Additional Inherited Members

7.133.1 Detailed Description

Static construction of BIBD(21, 5, 1) and BIBD(25, 5, 1). These designs have been constructed by the incremental / quadratic methods. Those methods are, however, unreliable (license and memory issues) so the constructed designs are stored here for faster and more reliable access.

7.133.2 Constructor & Destructor Documentation

7.133.2.1 SmallBIBDForWhistConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.Small↔
BIBDForWhistConstruction.SmallBIBDForWhistConstruction (
    int v )
```

Initializes a new instance of the [SmallBIBDForWhistConstruction](#) class.

Parameters

v	BIBD order.
---	-------------

The documentation for this class was generated from the following file:

- [SmallBIBDForWhistConstruction.cs](#)

7.134 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ BlockDesigns.BIBDConstruction.SmallBIBDForWhist↔ ConstructionTests Class Reference

Tests for SmallBIBDForWhistConstruction.

Public Member Functions

- void [BIBDConstructionTest](#) ()

7.134.1 Detailed Description

Tests for SmallBIBDForWhistConstruction.

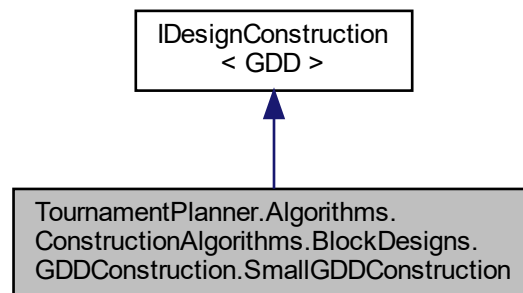
The documentation for this class was generated from the following file:

- SmallBIBDForWhistConstructionTests.cs

7.135 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block↔ Designs.GDDConstruction.SmallGDDConstruction Class Reference

Construct a few small order GDDs which are then used i Wilsons fundamental construction.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.↔
SmallGDDConstruction:



Public Member Functions

- [SmallGDDConstruction](#) (int v)
Initializes a new instance of the [SmallGDDConstruction](#) class.
- [GDD ConstructDesign](#) ()
Construct a GDD of small order.

7.135.1 Detailed Description

Construct a few small order GDDs which are then used i Wilsons fundamental construction.

7.135.2 Constructor & Destructor Documentation

7.135.2.1 SmallGDDConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.SmallGDDConstruction.↔
SmallGDDConstruction (
    int v )
```

Initializes a new instance of the [SmallGDDConstruction](#) class.

Parameters

v	Size of GDD.
---	--------------

7.135.3 Member Function Documentation

7.135.3.1 ConstructDesign()

```
GDD TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Small↔
GDDConstruction.ConstructDesign ( )
```

Construct a GDD of small order.

Returns

Constructed GDD.

The documentation for this class was generated from the following file:

- [SmallGDDConstruction.cs](#)

7.136 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ BlockDesigns.GDDConstruction.SmallGDDConstructionTests Class Reference

Tests for the SmallGDDConstruction.

Public Member Functions

- void [TestSmallGDDConstructionForAvailableOrders](#) ()
Test the small GDD construction for all the available orders.

7.136.1 Detailed Description

Tests for the SmallGDDConstruction.

7.136.2 Member Function Documentation

7.136.2.1 TestSmallGDDConstructionForAvailableOrders()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.SmallGDDConstructionTests.TestSmallGDDConstructionForAvailableOrders ( )
```

Test the small GDD construction for all the available orders.

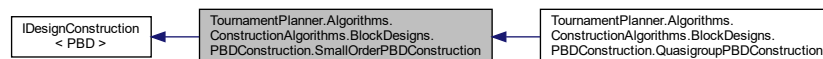
The documentation for this class was generated from the following file:

- SmallGDDConstructionTests.cs

7.137 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.SmallOrderPBDConstruction Class Reference

A class facilitating the construction of PBDs of smaller orders in form $3N + 1$. All the constructed design have block sizes in $\{4, 7, 10, 19\}$. There are several construction methods, not a single unified one.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.SmallOrderPBDConstruction:



Public Member Functions

- [SmallOrderPBDConstruction](#) (int v)
Initializes a new instance of the [SmallOrderPBDConstruction](#) class.
- [PBD ConstructDesign](#) ()
Construct a PBD.

Properties

- int [V](#) [get]
Gets the order of the PBD to construct $V = 3N + 1$.

7.137.1 Detailed Description

A class facilitating the construction of PBDs of smaller orders in form $3N + 1$. All the constructed design have block sizes in $\{4, 7, 10, 19\}$. There are several construction methods, not a single unified one.

7.137.2 Constructor & Destructor Documentation

7.137.2.1 SmallOrderPBDConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.SmallOrder↔  
PBDConstruction.SmallOrderPBDConstruction (   
    int v )
```

Initializes a new instance of the [SmallOrderPBDConstruction](#) class.

Parameters

v	The order of the PBD.
---	-----------------------

7.137.3 Member Function Documentation

7.137.3.1 ConstructDesign()

```
PBD TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Small↔  
OrderPBDConstruction.ConstructDesign ( )
```

Construct a PBD.

Returns

Constructed PBD.

7.137.4 Property Documentation

7.137.4.1 V

```
int TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.Small↔  
OrderPBDConstruction.V [get], [protected]
```

Gets the order of the PBD to construct $V = 3N + 1$.

The documentation for this class was generated from the following file:

- SmallOrderPBDConstruction.cs

7.138 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ BlockDesigns.PBDConstruction.SmallOrderPBDConstruction.↔ Tests Class Reference

Tests for the SmallOrderPBDConstruction.

Public Member Functions

- void [SmallPBDUpToOrder46ConstructionTest](#) ()
Test the construction of PBDs up to the order 46.
- void [PBD79And82ConstructionTest](#) ()
Tests the PBD(79) and PBD(82) construction.

7.138.1 Detailed Description

Tests for the SmallOrderPBDConstruction.

7.138.2 Member Function Documentation

7.138.2.1 PBD79And82ConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.↔  
SmallOrderPBDConstructionTests.PBD79And82ConstructionTest ( )
```

Tests the PBD(79) and PBD(82) construction.

7.138.2.2 SmallPBDUpToOrder46ConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.↔  
SmallOrderPBDConstructionTests.SmallPBDUpToOrder46ConstructionTest ( )
```

Test the construction of PBDs up to the order 46.

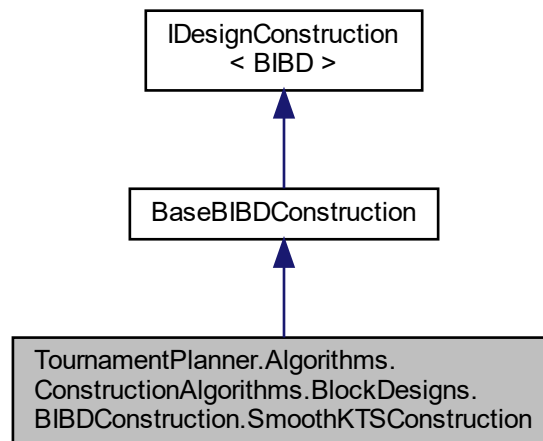
The documentation for this class was generated from the following file:

- SmallOrderPBDConstructionTests.cs

7.139 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmoothKTSConstruction Class Reference

Algorithm for construction of smooth Kirkman triple systems. The size of smooth KTS is a power of 3. <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.501.8928&rep=rep1&type=pdf>

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmoothKTSConstruction:



Public Member Functions

- `SmoothKTSConstruction` (int v)
Initializes a new instance of the `SmoothKTSConstruction` class.
- override `STS ConstructDesign` ()
Construct a smooth KTS where the size is a power of 3.

Additional Inherited Members

7.139.1 Detailed Description

Algorithm for construction of smooth Kirkman triple systems. The size of smooth KTS is a power of 3. <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.501.8928&rep=rep1&type=pdf>

7.139.2 Constructor & Destructor Documentation

7.139.2.1 SmoothKTSConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmoothKTSConstruction.SmoothKTSConstruction (
    int v )
```

Initializes a new instance of the [SmoothKTSConstruction](#) class.

Parameters

v	Set size.
---	-----------

7.139.3 Member Function Documentation

7.139.3.1 ConstructDesign()

```
override STS TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmoothKTSConstruction.ConstructDesign ( ) [virtual]
```

Construct a smooth KTS where the size is a power of 3.

Returns

Smooth KTS.

Implements [TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction](#).

The documentation for this class was generated from the following file:

- [SmoothKTSConstruction.cs](#)

7.140 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmoothKTSConstructionTests Class Reference ↩

Tests for the SmoothKTSConstruction.

Public Member Functions

- void [SmoothKTSConstructionTest](#) ()
Tests the construction of KTS where the order is a power of 3.

7.140.1 Detailed Description

Tests for the SmoothKTSConstruction.

7.140.2 Member Function Documentation

7.140.2.1 SmoothKTSConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.↔  
SmoothKTSConstructionTests.SmoothKTSConstructionTest ( )
```

Tests the construction of KTS where the order is a power of 3.

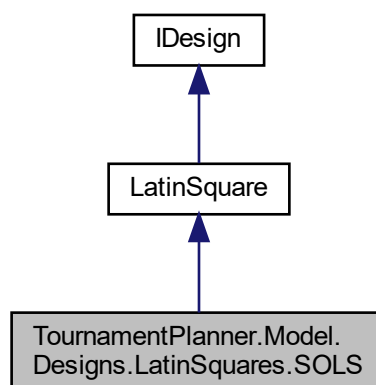
The documentation for this class was generated from the following file:

- SmoothKTSConstructionTests.cs

7.141 TournamentPlanner.Model.Designs.LatinSquares.SOLS Class Reference

Self orthogonal Latin square.

Inheritance diagram for TournamentPlanner.Model.Designs.LatinSquares.SOLS:



Public Member Functions

- [SOLS](#) (int v, int[,] ls)
Initializes a new instance of the SOLS class.
- int[,] [Transpose](#) ()
Get the transpose of the matrix.

Additional Inherited Members

7.141.1 Detailed Description

Self orthogonal Latin square.

7.141.2 Constructor & Destructor Documentation

7.141.2.1 SOLS()

```
TournamentPlanner.Model.Designs.LatinSquares.SOLS.SOLS (
    int v,
    int ls[, ] )
```

Initializes a new instance of the [SOLS](#) class.

Parameters

v	The order.
ls	The Latin square matrix.

7.141.3 Member Function Documentation

7.141.3.1 Transpose()

```
int [, ] TournamentPlanner.Model.Designs.LatinSquares.SOLS.Transpose ( )
```

Get the transpose of the matrix.

Returns

The transpose.

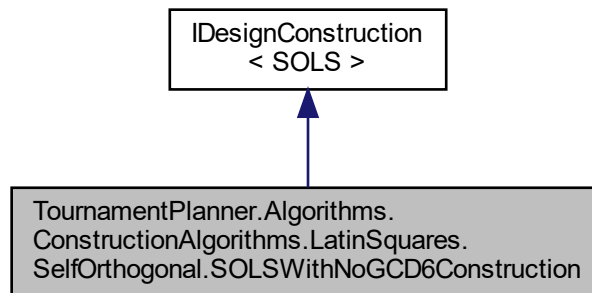
The documentation for this class was generated from the following file:

- SOLS.cs

7.142 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNoGCD6Construction Class Reference

Construct a cyclic SOLS such that $\text{GCD}(n,6) = 1$.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNoGCD6Construction:



Public Member Functions

- [SOLSWithNoGCD6Construction](#) (int order)
Initializes a new instance of the [SOLSWithNoGCD6Construction](#) class.
- [SOLS ConstructDesign](#) ()
Construct a SOLS.

7.142.1 Detailed Description

Construct a cyclic SOLS such that $\text{GCD}(n,6) = 1$.

7.142.2 Constructor & Destructor Documentation

7.142.2.1 SOLSWithNoGCD6Construction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNoGCD6Construction.SOLSWithNoGCD6Construction (
    int order )
```

Initializes a new instance of the [SOLSWithNoGCD6Construction](#) class.

Parameters

<i>order</i>	Order of the SOLS.
--------------	--------------------

7.142.3 Member Function Documentation

7.142.3.1 ConstructDesign()

[SOLS](#) TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWith↔
NoGCD6Construction.ConstructDesign ()

Construct a SOLS.

Returns

Constructed SOLS.

The documentation for this class was generated from the following file:

- SOLSWithNoGCD6Construction.cs

7.143 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ LatinSquares.SelfOrthogonal.SOLSWithNoGCD6Construction↔ Tests Class Reference

Tests for SOLSWithNoGCD6Construction.

Public Member Functions

- void [SOLSConstructionTest](#) ()
Test on the orders required for

7.143.1 Detailed Description

Tests for SOLSWithNoGCD6Construction.

7.143.2 Member Function Documentation

7.143.2.1 SOLSConstructionTest()

```
void TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.↔
SOLSWithNoGCD6ConstructionTests.SOLSConstructionTest ( )
```

Test on the orders required for

The documentation for this class was generated from the following file:

- SOLSWithNoGCD6ConstructionTests.cs

7.144 TournamentPlanner.Startup Class Reference

Public Member Functions

- **Startup** (IConfiguration configuration)
- void **ConfigureServices** (IServiceCollection services)
- void **Configure** (IApplicationBuilder app, IWebHostEnvironment env)

Properties

- IConfiguration **Configuration** [get]

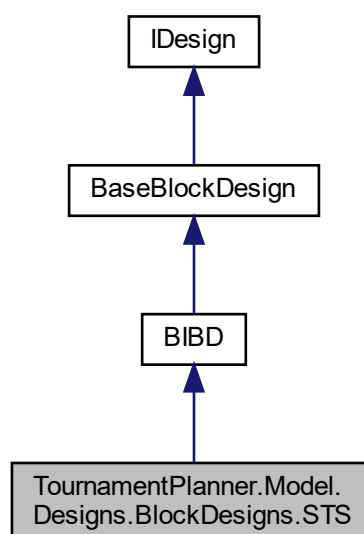
The documentation for this class was generated from the following file:

- Startup.cs

7.145 TournamentPlanner.Model.Designs.BlockDesigns.STS Class Reference

Class representing a Steiner triple system. A Steiner triple system is a [BIBD](#) with $K = 3$ and $\Lambda = 1$.

Inheritance diagram for TournamentPlanner.Model.Designs.BlockDesigns.STS:



Public Member Functions

- [STS](#) (int v)
Initializes a new instance of the [STS](#) class.
- [STS](#) (int v, List< List< int >> blocks)
Initializes a new instance of the [STS](#) class.

Additional Inherited Members

7.145.1 Detailed Description

Class representing a Steiner triple system. A Steiner triple system is a [BIBD](#) with $K = 3$ and $\text{Lambda} = 1$.

7.145.2 Constructor & Destructor Documentation

7.145.2.1 STS() [1/2]

```
TournamentPlanner.Model.Designs.BlockDesigns.STS.STS (
    int v )
```

Initializes a new instance of the [STS](#) class.

Parameters

v	Set size.
---	-----------

7.145.2.2 STS() [2/2]

```
TournamentPlanner.Model.Designs.BlockDesigns.STS.STS (
    int v,
    List< List< int >> blocks )
```

Initializes a new instance of the [STS](#) class.

Parameters

v	Set size.
blocks	The blocks.

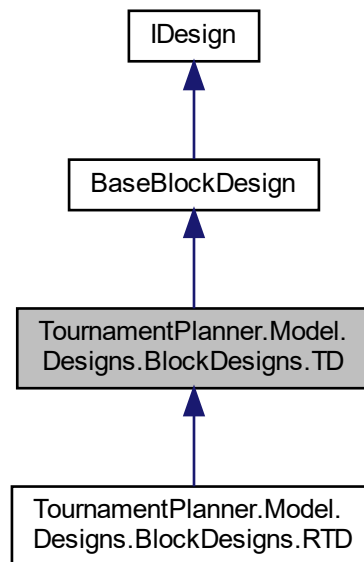
The documentation for this class was generated from the following file:

- STS.cs

7.146 TournamentPlanner.Model.Designs.BlockDesigns.TD Class Reference

Class representing the transversal design.

Inheritance diagram for TournamentPlanner.Model.Designs.BlockDesigns.TD:



Public Member Functions

- **TD** (int k, int m)
Initializes a new instance of the TD class.
- **TD** (int k, int m, List< List< int >> groups, List< List< int >> blocks)
Initializes a new instance of the TD class.

Properties

- int **K** [get]
Gets the block size.
- int **M** [get]
Gets the group size.
- List< List< int >> **Groups** [get, protected set]
Gets or sets the groups partitioning the elements. Should be of size M.

Additional Inherited Members

7.146.1 Detailed Description

Class representing the transversal design.

7.146.2 Constructor & Destructor Documentation

7.146.2.1 TD() [1/2]

```
TournamentPlanner.Model.Designs.BlockDesigns.TD.TD (
    int k,
    int m )
```

Initializes a new instance of the [TD](#) class.

Parameters

<i>k</i>	Block sizes.
<i>m</i>	Group sizes.

7.146.2.2 TD() [2/2]

```
TournamentPlanner.Model.Designs.BlockDesigns.TD.TD (
    int k,
    int m,
    List< List< int >> groups,
    List< List< int >> blocks )
```

Initializes a new instance of the [TD](#) class.

Parameters

<i>k</i>	Block sizes.
<i>m</i>	Group sizes.
<i>groups</i>	The groups.
<i>blocks</i>	The blocks.

7.146.3 Property Documentation

7.146.3.1 Groups

```
List<List<int> > TournamentPlanner.Model.Designs.BlockDesigns.TD.Groups [get], [protected set]
```

Gets or sets the groups partitioning the elements. Should be of size M.

7.146.3.2 K

```
int TournamentPlanner.Model.Designs.BlockDesigns.TD.K [get]
```

Gets the block size.

7.146.3.3 M

```
int TournamentPlanner.Model.Designs.BlockDesigns.TD.M [get]
```

Gets the group size.

The documentation for this class was generated from the following file:

- TD.cs

7.147 **TournamentPlannerTests.Algorithms.ConstructionAlgorithms. BlockDesigns.TDConstruction.TD17ConstructionTests Class Reference**

Tests for the TD17Construction.

Public Member Functions

- void **TestTD17_19Construction** ()

7.147.1 Detailed Description

Tests for the TD17Construction.

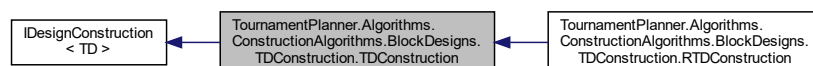
The documentation for this class was generated from the following file:

- TD17ConstructionTests.cs

7.148 **TournamentPlanner.Algorithms.ConstructionAlgorithms.Block Designs.TDConstruction.TDConstruction Class Reference**

Construction of TD(k, g) that are then used to construct Whist tournaments.

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction:



Public Member Functions

- [TDConstruction](#) (int *k*, int *m*)
Initializes a new instance of the TDConstruction class.
- [TD ConstructDesign](#) ()
Construct a TD(17, g).

Protected Attributes

- int *k*
Block sizes of the TD.
- int *m*
Group sizes of the TD.

7.148.1 Detailed Description

Construction of TD(*k*, *g*) that are then used to construct Whist tournaments.

7.148.2 Constructor & Destructor Documentation**7.148.2.1 TDConstruction()**

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction.↔
TDConstruction (
    int k,
    int m )
```

Initializes a new instance of the [TDConstruction](#) class.

Parameters

<i>k</i>	Size of the blocks.
<i>m</i>	Size of the groups.

7.148.3 Member Function Documentation**7.148.3.1 ConstructDesign()**

```
TD TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction.↔
ConstructDesign ( )
```

Construct a TD(17, *g*).

Returns

Constructed TD.

7.148.4 Member Data Documentation**7.148.4.1 k**

```
int TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction.↔
k [protected]
```

Block sizes of the TD.

7.148.4.2 m

```
int TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction.↔
m [protected]
```

Group sizes of the TD.

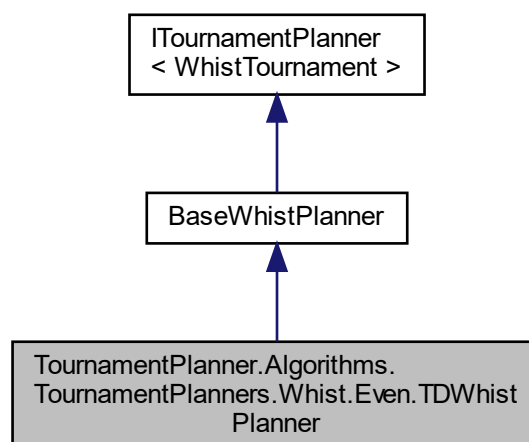
The documentation for this class was generated from the following file:

- TDConstruction.cs

7.149 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.TDWhistPlanner Class Reference

Same method as in the larger whist construction but with TD (which is also a GDD basically). TODO unite.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.TDWhistPlanner:



Public Member Functions

- [TDWhistPlanner](#) (int v)
Initializes a new instance of the [TDWhistPlanner](#) class.
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.149.1 Detailed Description

Same method as in the larger whist construction but with TD (which is also a GDD basically). TODO unite.

7.149.2 Constructor & Destructor Documentation

7.149.2.1 TDWhistPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.TDWhistPlanner.TDWhistPlanner (
    int v )
```

Initializes a new instance of the [TDWhistPlanner](#) class.

Parameters

v	The order.
---	------------

The documentation for this class was generated from the following file:

- [TDWhistPlanner.cs](#)

7.150 TournamentPlannerTests.Algorithms.TournamentPlanners.[↩](#) Whist.Even.TDWhistPlannerTests Class Reference

Tests for the TDWhistPlanner.

Public Member Functions

- void [TestWhist56Construction](#) ()

7.150.1 Detailed Description

Tests for the TDWhistPlanner.

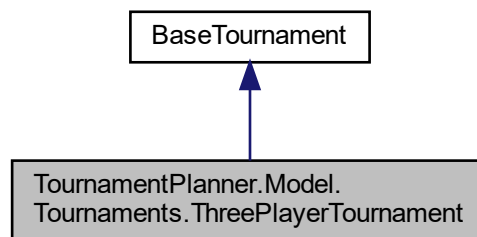
The documentation for this class was generated from the following file:

- TDWhistPlannerTests.cs

7.151 TournamentPlanner.Model.Tournaments.ThreePlayerTournament Class Reference

Class representing a three player tournament.

Inheritance diagram for TournamentPlanner.Model.Tournaments.ThreePlayerTournament:



Public Member Functions

- [ThreePlayerTournament](#) (int nbPlayers, List< List< List< int >>> gamesByRounds, List< List< int >> groups=null)
Initializes a new instance of the [ThreePlayerTournament](#) class.
- int [TotalNumberOfGames](#) ()
Get the total number of games in the tournament.

Properties

- List< List< int > > [PlayerGroups](#) [get]
Gets the player groups. Player that are together in a group won't play with each others.

Additional Inherited Members

7.151.1 Detailed Description

Class representing a three player tournament.

7.151.2 Constructor & Destructor Documentation

7.151.2.1 ThreePlayerTournament()

```
TournamentPlanner.Model.Tournaments.ThreePlayerTournament.ThreePlayerTournament (
    int nbPlayers,
    List< List< List< int >>> gamesByRounds,
    List< List< int >> groups = null )
```

Initializes a new instance of the [ThreePlayerTournament](#) class.

Parameters

<i>nbPlayers</i>	Number of players.
<i>gamesByRounds</i>	Games by round.
<i>groups</i>	The groups. Optional.

7.151.3 Member Function Documentation

7.151.3.1 TotalNumberOfGames()

```
int TournamentPlanner.Model.Tournaments.ThreePlayerTournament.TotalNumberOfGames ( )
```

Get the total number of games in the tournament.

Returns

The total number of games.

7.151.4 Property Documentation

7.151.4.1 PlayerGroups

```
List<List<int> > TournamentPlanner.Model.Tournaments.ThreePlayerTournament.PlayerGroups [get]
```

Gets the player groups. Player that are together in a group won't play with each others.

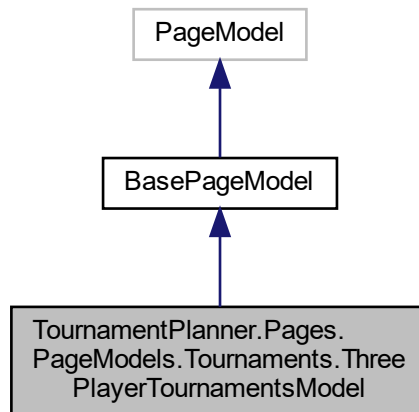
The documentation for this class was generated from the following file:

- ThreePlayerTournament.cs

7.152 TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournamentsModel Class Reference

Page model for three player tournaments.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournamentsModel:



Public Member Functions

- void [OnPost](#) ()

Properties

- int [NbPlayers](#) = 15 [get, set]
Gets or sets the number of players in the tournament.
- [ThreePlayerTournament KTS](#) [Tournament](#) [get]
Gets the tournament constructed using KTS.
- [ThreePlayerTournament GDD](#) [Tournament](#) [get]
Gets the tournament constructed using GDD.

7.152.1 Detailed Description

Page model for three player tournaments.

7.152.2 Property Documentation

7.152.2.1 GDDTournament

```
ThreePlayerTournament TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournaments←  
Model.GDDTournament [get]
```

Gets the tournament constructed using GDD.

7.152.2.2 KTSTournament

```
ThreePlayerTournament TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournaments←  
Model.KTSTournament [get]
```

Gets the tournament constructed using KTS.

7.152.2.3 NbPlayers

```
int TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournamentsModel.NbPlayers = 15  
[get], [set]
```

Gets or sets the number of players in the tournament.

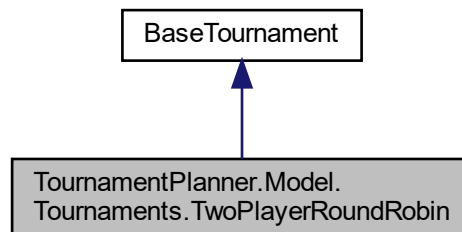
The documentation for this class was generated from the following file:

- ThreePlayerTournaments.cshtml.cs

7.153 TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin Class Reference

Represents a 2 player round robin tournament schedule.

Inheritance diagram for TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin:



Public Member Functions

- [TwoPlayerRoundRobin](#) (int order)
Initializes a new instance of the [TwoPlayerRoundRobin](#) class.
- void [AddRoundSchedule](#) (List< List< int >> round)
Add round.
- void [AddRoundSchedule](#) (List< List< int >> round, int inactive)
Add round and inactive player.
- new string [GameToString](#) (List< int > game)

Public Attributes

- List< int > [InactiveByRounds](#) = new List<int>()
Gets or sets the inactive players by round.

Additional Inherited Members

7.153.1 Detailed Description

Represents a 2 player round robin tournament schedule.

7.153.2 Constructor & Destructor Documentation

7.153.2.1 TwoPlayerRoundRobin()

```
TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin.TwoPlayerRoundRobin (
    int order )
```

Initializes a new instance of the [TwoPlayerRoundRobin](#) class.

Parameters

<i>order</i>	The order.
--------------	------------

7.153.3 Member Function Documentation

7.153.3.1 AddRoundSchedule() [1/2]

```
void TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin.AddRoundSchedule (
    List< List< int >> round )
```

Add round.

Parameters

<i>round</i>	The round.
--------------	------------

7.153.3.2 AddRoundSchedule() [2/2]

```
void TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin.AddRoundSchedule (
    List< List< int >> round,
    int inactive )
```

Add round and inactive player.

Parameters

<i>round</i>	The round.
<i>inactive</i>	The inactive player.

7.153.4 Member Data Documentation

7.153.4.1 InactiveByRounds

```
List<int> TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin.InactiveByRounds = new
List<int>()
```

Gets or sets the inactive players by round.

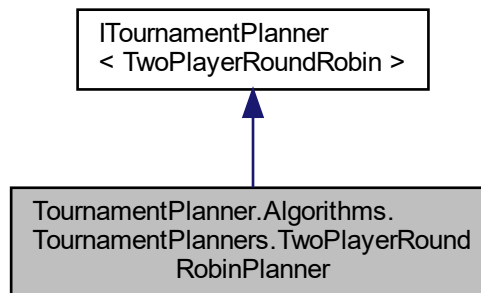
The documentation for this class was generated from the following file:

- TwoPlayerRoundRobin.cs

7.154 TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlanner Class Reference ↩↪

Tournament planner for two player round robins.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlanner:



Public Member Functions

- [TwoPlayerRoundRobinPlanner](#) (int nbPlayers)
Initializes a new instance of the [TwoPlayerRoundRobinPlanner](#) class.
- [TwoPlayerRoundRobin PlanTournament](#) ()
Plan a two player round robin tournament.

Properties

- int [NbPlayers](#) [get]
Gets the number of players.
- int [N](#) [get]
Gets N as the number of players divided by 2 and floored.
- bool [OddNbPlayers](#) [get]
Gets a value indicating whether the number of players is odd.

7.154.1 Detailed Description

Tournament planner for two player round robins.

7.154.2 Constructor & Destructor Documentation

7.154.2.1 TwoPlayerRoundRobinPlanner()

```

TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlanner.TwoPlayerRoundRobinPlanner (
    int nbPlayers )
  
```

Initializes a new instance of the [TwoPlayerRoundRobinPlanner](#) class.

Parameters

<code>nbPlayers</code>	The number of players.
------------------------	------------------------

7.154.3 Member Function Documentation

7.154.3.1 PlanTournament()

`TwoPlayerRoundRobin` TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlanner.PlanTournament ()

Plan a two player round robin tournament.

Returns

The tournament.

7.154.4 Property Documentation

7.154.4.1 N

`int` TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlanner.N [get]

Gets N as the number of players divided by 2 and floored.

7.154.4.2 NbPlayers

`int` TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlanner.NbPlayers [get]

Gets the number of players.

7.154.4.3 OddNbPlayers

`bool` TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlanner.OddNbPlayers [get]

Gets a value indicating whether the number of players is odd.

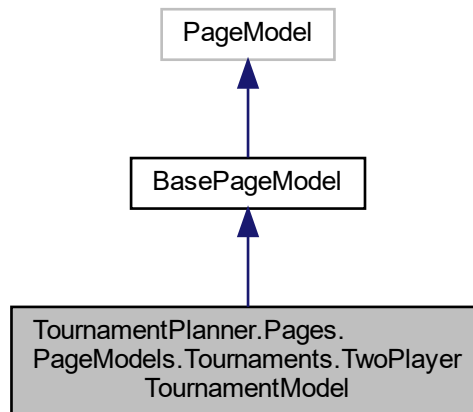
The documentation for this class was generated from the following file:

- TwoPlayerRoundRobinPlanner.cs

7.155 TournamentPlanner.Pages.PageModels.Tournaments.TwoPlayer TournamentModel Class Reference ↔

Page model for the construction of two player round robin tournaments.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Tournaments.TwoPlayerTournamentModel:



Public Member Functions

- void [OnPost](#) ()

Properties

- int [NbPlayers](#) = 8 [get, set]
Gets or sets the number of players.
- [TwoPlayerRoundRobin Tournament](#) [get]
Gets the constructed tournament.

7.155.1 Detailed Description

Page model for the construction of two player round robin tournaments.

7.155.2 Property Documentation

7.155.2.1 NbPlayers

```
int TournamentPlanner.Pages.PageModels.Tournaments.TwoPlayerTournamentModel.NbPlayers = 8
[get], [set]
```

Gets or sets the number of players.

7.155.2.2 Tournament

```
TwoPlayerRoundRobin TournamentPlanner.Pages.PageModels.Tournaments.TwoPlayerTournamentModel.←
Tournament [get]
```

Gets the constructed tournament.

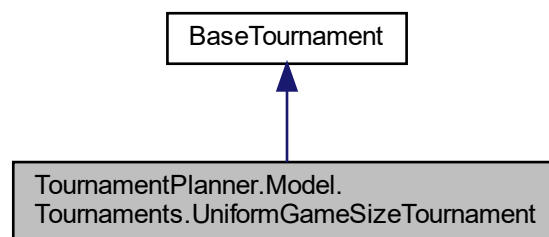
The documentation for this class was generated from the following file:

- TwoPlayerTournament.cshtml.cs

7.156 TournamentPlanner.Model.Tournaments.UniformGameSize← Tournament Class Reference

[Model](#) for the tournament, where the sizes of the games are uniform - all the games have k-players.

Inheritance diagram for TournamentPlanner.Model.Tournaments.UniformGameSizeTournament:



Public Member Functions

- [UniformGameSizeTournament](#) (int nbPlayers, List< List< List< int >>> rounds, int objectiveValue, List< List< int >> leftOverPlayersByRounds=null)
Initializes a new instance of the [UniformGameSizeTournament](#) class.
- int [TotalNumberOfGames](#) ()
Gets the total number of games.

Properties

- List< List< int > > [LeftOverPlayersByRounds](#) [get]
Gets the left over players.
- int [ObjectiveValue](#) [get]
Gets the criterion value of the solution.

Additional Inherited Members

7.156.1 Detailed Description

[Model](#) for the tournament, where the sizes of the games are uniform - all the games have k-players.

7.156.2 Constructor & Destructor Documentation

7.156.2.1 UniformGameSizeTournament()

```
TournamentPlanner.Model.Tournaments.UniformGameSizeTournament.UniformGameSizeTournament (
    int nbPlayers,
    List< List< List< int >>> rounds,
    int objectiveValue,
    List< List< int >> leftOverPlayersByRounds = null )
```

Initializes a new instance of the [UniformGameSizeTournament](#) class.

Parameters

<i>nbPlayers</i>	Number of players.
<i>rounds</i>	The rounds.
<i>objectiveValue</i>	The objective value reached by the solver.
<i>leftOverPlayersByRounds</i>	The left over players by rounds.

7.156.3 Member Function Documentation

7.156.3.1 TotalNumberOfGames()

```
int TournamentPlanner.Model.Tournaments.UniformGameSizeTournament.TotalNumberOfGames ( )
```

Gets the total number of games.

Returns

The total number of games.

7.156.4 Property Documentation

7.156.4.1 LeftOverPlayersByRounds

```
List<List<int> > TournamentPlanner.Model.Tournaments.UniformGameSizeTournament.LeftOver←  
PlayersByRounds [get]
```

Gets the left over players.

7.156.4.2 ObjectiveValue

```
int TournamentPlanner.Model.Tournaments.UniformGameSizeTournament.ObjectiveValue [get]
```

Gets the criterion value of the solution.

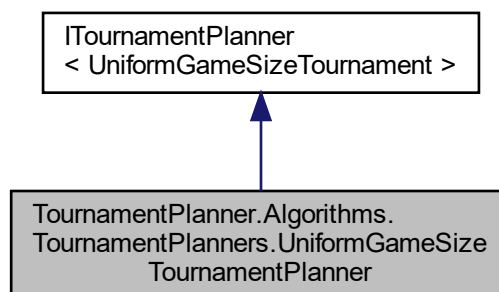
The documentation for this class was generated from the following file:

- UniformGameSizeTournament.cs

7.157 TournamentPlanner.Algorithms.TournamentPlanners.Uniform← GameSizeTournamentPlanner Class Reference

CP approach to the planning of tournaments, where the game sizes are uniform.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.UniformGameSizeTournament←
Planner:



Public Member Functions

- [UniformGameSizeTournamentPlanner](#) (int nbPlayers, int gameSize, int nbRounds, int timeLimit, CpSolver↔ SolutionCallback cb=null)
Initializes a new instance of the [UniformGameSizeTournamentPlanner](#) class.
- [UniformGameSizeTournament PlanTournament](#) ()

7.157.1 Detailed Description

CP approach to the planning of tournaments, where the game sizes are uniform.

7.157.2 Constructor & Destructor Documentation

7.157.2.1 UniformGameSizeTournamentPlanner()

```
TournamentPlanner.Algorithms.TournamentPlanners.UniformGameSizeTournamentPlanner.UniformGame↔
SizeTournamentPlanner (
    int nbPlayers,
    int gameSize,
    int nbRounds,
    int timeLimit,
    CpSolverSolutionCallback cb = null )
```

Initializes a new instance of the [UniformGameSizeTournamentPlanner](#) class.

Parameters

<i>nbPlayers</i>	Then number of players.
<i>gameSize</i>	The size of a game.
<i>nbRounds</i>	The number of rounds.
<i>timeLimit</i>	The time limit for the solver.
<i>cb</i>	The optional solution callback.

The documentation for this class was generated from the following file:

- [UniformGameSizeTournamentPlanner.cs](#)

7.158 TournamentPlannerTests.Algorithms.TournamentPlanners.↔ UniformGameSizeTournamentPlannerTests Class Reference

Tests for the [UniformGameSizeTournamentPlanner](#).

Public Member Functions

- void [KTS9EquivalentTest](#) ()

7.158.1 Detailed Description

Tests for the UniformGameSizeTournamentPlanner.

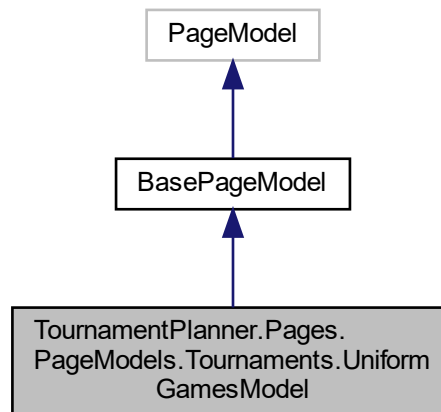
The documentation for this class was generated from the following file:

- UniformGameSizeTournamentPlannerTests.cs

7.159 TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel Class Reference

Page model for the tournament, in which the games are uniformly sized.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel:



Public Member Functions

- void `OnPost` ()

Properties

- int `NbPlayers` = 13 [get, set]
Gets or sets the number of players.
- int `GameSize` = 4 [get, set]
Gets or sets the game size.
- int `NbRounds` = 3 [get, set]
Gets or sets the number of rounds.
- int `TimeLimit` = 10 [get, set]
Gets or sets the time limit.
- `UniformGameSizeTournament Tournament` [get]
Gets the constructed tournament.

7.159.1 Detailed Description

Page model for the tournament, in which the games are uniformly sized.

7.159.2 Property Documentation

7.159.2.1 GameSize

```
int TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel.GameSize = 4 [get], [set]
```

Gets or sets the game size.

7.159.2.2 NbPlayers

```
int TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel.NbPlayers = 13 [get],  
[set]
```

Gets or sets the number of players.

7.159.2.3 NbRounds

```
int TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel.NbRounds = 3 [get], [set]
```

Gets or sets the number of rounds.

7.159.2.4 TimeLimit

```
int TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel.TimeLimit = 10 [get],  
[set]
```

Gets or sets the time limit.

7.159.2.5 Tournament

`UniformGameSizeTournament` TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel.↔
 Tournament [get]

Gets the constructed tournament.

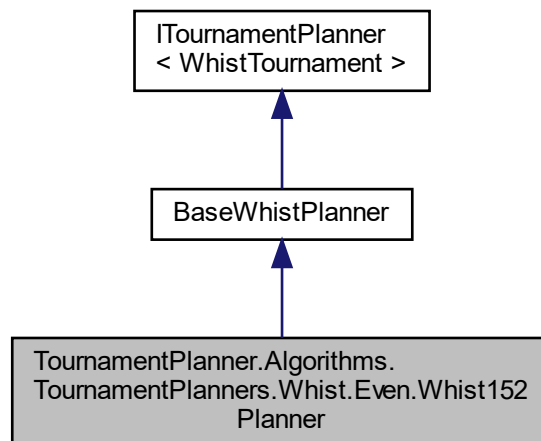
The documentation for this class was generated from the following file:

- UniformGames.cshtml.cs

7.160 TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.Whist152Planner Class Reference

Class for the construction of [Whist](#) 152. Uses even SAMDRR.

Inheritance diagram for TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.Whist152Planner:



Public Member Functions

- [Whist152Planner](#) ()
Initializes a new instance of the [Whist152Planner](#) class.
- override [WhistTournament PlanTournament](#) ()
Plan the [Whist](#) tournament.

Additional Inherited Members

7.160.1 Detailed Description

Class for the construction of [Whist](#) 152. Uses even SAMDRR.

7.160.2 Constructor & Destructor Documentation

7.160.2.1 Whist152Planner()

```
TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.Whist152Planner.Whist152Planner ( )
```

Initializes a new instance of the [Whist152Planner](#) class.

The documentation for this class was generated from the following file:

- Whist152Planner.cs

7.161 TournamentPlannerTests.Algorithms.TournamentPlanners.↔ Whist.Even.Whist152PlannerTests Class Reference

Tests for the Whist152Planner.

Public Member Functions

- void [Whist152ConstructionTest](#) ()
Test the construction of Whist 152.

7.161.1 Detailed Description

Tests for the Whist152Planner.

7.161.2 Member Function Documentation

7.161.2.1 Whist152ConstructionTest()

```
void TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.Whist152PlannerTests.↔  
Whist152ConstructionTest ( )
```

Test the construction of [Whist 152](#).

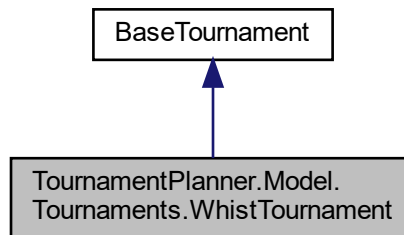
The documentation for this class was generated from the following file:

- Whist152PlannerTests.cs

7.162 TournamentPlanner.Model.Tournaments.WhistTournament Class Reference

The model of the whist tournament.

Inheritance diagram for TournamentPlanner.Model.Tournaments.WhistTournament:



Public Member Functions

- [WhistTournament](#) (int order)
Initializes a new instance of the [WhistTournament](#) class.
- [WhistTournament](#) (int order, List< List< List< int >>> rounds)
Initializes a new instance of the [WhistTournament](#) class.
- [WhistTournament](#) (int order, List< List< List< int >>> rounds, List< int > leftOvers)
Initializes a new instance of the [WhistTournament](#) class.

Properties

- List< int > [LeftOverByRounds](#) = new () [get]
Gets the left over players.

Additional Inherited Members

7.162.1 Detailed Description

The model of the whist tournament.

7.162.2 Constructor & Destructor Documentation

7.162.2.1 WhistTournament() [1/3]

```
TournamentPlanner.Model.Tournaments.WhistTournament.WhistTournament (
    int order )
```

Initializes a new instance of the [WhistTournament](#) class.

Parameters

<i>order</i>	The order.
--------------	------------

7.162.2.2 WhistTournament() [2/3]

```
TournamentPlanner.Model.Tournaments.WhistTournament.WhistTournament (
    int order,
    List< List< List< int >>> rounds )
```

Initializes a new instance of the [WhistTournament](#) class.

Parameters

<i>order</i>	The order.
<i>rounds</i>	The rounds.

7.162.2.3 WhistTournament() [3/3]

```
TournamentPlanner.Model.Tournaments.WhistTournament.WhistTournament (
    int order,
    List< List< List< int >>> rounds,
    List< int > leftOvers )
```

Initializes a new instance of the [WhistTournament](#) class.

Parameters

<i>order</i>	The order.
<i>rounds</i>	The rounds.
<i>leftOvers</i>	The left over players.

7.162.3 Property Documentation**7.162.3.1 LeftOverByRounds**

```
List<int> TournamentPlanner.Model.Tournaments.WhistTournament.LeftOverByRounds = new () [get]
```

Gets the left over players.

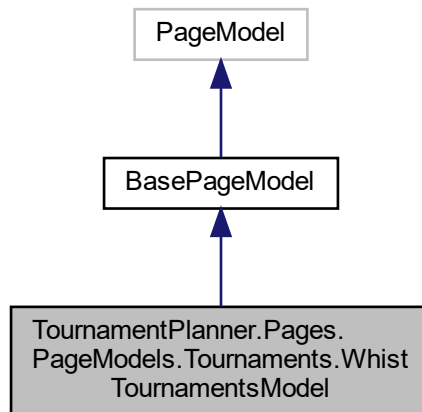
The documentation for this class was generated from the following file:

- WhistTournament.cs

7.163 TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel Class Reference

Page model for the Whist tournaments.

Inheritance diagram for TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel:



Public Member Functions

- void `OnPost` ()

Properties

- int `NbPlayers` = 9 [get, set]
Gets or sets the number of players in the tournament.
- `WhistTournament Whist` [get]
Gets the constructed tournament.

7.163.1 Detailed Description

Page model for the Whist tournaments.

7.163.2 Property Documentation

7.163.2.1 NbPlayers

```
int TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel.NbPlayers = 9 [get],
[set]
```

Gets or sets the number of players in the tournament.

7.163.2.2 Whist

```
WhistTournament TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel.Whist
[get]
```

Gets the constructed tournament.

The documentation for this class was generated from the following file:

- WhistTournaments.cshtml.cs

7.164 TournamentPlannerTests.Model.Tournaments.WhistTournamentValidator Class Reference

A validator for whist tournaments.

Static Public Member Functions

- static void [ValidateWhistTournament](#) ([WhistTournament](#) whist)
Validate the properties of a whist tournament.

7.164.1 Detailed Description

A validator for whist tournaments.

7.164.2 Member Function Documentation

7.164.2.1 ValidateWhistTournament()

```
static void TournamentPlannerTests.Model.Tournaments.WhistTournamentValidator.ValidateWhist<->
Tournament (
    WhistTournament whist ) [static]
```

Validate the properties of a whist tournament.

Parameters

<i>whist</i>	The whist tournament to validate.
--------------	-----------------------------------

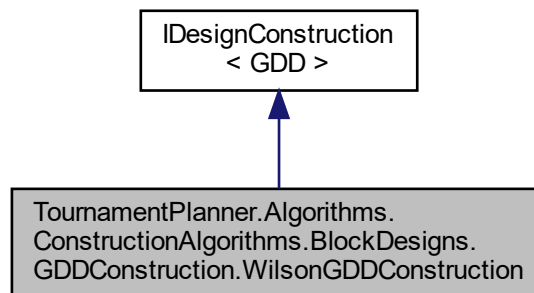
The documentation for this class was generated from the following file:

- WhistTournamentValidator.cs

7.165 TournamentPlanner.Algorithms.ConstructionAlgorithms.Block↔ Designs.GDDConstruction.WilsonGDDConstruction Class Reference

Wilson's construction methods using smaller GDDs to construct bigger ones. <https://www.semanticscholar.org/paper/An-Existence-Theory-for-Pairwise-Balanced-Designs-Wilson/b9c1c6bb1d7828b4f47f2ee3330dab26adb946dc>

Inheritance diagram for TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.WilsonGDDConstruction:



Public Member Functions

- [WilsonGDDConstruction](#) (int w, GDD initGDD)
Initializes a new instance of the [WilsonGDDConstruction](#) class.
- [GDD ConstructDesign](#) ()
Construct a GDD using Wilson's method.

7.165.1 Detailed Description

Wilson's construction methods using smaller GDDs to construct bigger ones. <https://www.semanticscholar.org/paper/An-Existence-Theory-for-Pairwise-Balanced-Designs-Wilson/b9c1c6bb1d7828b4f47f2ee3330dab26adb946dc>

7.165.2 Constructor & Destructor Documentation

7.165.2.1 WilsonGDDConstruction()

```
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Wilson↔
GDDConstruction.WilsonGDDConstruction (
    int w,
    GDD initGDD )
```

Initializes a new instance of the [WilsonGDDConstruction](#) class.

Parameters

<i>w</i>	The multiplier.
<i>initGDD</i>	The initial GDD.

7.165.3 Member Function Documentation

7.165.3.1 ConstructDesign()

