FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Posudek oponenta závěrečné práce

Student:	Jan Jindráček
Oponent práce:	Ing. Jan Trávníček, Ph.D.
Název práce:	Pattern matching in C11
Obor:	Webové a softwarové inženýrství

Datum vytvoření: 10. 6. 2019

Hodnotící kritérium:	*Způsob hodnocení – následující škálou 1 až 4:*
1. Splnění zadání	***1=zadání splněno,*** *2=zadání splněno s menšími výhradami,* *3=zadání splněno s většími výhradami,* *4=zadání nesplněno*

Popis kritéria:
Posuďte, zda předložená ZP dostatečně a v souladu se zadáním obsahově vymezuje cíle, správně je formuluje a v dostatečné kvalitě naplňuje. V komentáři uveďte body zadání, které nebyly splněny, posuďte závažnost, dopady a případně i příčiny jednotlivých nedostatků. Pokud zadání svou náročností vybočuje ze standardů pro daný typ práce nebo student případně vypracoval ZP nad rámec zadání, popište, jak se to projevilo na požadované kvalitě splnění zadání a jakým způsobem toto ovlivnilo výsledné hodnocení.

Komentář:
Cílem práce bylo implementovat rozšíření jazyka C ve variantě C11 o příkaz match dovolující pattern matching na struktuře, hodnotě nebo i hodnotě prvků instance datového typu. Příkaz match je známý z jiných programovacích jazyků. Cílem práce je také vytvořit dokumentaci k novému příkazu.

Hodnotící kritérium:	*Způsob hodnocení – bodové hodnocení 0 až 100 bodů (známka A až F):*
2. Písemná část práce	*62 (D)*

Popis kritéria:
Zhodnoťte přiměřenost rozsahu předložené ZP vzhledem k obsahu, tj. zda všechny části ZP jsou informačně bohaté a ZP neobsahuje zbytečné části. Dále posuďte, zda předložená ZP je po věcné stránce v pořádku, případně vyskytují-li se v práci věcné chyby nebo nepřesnosti. Zhodnoťte dále logickou strukturu ZP, návaznosti jednotlivých kapitol a pochopitelnost textu pro čtenáře. Posuďte správnost používání formálních zápisů obsažených v práci. Posuďte typografickou a jazykovou stránku ZP, viz Směrnice děkana č. 26/2017, článek 3. Posuďte, zda student využil a správně citoval relevantní zdroje. Ověřte, zda jsou všechny převzaté prvky řádně odlišeny od vlastních výsledků, zda nedošlo k porušení citační etiky a zda jsou bibliografické citace úplné a v souladu s citačními zvyklostmi a normami. Zhodnoťte, zda převzatý software a jiná autorská díla, byly v ZP použity v souladu s licenčními podmínkami.

Komentář:

Typografie:
Ukázky kódu nejsou vkládané do plovoucího prostředí, a text je pak nepřirozeně zalamovaný a nevyplňuje korektně stránku.
V práci jsou použity spojovníky místo symbolu ndash.
Práce využívá dopředných odkazů například na figuru 2.5 na konci kapitoly 1, sekci 2.5 na konci sekce 2.1.7.
Operátory porovnání jsou sázeny jako jedno dlouhé rovnítko.
V kapitole 2 je zbytečně mnoho nevyužitého volného prostoru vynecháno z důvodu chybného sázení ukázek textu.
V textu občas chybí členy.
Obrázky 3.1, 3.2 a 3.3 jsou nečitelné
Sekce 4.5 obsahuje chybně vysázený odkaz na kapitolu 1.

Faktické poznámky:
Český a anglický abstrakt si navzájem obsahově plně neodpovídají.
Sekce 2.1.4 vysvětluje vlatnost použitou už v příkladu 2.1.3 a měla by tedy být vysvětlena dříve.
Překlad třetího case ve figuře 2.5 obsahuje porovnání s hodnotou 0, i když pattern v match příkazu používal NULL.

Práce je netypicky strukturovaná: Kapitola 1 je rešerše příkazu match v jiných programovacích jazycích, ale představoval bych si v ní více příkladů, tak jako je obsahuje kapitola 2, nicméně bez navrhovaného překladu do čistého C. Navíc by do kapitoly rešerše měla patřit i část o možnostech rozšíření existujícího jazyka o nové příkazy, jako je v kapitole implementace a rozhodně do rešerše patří i obsah sekce 3.1.1, která zmiňuje rešerši podobné funcionality.

Vlastní návrh překladu match příkazu, jeho syntaxe a sémantiky jsou v kapitole návrh korektně. Sémantiku příkazu match bych ale definoval až po jeho syntaxi.

