# Implementační dokumentace k projektu do IPP 2017/2018

Jméno a příjmení: Josef Kadleček
Login: xkadle35

Duben 2018

## 1 Analyzátor kódu v IPPcode18

Skript nejprve vykoná nutné formality, jako je zpracování parametrů pomocí getopt(), zkontrolování hlavičky a vypsání hlavičky a dále načítá řádek po řádku instrukce. K usnadnění práce existuje globální mapa instrukcí, jež obsahuje jak jména instrukcí, tak počet a typ jejich parametrů. Každý řádek je předzpracován a zbaven komentářů, načež projde následujícími operacemi

1. Funkce instrValidate($instr) zkontroluje, jde li o validní instrukci, s požadovaným počtem parametrů.
2. Funkce paramValidate($instr) zkontroluje, že danné parametry jsou požadovaného typu.

Proběhne-li vše v pořádku, je na výstup vypsána XML reprezentace instrukce. V opačném případě je skript ukončen s příslušným chybovým hlášením a odpovídající návratovou hodnotou.

Pokud byly úspěšně zpracovány všechny instrukce, je vytištěna patička a případné bonusové statistiky o počtu řádků a komentářů v programu.

## 2 Interpret XML reprezentace kódu

Ve skriptu jsou připravené třídy:

1. Třída "variable", jež uchovává jméno, typ a hodnotu proměnné, pro ladící účely má přetíženou metodu __str__
2. Třída "instruction", uchovává argumenty instrukce, pro ladící účely má přetíženou metodu __str__ a metodu exec() jež danou instrukci vykoná.

Skript nejprve načte celý XML strom a zkontroluje jeho správnost. Sekvenčně pracuje s elementy instrukce,dělá z nich objekty třídy "instruction", a vkládá je do pole. Také provádí počáteční kontrolu kódu, kde zjistí případné chyby, a

poznačuje si pozici návěští.
Když je program předzpracován, sekvenčně se volá metoda exec jednotlivých instukcí. Tělo metody spočíva prakticky jen ve vhodném volání funkce: "globals()[self.opcode](self)" jež zavolá funkci stejného jména, jako je jméno intrukce a předá jí sama sebe.
Tento způsob je lehce redundantní (je zvlášť funkce pro ADD a SUB aj.) avšak umožňuje zvýšit čistotu a přehlednost kódu a v neposladní řadě snažší lazení instrukcí. Pokud instrukce skoku chce změnit čítač instrukcí, učiní tak přez globální proměnnou. Stejně jsou realizovány i zásobníky a rámce.

# 3 Testovací rámec

Skript, jež dle vstupních parametrů načte jména a cesty všech požadovanźch testů (všechny .src soubory, jež se nachází v parametry specifikovaných složkách) do pole testů. Poté je pro každý z testů spuštěna metoda, jež provede následující kroky:

1. Spustí se analyzátor kódu
2. Zkontrolují se návratové kódy, pokud jsou rozdílné, test selhal, pokud jsou totožné nenulové, test končí úspěchem
3. XML výstup je předán interpretu pomocí –source= a .in soubor je přesměrován na vstup.
4. Zkontrolují se návratové kódy, pokud jsou rozdílné, test selhal, pokud jsou totožné nenulové, test končí úspěchem
5. Pokud jsou totožné, nulové, provede se poslední fáze - diff výstupu interpretu a diff referenčnjího výstupu.

Stav testu se vypíše v jednoduchém html, kdy úspěšné testy jsou obarveny zeleně, neúspěšné rudě. Skript pokračuje, dokud nezprcoval všechny požadované testy.