```
GDD TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Wilson↔
GDDConstruction.ConstructDesign ( )
```

Construct a GDD using wilsons method.

Returns

Constructed GDD.

The documentation for this class was generated from the following file:

- WilsonGDDConstruction.cs

7.166 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.↔ BlockDesigns.GDDConstruction.WilsonGDDConstructionTests Class Reference

Tests for the WilsonGDDConstruction.

Public Member Functions

- void **GDD81ConstructionTest** ()
- void **GDD42ConstructionTest** ()
- void **GDD45Test** ()
- void **GDD78Construction** ()

7.166.1 Detailed Description

Tests for the WilsonGDDConstruction.

The documentation for this class was generated from the following file:

- WilsonGDDConstructionTests.cs

Index

AddBlock
TournamentPlanner.Model.Designs.BlockDesigns.BIBD, 56
TournamentPlanner.Model.Designs.BlockDesigns.GDD, 88
TournamentPlanner.Model.Designs.BlockDesigns.PBB, 165

AddBlocks
TournamentPlanner.Model.Designs.BlockDesigns.BIBD, 56

AddOrthogonalLatinSquare
TournamentPlanner.Model.Designs.LatinSquares.MOB, 117

AddParallelClass
TournamentPlanner.Model.Designs.BlockDesigns.KTS, 108
TournamentPlanner.Model.Designs.BlockDesigns.RB, 180
TournamentPlanner.Model.Designs.BlockDesigns.RTD, 187

AddParallelClassAndBlocks
TournamentPlanner.Model.Designs.BlockDesigns.GDD, 88
TournamentPlanner.Model.Designs.BlockDesigns.KTS, 108

AddRound
TournamentPlanner.Model.Tournaments.BaseTournament, 50

AddRoundSchedule
TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin, 226, 227

ArgumentException
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Planner.BaseMOLSConstruction, 47

Automatic
TournamentPlanner.Pages.PageModels.Designs, 30

B
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction, 42
TournamentPlanner.Model.Designs.BlockDesigns.BIBD, 56

Balanced3PlayerTournamentPlanner
TournamentPlanner.Algorithms.TournamentPlanners.Balanced3PlayerTournamentPlanner, 40

BaseBIBDConstruction
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction, 41

BaseBlockDesign
TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign, 44
BaseMOLSConstruction
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Planner.BaseMOLSConstruction, 46

BaseTournament
TournamentPlanner.Model.Tournaments.BaseTournament, 50

BaseWhistPlanner
TournamentPlanner.Algorithms.TournamentPlanners.Whist.BaseWhistPlanner, 53

BIBD
TournamentPlanner.Model.Designs.BlockDesigns.BIBD, 55

BipartiteTournament
TournamentPlanner.Model.Tournaments.BipartiteTournament, 60
BipartiteTournamentPlanner
TournamentPlanner.Algorithms.TournamentPlanners.BipartiteTournamentPlanner, 62

BlockBuildingMethod
TournamentPlanner.Pages.PageModels.Designs.BIBDModel, 58

Blocks
TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign, 44

BlockSizes
TournamentPlanner.Model.Designs.BlockDesigns.GDD, 89

BoseBlockBuilding
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Planner.BaseMOLSConstruction, 30

BoseSTSCONSTRUCTION
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BoseSTSCONSTRUCTION, 64

BoseSTSCONSTRUCTIONTest
TournamentPlanner.Tests.Algorithms.ConstructionAlgorithms.BlockDesigns.BoseSTSCONSTRUCTIONTest, 65

CommutativeIdempotentLatinSquareConstruction
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Planner.CommutativeIdempotentLatinSquareConstruction, 67

Constrained3PlayerTournamentPlanner
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.Constrained3PlayerTournamentPlanner, 182

ConstructionAlgorithms
TournamentPlanner.Algorithms.ConstructionAlgorithms, 42

- TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BasePageModel, 65
- TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.MooreRBIBDConstruction, 124
- TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.RecursiveRBIBDConstruction, 182
- TournamentPlanner.Algorithms.ConstructionAlgorithms.EvenEvenProductWhistPlanner, 200
- TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmoothKTSCONSTRUCTION, 209
- TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.ResolvableGDDConstruction, 185
- TournamentPlanner.Algorithms.ConstructionAlgorithms.EvenOddShouldThrowExceptionTest, 204
- TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.WilsonGDDConstruction, 244
- TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.PBD34Construction, 167
- TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.SmallOrderPBDConstruction, 206
- TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.RTDCONSTRUCTION, 189
- TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction, 219
- TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction< IDesign >, 102
- TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.CommutativeIdempotentLatinSquareConstruction, 67
- TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.OrthogonalArrayConstruction, 135
- TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.BaseMOLSCONSTRUCTION, 46
- TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNoGCD6Construction, 213
- TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SingleLatinSquareConstruction, 197
- ConstructionTestForSeveralOrdersOfRBIBD
- TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.MooreRBIBDConstructionTests, 125
- ConstructPairwiseTestCases
- TournamentPlanner.Algorithms.Testing.PairwiseTestCasesConstruction, 160
- CPApproach
- TournamentPlanner.Pages.PageModels.Designs, 30
- CyclicOddWhistPlanner
- TournamentPlanner.Algorithms.TournamentPlanners.WhistOdd.CyclicOddWhistPlanner, 70
- CyclicWhistConstructionsTest
- TournamentPlannerTests.Algorithms.TournamentPlanners.WhistEven.CyclicWhistPlannerTests, 73
- CyclicWhistPlanner
- TournamentPlanner.Algorithms.TournamentPlanners.WhistEven.CyclicWhistPlanner, 72
- DifferenceApproach
- TournamentPlanner.Pages.PageModels.Designs, 30
- DisplayResults
- TournamentPlanner.Pages.PageModels.BasePageModel, 49
- TournamentPlanner.Pages.PageModels.BasePageModel, 49
- TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenEvenProductWhistPlanner, 79
- TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.EvenOddProductWhistPlanner, 81
- TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquareConstructionTest, 198
- WilsonGDDConstruction, 30
- SmallOrderPBDConstruction, 30
- GameSize
- TournamentPlanner.Pages.PageModels.Tournaments.UniformGames, 236
- GameToString
- TournamentPlanner.Model.Tournaments.BaseTournament, 51
- GDD
- TournamentPlanner.Model.Designs.BlockDesigns.GDD, 86, 87
- OrthogonalArrayConstruction, 92
- BaseMOLSCONSTRUCTION, 185
- GDDForWhistConstruction
- SingleLatinSquareConstruction, 90
- GDDTournament
- BlockDesigns.BIBDConstruction.MooreRBIBDConstructionTests, 224
- GeneralEvenWhistPlanner
- TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.GeneralEvenWhistPlanner, 94
- GeneralOddWhistPlanner
- TournamentPlanner.Algorithms.TournamentPlanners.Whist.Odd.GeneralOddWhistPlanner, 97
- GeneralIPBD3NPlus1Construction
- WhistOdd.CyclicOddWhistPlanner, 99
- GetGroupSizes
- WhistEven.CyclicWhistPlannerTests, 88
- Model.Designs.BlockDesigns.GDD, 118
- GetMOLSMatrices
- WhistEven.CyclicWhistPlanner, 118
- TournamentPlanner.Model.Designs.LatinSquares.MOLS, 118
- GetNumberOfMOLS
- Model.Designs.LatinSquares.MOLS, 118
- Groups

TournamentPlanner.Model.Designs.BlockDesigns.GDDTSTournament	89	TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTo	225
TournamentPlanner.Model.Designs.BlockDesigns.TD,	217		
		L	
Height		TournamentPlanner.Pages.PageModels.Designs.BIBDModel,	58
TournamentPlanner.Model.Designs.OrthogonalArray,	132	Lambda	
InactiveByRounds		TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns	42
TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin	227	TournamentPlanner.Model.Designs.BlockDesigns.BIBD,	57
IncrementalBIBDConstructionTest		LargerConstructionWithoutValidationTest	
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.TournamentPlannerTests.Algorithms.TournamentPlannerTests.WhistEven	99	TournamentPlannerTests.Algorithms.ConstructionAlgorithms.TournamentPlannerTests.Algorithms.TournamentPlannerTests.WhistEven	95
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.IncrementalBIBDConstructionTest	103	TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns	111
IncrementalBlockBuilding		LargerWhistConstructionTest	
TournamentPlanner.Pages.PageModels.Designs,	30	TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even	95
InvalidMOLSTest		LatinSquare	
TournamentPlannerTests.Designs.MOLSSetValidatorTests	121	TournamentPlanner.Model.Designs.LatinSquares.LatinSquare,	114
InvalidOrderShouldThrowException		LatinSquares	
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares	69	LatinSquares.Orthogonal.ConstraintProgrammingMOLSPairConst	
InvalidOrderTest		TournamentPlanner.Model.Designs.LatinSquares.MOLS,	118
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.DifferenceMOLSPairConstructionTests,	77	LeftOverPlayersByRounds	
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.Orthogonal.FourKMOLSPairConstructionTests,	84	LeftOverPlayersByRounds	
IsPrimeOrPrimePowerTest		TournamentPlanner.Model.Tournaments.UniformGameSizeTournament	233
TournamentPlannerTests.Utils.PrimeFactorizationTests,	171		
IsPrimeTest		M	
TournamentPlannerTests.Utils.PrimeFactorizationTests,	171	TournamentPlanner.Model.Designs.BlockDesigns.TD,	218
K		TournamentPlanner.Pages.PageModels.Designs.GDDModel,	92
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.BaseBIBDConstruction,	42	TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns	220
TournamentPlanner.Model.Designs.BlockDesigns.BIBD,	56	Matrices	
TournamentPlanner.Model.Designs.BlockDesigns.GDD,	89	TournamentPlanner.Datastore.MongoMOLS,	122
TournamentPlanner.Model.Designs.BlockDesigns.PBD,	166	Matrix	
TournamentPlanner.Model.Designs.BlockDesigns.TD,	217	TournamentPlanner.Model.Designs.LatinSquares.LatinSquare,	115
TournamentPlanner.Pages.PageModels.Designs.BIBDModel	58	TournamentPlanner.Model.Designs.OrthogonalArray,	133
k		Method	
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction,	220	TournamentPlanner.Pages.PageModels.Designs,	30
KTS		TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels	141
TournamentPlanner.Model.Designs.BlockDesigns.KTS,	107	TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageM	149
TournamentPlanner.Pages.PageModels.Designs.KTSMModel	109	TournamentPlanner.Pages.Pages_Error,	138

TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDCConstructionTests
 207

PlanTournament RTD
 TournamentPlanner.Algorithms.TournamentPlanners.BipartiteTournamentPlannerModel.Designs.BlockDesigns.RTD,
 63 187

TournamentPlanner.Algorithms.TournamentPlanners.RoundConstructionPlanner<
 BaseTournament >, 106 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns
 TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SAMDRR38Planner,
 192 RTDConstructionTest

TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlannerTests.Algorithms.ConstructionAlgorithms.BlockDe
 193 189

TournamentPlanner.Algorithms.TournamentPlanners.Spouses.SimpleSAMDRRPlanner,
 195 SAMDRR

TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRoomPlanner, TournamentPlannerModel.Tournaments.SAMDRR,
 229 191

TournamentPlanner.Algorithms.TournamentPlanners.WhistBaseWhistPlanner, SeveralOrderTest
 53 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSq

PlayerGroups 75

TournamentPlanner.Model.Tournaments.ThreePlayerTournament, TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSq
 223 198

PrimePowerWhistPlanner SimpleSAMDRRPlanner
 TournamentPlanner.Algorithms.TournamentPlanners.Whist.OddPrimePowerWhistPlanner, TournamentPlanner.Algorithms.TournamentPlanners.Spouses.Simple
 172 195

QuadraticProgramming SingleLatinSquareConstruction
 TournamentPlanner.Pages.PageModels.Designs, TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares
 30 197

QuasigroupConstructionTest SingleLatinSquareOfOrder10Test
 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDCConstruction.QuasigroupPBDCConstructionTests
 178 198

QuasigroupPBDCConstruction TournamentPlanner.Pages.PageModels.Designs,
 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDCConstruction.QuasigroupPBDCConstruction,
 177 SkolemSTSConstruction

QuadraticConstructionTest TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns
 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDCConstruction.QuadraticProgrammingBIBDCCons
 176 SkolemSTSConstructionTest

R TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDe
 201

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDCConstruction.BaseBIBDCConstruction,
 43 TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDe

TournamentPlanner.Model.Designs.BlockDesigns.BIBD, 201
 57 SmallBIBDForWhistConstruction

RBIBD TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns
 TournamentPlanner.Model.Designs.BlockDesigns.RBIBD, 202

RecursiveKTSConstruction SmallerWhistConstructionTest
 TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even
 181 181

RecursiveKTSConstructionTest TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns
 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDCConstruction.RecursiveKTSConstruction,
 183 SmallGDDConstruction

RepresentedMOLSCount TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns
 TournamentPlanner.Model.Designs.OrthogonalArray, 206
 132 SmallPBDDUpToOrder46ConstructionTest

Resolvable3GDDConstruction TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDe
 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.Resolvable3GDDConstruction,
 184 184

Rounds SmoothKTSConstruction
 TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns
 208

- SmoothKTSConstructionTest
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction.SmoothKTSConstructionTests, 210
- SOLS
TournamentPlanner.Model.Designs.LatinSquares.SOLS, 211
TournamentPlanner.Pages.PageModels.Tournaments.UniformGames, 236
TotalNumberOfGames
- SOLSConstructionTest
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNoGCD6ConstructionTest, 213
TournamentPlanner.Model.Tournaments.UniformGameSizeTournament
- SOLSWithNoGCD6Construction
TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.SelfOrthogonal.SOLSWithNoGCD6Construction, 212
TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournament
- STS
TournamentPlanner.Model.Designs.BlockDesigns.STS, 215
TournamentPlanner.Pages.PageModels.Tournaments.TwoPlayerTournament, 231
TournamentPlanner.Pages.PageModels.Tournaments.UniformGames, 236
- TD
TournamentPlanner.Model.Designs.BlockDesigns.TD, 217
TournamentPlanner, 21
TournamentPlanner.Algorithms, 21
- TDConstruction
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.TDConstruction.TDConstruction, 219
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns,
- TDWhistPlanner
TournamentPlanner.Algorithms.TournamentPlanners.WhistEvenTDWhistPlanner, 221
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction, 21
- TeamSize
TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournamentModel, 61
TournamentPlanner, 42
- TestCases
TournamentPlanner.Model.Testing.PairwiseTestCases, 158
TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel, 162
BaseBIBDConstruction, 41
ConstructDesign, 42
K, 42
R, 43
V, 43
- TestPBD49To76Construction
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction.LargerOrderPBDCConstructionTests, 112
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.PBDConstruction, 63
- TestSAMDRR11Construction
TournamentPlannerTests.Algorithms.TournamentPlanners.ConstructDesign, 196
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction, 65
- TestSeveralValidInputs
TournamentPlannerTests.Algorithms.TournamentPlanners.WhistOldPrimePowerWhistPlannerTests, 173
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction, 123
- TestSmallGDDConstructionForAvailableOrders
TournamentPlannerTests.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.SmallGDDConstructionTests, 205
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction, 124
ModernBIBDConstruction, 124
- TestValidOrdersOfMooreConstruction
TournamentPlannerTests.Algorithms.TournamentPlanners.WhistEvenMooreWhistPlannerTests, 127
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction, 180
- TestWhist33Construction
TournamentPlannerTests.Algorithms.TournamentPlanners.ConstructDesign, 192
ConstructBlockInducedParallelClasses, 182
WhistOddCyclicOddWhistPlannerTests, 182
RecursiveKTSConstruction, 181
- TestWhist57Construction
TournamentPlannerTests.Algorithms.TournamentPlanners.WhistOddCyclicOddWhistPlannerTests, 71
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction, 200
ConstructDesign, 200
- TestWhistConstructionUsingSAMDRR
TournamentPlannerTests.Algorithms.TournamentPlanners.WhistEvenOddSAMDRRWhistPlannerTests, 131
TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.BIBDConstruction, 201
- ThreePlayerTournament
SmallBIBDForWhistConstruction, 202

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 208
 ConstructDesign, 209
 SmoothKTSConstruction, 208

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 22
 GDDForWhistConstruction, 90

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 183
 ConstructDesign, 185
 Resolvable3GDDConstruction, 184

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 203
 ConstructDesign, 204
 SmallGDDConstruction, 203

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 243
 ConstructDesign, 244
 WilsonGDDConstruction, 244

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 22
 GeneralPBD3NPlus1Construction, 99

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 110
 LargerOrderPBDConstruction, 111

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 166
 ConstructDesign, 167

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 176
 QuasigroupPBDConstruction, 177

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 205
 ConstructDesign, 206
 SmallOrderPBDConstruction, 206
 V, 206

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 23
 SingleTDConstruction, 197

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 187
 ConstructDesign, 189
 RTDConstruction, 188

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.
 218
 ConstructDesign, 219
 k, 220
 m, 220
 TDConstruction, 219

TournamentPlanner.Algorithms.ConstructionAlgorithms.IDesignConstruction.
 IDesign >, 101
 ConstructDesign, 102

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.
 23

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.
 108
 CommutativeIdempotentLatinSquareConstruction,
 ConstructDesign, 67

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 23

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 134
 ConstructDesign, 135

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 24
 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 24

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 47
 ArgumentExceptionMessage, 47
 BaseMOLSConstruction, 46
 ConstructDesign, 46

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 46
 Single, 47
 WilsonGDDConstruction, ValidateOrder, 46

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 74

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 76

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 96
 TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 96

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 127

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 212

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 218
 SmallOrderPBDConstruction, SOLSWithNoGCD6Construction, 212

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 196
 ConstructDesign, 197

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 24
 TournamentPlanner.Algorithms.Testing, 24

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 159
 ConstructPairwiseTestCases, 160
 PairwiseTestCasesConstruction, 159

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 25
 TournamentPlanner.Algorithms.TournamentPlanners.Balanced3PlayerTou
 39
 Balanced3PlayerTournamentPlanner, 40

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 62
 TournamentPlanner.Algorithms.TournamentPlanners.BipartiteTournament
 62
 BipartiteTournamentPlanner, 62
 PlanTournament, 63

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 105
 TournamentPlanner.Algorithms.TournamentPlanners.ITournamentPlanner
 BaseTournament >, 105

TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares.OA.
 108
 CommutativeIdempotentLatinSquareConstruction,

- TournamentPlanner.Model.Designs.BlockDesigns.RBIBD,
 - 179
 - AddParallelClass, 180
 - RBIBD, 180
- TournamentPlanner.Model.Designs.BlockDesigns.RTD,
 - 186
 - AddParallelClass, 187
 - RTD, 187
- TournamentPlanner.Model.Designs.BlockDesigns.STS,
 - 214
 - STS, 215
- TournamentPlanner.Model.Designs.BlockDesigns.TD,
 - 216
 - Groups, 217
 - K, 217
 - M, 218
 - TD, 217
- TournamentPlanner.Model.Designs.IDesign, 100
 - V, 101
- TournamentPlanner.Model.Designs.IResolvableDesign,
 - 104
 - ParallelClasses, 105
- TournamentPlanner.Model.Designs.LatinSquares, 28
- TournamentPlanner.Model.Designs.LatinSquares.LatinSquare,
 - 114
 - LatinSquare, 114
 - Matrix, 115
 - V, 115
- TournamentPlanner.Model.Designs.LatinSquares.MOLS,
 - 115
 - AddOrthogonalLatinSquare, 117
 - GetMOLSMatrices, 118
 - GetNumberOfMOLS, 118
 - LatinSquares, 118
 - MOLS, 116, 117
 - V, 118
- TournamentPlanner.Model.Designs.LatinSquares.SOLS,
 - 210
 - SOLS, 211
 - Transpose, 211
- TournamentPlanner.Model.Designs.OrthogonalArray,
 - 131
 - Height, 132
 - Matrix, 133
 - OrthogonalArray, 132
 - RepresentedMOLSCount, 132
 - V, 133
 - Width, 133
- TournamentPlanner.Model.Testing, 28
- TournamentPlanner.Model.Testing.PairwiseTestCases,
 - 157
 - PairwiseTestCases, 158
 - TestCases, 158
 - TranslatedTestCases, 158
 - VariableNames, 158
 - VariableValues, 159
- TournamentPlanner.Model.Tournaments, 28
- TournamentPlanner.Model.Tournaments.BaseTournament,
 - 49
 - AddRound, 50
 - BaseTournament, 50
 - GameToString, 51
 - Order, 51
 - Rounds, 51
- TournamentPlanner.Model.Tournaments.BipartiteTournament,
 - 59
 - BipartiteTournament, 60
- TournamentPlanner.Model.Tournaments.SAMDRR, 190
 - SAMDRR, 191
- TournamentPlanner.Model.Tournaments.ThreePlayerTournament,
 - 222
 - PlayerGroups, 223
 - ThreePlayerTournament, 223
 - TotalNumberOfGames, 223
- TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin,
 - 225
 - AddRoundSchedule, 226, 227
 - InactiveByRounds, 227
 - TwoPlayerRoundRobin, 226
- TournamentPlanner.Model.Tournaments.UniformGameSizeTournament,
 - 231
 - LeftOverPlayersByRounds, 233
 - ObjectiveValue, 233
 - TotalNumberOfGames, 232
 - UniformGameSizeTournament, 232
- TournamentPlanner.Model.Tournaments.WhistTournament,
 - 239
 - LeftOverByRounds, 240
 - WhistTournament, 239, 240
- TournamentPlanner.Pages, 29
- TournamentPlanner.Pages.ErrorModel, 78
- TournamentPlanner.Pages.IndexModel, 104
- TournamentPlanner.Pages.PageModels, 29
- TournamentPlanner.Pages.PageModels.BasePageModel,
 - 48
 - DisplayResults, 49
 - ErrorMessage, 49
- TournamentPlanner.Pages.PageModels.Designs, 29
 - Automatic, 30
 - BoseBlockBuilding, 30
 - CPApproach, 30
 - DifferenceApproach, 30
 - FourKOrder, 30
 - IncrementalBlockBuilding, 30
 - Method, 30
 - MOLSMethod, 30
 - OddOrderMOLS, 30
 - QuadraticProgramming, 30
 - SkolemBlockBuilding, 30
- TournamentPlanner.Pages.PageModels.Designs.BIBDModel,
 - 57
 - BIBD, 58
 - BlockBuildingMethod, 58
 - K, 58
 - L, 58
 - V, 59

- TournamentPlanner.Pages.PageModels.Designs.GDDModel,
 - 91
 - GDD, 92
 - M, 92
 - V, 93
- TournamentPlanner.Pages.PageModels.Designs.KTSMModel,
 - 108
 - KTS, 109
 - Multiplier, 109
 - V, 110
- TournamentPlanner.Pages.PageModels.Designs.MOLSMModel,
 - 119
 - MOLS, 120
 - MOLSConstructionMethod, 120
 - OA, 120
 - Order, 120
- TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_BIBD,
 - 140
- TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_GDD,
 - 141
 - ModelExpressionProvider, 141
- TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_PAGES,
 - 142
- TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_PAGES_Index,
 - 143
- TournamentPlanner.Pages.PageModels.Designs.Pages_PageModels_Designs_PAGES_PrivacyModel,
 - 144
- TournamentPlanner.Pages.PageModels.Designs.PBDModel,
 - 168
 - Multiplier, 168
 - PBD, 169
 - V, 169
- TournamentPlanner.Pages.PageModels.Testing, 30
- TournamentPlanner.Pages.PageModels.Testing.Pages_PageModels_Testing_PAGES_Testing,
 - 145
- TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingModel,
 - 161
 - PairwiseTestingModel, 161
 - TestCases, 162
 - VariableValues, 162
- TournamentPlanner.Pages.PageModels.Tournaments,
 - 31
- TournamentPlanner.Pages.PageModels.Tournaments.BipartiteTournament,
 - 60
 - TeamSize, 61
 - Tournament, 61
- TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_BipartiteTournament,
 - 146
- TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_ThreePlayerTournaments,
 - 147
- TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_TwoPlayerTournament,
 - 148
 - ModelExpressionProvider, 149
- TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_UniformGames,
 - 150
- TournamentPlanner.Pages.PageModels.Tournaments.Pages_PageModels_Tournaments_WhistTournaments,
 - 151
- TournamentPlanner.Pages.PageModels.Tournaments.ThreePlayerTournaments,
 - 151
- GDDTournament, 224
- KTSTournament, 225
- NbPlayers, 225
- TournamentPlanner.Pages.PageModels.Tournaments.TwoPlayerTournament,
 - 224
- NbPlayers, 230
- Tournament, 231
- TournamentPlanner.Pages.PageModels.Tournaments.UniformGamesModel,
 - 235
 - GameSize, 236
 - NbPlayers, 236
 - NbRounds, 236
 - TimeLimit, 236
 - Tournament, 236
- TournamentPlanner.Pages.PageModels.Tournaments.WhistTournaments,
 - 241
 - NbPlayers, 241
- Pages_PageModels_Designs_PAGES_ViewImports, 136
- Pages_PageModels_Designs_PAGES_ViewStart, 137
- Pages_PageModels_Designs_PAGES_Error, 138
- Pages_PageModels_Designs_PAGES_ModelExpressionProvider, 138
- Pages_PageModels_Designs_PAGES_Index, 139
- Pages_PageModels_Designs_PAGES_Privacy, 152
- Pages_PageModels_Designs_PAGES_PrivacyModel, 174
- Pages_Shared, 31
- Pages_Shared_Pages_Shared_Layout, 153
- Pages_Shared_ModelExpressionProvider, 153
- Pages_Shared_Pages_Shared_ValidationScriptsPart, 154
- Pages_Tournament, 31
- Pages_Tournament_BipartiteTournament, 155
- Pages_Tournament_TwoPlayerTournament, 156
- Pages_Tournament_ModelExpressionProvider, 156
- Pages_Tournament_Program, 174
- Pages_Tournament_Startup, 214
- Pages_Tournament_Utils, 31
- Pages_Tournament_Tests, 32
- Pages_Tournament_Tests_Algorithms, 32
- Pages_Tournament_Tests_Algorithms_ConstructionAlgorithms, 32
- Pages_Tournament_Tests_Algorithms_ConstructionAlgorithms_BlockDesigns, 32
- Pages_Tournament_Tests_Algorithms_ConstructionAlgorithms_BlockDesigns_BoseSTSCreationTest, 65
- Pages_Tournament_Tests_Algorithms_ConstructionAlgorithms_BlockDesigns_IncrementalBIBDConstructionTest, 99
- Pages_Tournament_Tests_Algorithms_ConstructionAlgorithms_BlockDesigns_ThreePlayerTournaments, 100
- Pages_Tournament_Tests_Algorithms_ConstructionAlgorithms_BlockDesigns_ThreePlayerTournaments_TwoPlayerTournament, 100

- LargerConstructionWithoutValidationTest, [95](#)
- LargerWhistConstructionTest, [95](#)
- SmallerWhistConstructionTest, [95](#)
- TournamentPlannerTests.Algorithms.TournamentPlanners.WholesaleRoundRobinWhistPlannerTests, [113](#)
- TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.EvenMooreWhistPlannerTests, [126](#)
- TestValidOrdersOfMooreConstruction, [127](#)
- TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.EvenSAMDRRWhistPlannerTests, [130](#)
- TestWhistConstructionUsingSAMDRR, [131](#)
- TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.EvenWhistPlannerTests, [221](#)
- TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.EvenWhistPlannerTests, [238](#)
- Whist152ConstructionTest, [238](#)
- TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd, [35](#)
- TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.OddCyclicOddWhistPlannerTests, [71](#)
- TestWhist33Construction, [71](#)
- TestWhist57Construction, [71](#)
- TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.OddPBDWhistPlannerTests, [170](#)
- Whist21ConstructionTest, [170](#)
- TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.OddPrimePowerWhistPlannerTests, [173](#)
- TestSeveralValidInputs, [173](#)
- TournamentPlannerTests.Datastore, [35](#)
- TournamentPlannerTests.Datastore.DataReaderTests, [73](#)
- TournamentPlannerTests.Designs, [36](#)
- TournamentPlannerTests.Designs.MOLSSetValidatorTests, [120](#)
- InvalidMOLSTest, [121](#)
- ValidMOLSTest, [121](#)
- TournamentPlannerTests.Model, [36](#)
- TournamentPlannerTests.Model.Designs, [36](#)
- TournamentPlannerTests.Model.Designs.BlockDesigns, [36](#)
- TournamentPlannerTests.Model.Designs.LatinSquares, [36](#)
- TournamentPlannerTests.Model.Tournaments, [37](#)
- TournamentPlannerTests.Model.Tournaments.WhistTournamentValidator, [242](#)
- ValidateWhistTournament, [242](#)
- TournamentPlannerTests.Utills, [37](#)
- TournamentPlannerTests.Utills.OrthogonalArrayMOLSCoverterTests, [135](#)
- MOLS2OATest, [135](#)
- OA2MOLSTest, [136](#)
- TournamentPlannerTests.Utills.PrimeFactorizationTests, [171](#)
- IsPrimeOrPrimePowerTest, [171](#)
- IsPrimeTest, [171](#)
- TranslatedTestCases
 - TournamentPlanner.Model.Testing.PairwiseTestCases, [158](#)
- Transpose
 - TournamentPlanner.Model.Designs.LatinSquares.SOLS, [211](#)
 - TournamentPlanner.Model.Tournaments.TwoPlayerRoundRobin, [220](#)
 - MooreWhistPlannerTests, [220](#)
 - TwoPlayerRoundRobinPlanner
 - TournamentPlanner.Algorithms.TournamentPlanners.TwoPlayerRoundRobinPlannerTests, [220](#)
- UniformGameSizeTournament
 - TournamentPlanner.Model.Tournaments.UniformGameSizeTournamentPlannerTests, [232](#)
 - UniformGameSizeTournamentPlanner
 - TournamentPlanner.Algorithms.TournamentPlanners.UniformGameSizeTournamentPlannerTests, [234](#)
- V
 - TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns, [43](#)
 - TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns, [206](#)
 - TournamentPlanner.Algorithms.TournamentPlanners.Whist.BaseWhistPlannerTests, [54](#)
 - TournamentPlanner.Model.Designs.BlockDesigns.BaseBlockDesign, [44](#)
 - TournamentPlanner.Model.Designs.IDesign, [101](#)
 - TournamentPlanner.Model.Designs.LatinSquares.LatinSquare, [115](#)
 - TournamentPlanner.Model.Designs.LatinSquares.MOLS, [118](#)
 - TournamentPlanner.Model.Designs.OrthogonalArray, [133](#)
 - TournamentPlanner.Pages.PageModels.Designs.BIBDModel, [59](#)
 - TournamentPlanner.Pages.PageModels.Designs.GDDModel, [93](#)
 - TournamentPlanner.Pages.PageModels.Designs.KTSMModel, [110](#)
 - TournamentPlanner.Pages.PageModels.Designs.PBDMModel, [169](#)
- ValidateOrder
 - TournamentPlanner.Algorithms.ConstructionAlgorithms.LatinSquares, [46](#)
- ValidateWhistTournament
 - TournamentPlannerTests.Model.Tournaments.WhistTournamentValidator, [242](#)
- ValidMOLSTest
 - TournamentPlannerTests.Designs.MOLSSetValidatorTests, [121](#)
- VariableNames
 - TournamentPlanner.Model.Testing.PairwiseTestCases, [158](#)
- VariableValues
 - TournamentPlanner.Model.Testing.PairwiseTestCases, [159](#)
 - TournamentPlanner.Pages.PageModels.Testing.PairwiseTestingMode, [162](#)

Whist

TournamentPlanner.Pages.PageModels.Tournaments.WhistTournamentsModel,
[242](#)

Whist152ConstructionTest

TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Even.Whist152PlannerTests,
[238](#)

Whist152Planner

TournamentPlanner.Algorithms.TournamentPlanners.Whist.Even.Whist152Planner,
[238](#)

Whist21ConstructionTest

TournamentPlannerTests.Algorithms.TournamentPlanners.Whist.Odd.PBDWhistPlannerTests,
[170](#)

WhistTournament

TournamentPlanner.Model.Tournaments.WhistTournament,
[239](#), [240](#)

Width

TournamentPlanner.Model.Designs.OrthogonalArray,
[133](#)

WilsonGDDConstruction

TournamentPlanner.Algorithms.ConstructionAlgorithms.BlockDesigns.GDDConstruction.WilsonGDDConstruction,
[244](#)