Čtenáři bych stejně tak mnohem dříve vysvětlil význam proměnných onePassedMatch a matchMe, překlad samostatného case bloku a především důvod vziku mnoha "&& true" z podmínek case bloků.

Proměnná onePassedMatch by nemusela existovat při využití návěští a příkazu goto.

Sekce 2.4 vůbec nediskutuje způsob získání typu z výrazu.

Poslední věta sekce 2.4.2 zmiňuje zajímavou vlastnost, ale ta není dále diskutována.

Nesouhlasím s tvrzeními v sekci 4, v bodovém seznamu, položce 4, 5 a 6, protože neuvažují break nebo návěští s příkazem goto.

Sekce 3.5 testování by měla být spíše součástí kapitoly 4 vyhodnocení.
Oceňuji množství příkladů příkazu match, ale očekával bych i podrobnější testování. Práce bohužel pouze zmiňuje porovnání efektivity match příkazu a ekvivalentu v čistém C na čtyřech výkonnostních testech bez uvedení grafů nebo alespoň statistického shrnutí výsledků. Věřím, že když se grafy vytvořeného výkonostního testování nevyskytují v práci (podle sekce 3.5.2), nebo nebyly provedeny (podle kapitoly 4), budou grafy uvedeny v prezentaci.

V příkladu 4.6 mě zaujal rozdíl v implementaci findPOPWithUDPPort, kde běžná verze využívá cyklus a verze s match příkazem využívá rekurzi, i když příkaz match může být uvnitř cyklu.

Hodnotící kritérium:	*Způsob hodnocení – bodové hodnocení 0 až 100 bodů (známka A až F):*
3. Nepísemná část, přílohy	*90 (A)*

Popis kritéria:
Dle charakteru práce se případně vyjádřete k nepísemné části ZP. Například: SW dílo – kvalita vytvořeného programu a vhodnost a přiměřenost technologií, které byly využité od vývoje až po nasazení. HW – funkční vzorek – použité technologie a nástroje, Výzkumná a experimentální práce – opakovatelnost experimentů

Komentář:

Implementaci hodnotím jako funkční, chybí mi příkaz break a podrobnější testování, které je dodané jen ve formě unit testů a pár testů výkonu.

Hodnotící kritérium:	*Způsob hodnocení – bodové hodnocení 0 až 100 bodů (známka A až F):*
4. Hodnocení výsledků, jejich využitelnost	*90 (A)*

Popis kritéria:
Dle charakteru práce zhodnoťte možnosti nasazení výsledků práce v praxi nebo uveďte, zda výsledky ZP rozšiřují již publikované známé výsledky nebo přinášející zcela nové poznatky.

Komentář:

Pro reálné použití bych rád viděl navíc rozšíření existujícího kompilátoru (například gcc) o navrženou funkcionalitu.

Hodnotící kritérium:	*Způsob hodnocení – nehodnotí se*

5. Otázky k obhajobě

Popis kritéria:
Uveďte případné dotazy, které by měl student zodpovědět při obhajobě ZP před komisí (body oddělte odrážkami).

Otázky:
Co by se stalo, kdyby guard condition použilo proměnnou, podle které se match provádí, a změnilo ji. Jak zareaguje implementace?

Příkaz match v programovacím jazyce Rust, ze kterého navrhovaný ekvivalent (částečně) vychází, neumožňuje použití příkazu break uvnitř těla case bloku. Rozšíření je ale přidáváno do jazyka C alespoň myšlenkově jako silnější příkaz switch, který break v příkazu case umožňuje. Jak by se dal do Vámi navrhované implementace příkazu match přidat příkaz break se stejnou sémantikou jako má v C?

Hodnotící kritérium:	*Způsob hodnocení – bodové hodnocení 0 až 100 bodů (známka A až F):*
6. Celkové hodnocení	*75 (C)*

Popis kritéria:
Shrňte stránky ZP, které nejvíce ovlivnily Vaše celkové hodnocení. Celkové hodnocení nemusí být aritmetickým průměrem či jinou hodnotou vypočtenou z hodnocení v předchozích jednotlivých kritériích. Obecně platí, že bezvadně splněné zadání je hodnoceno klasifikačním stupněm A.

Text hodnocení:
Středně náročné zadání, funkčně implementované. Celkový dojem kazí v některých místech neuspořádaný text práce a především testování, které je v textu shrnuto na půl stránky. Celkově hodnotím práci 75 body, tedy stupněm C (dobře).

Podpis oponenta práce: