Implementační dokumentace k 1. úloze do IPP 2022/2023
Jméno a příjmení: Ondřej Lukášek
Login: xlukas15

Začátek skriptu

Způsob řešení první úlohy je poměrně strohý. Nejdřív jsem potřeboval namyslet základní strukturu programu, tedy načíst zdrojový soubor (ze standardního vstupu) a vytvořit z něj XML reprezentaci. Pro vytvoření XML formátu výstupního souboru jsem se rozhodl použít XMLWriter, protože mi přišel nejlépe zdokumentovaný a velmi široce používaný, tudíž okolo něj bylo možné najít velké množství návodů. Nejprve jsem se rozhodl, že potřebuji zkontrolovat vložené argumenty, přičemž jediný možný argument je `--help`. Dále jsem načetl data ze standardního vstupu pomocí `$lines = file(INPUT);`, kde `INPUT` je `php://stdin`. Pokud se nepodařilo začít ze standardního vstupu číst, volá se chybový kód 11.

Hlavní smyčka

Po načtení vstupu začíná hlavní smyčka programu, která spočívá v odstranění komentářů, to se dělá pomocí regulárního výrazu `preg_replace("/#.*/", "", $lines[$i])` a příkazu `trim()`, který odstraňuje bílé znaky na začátku a konci každého řádku, kontrole vstupu, pomocí funkce `checkHeader()`, která zkontroluje správnost povinné hlavičky každého vstupu dle specifikace. Poté již následuje čtení jednotlivých řádků vstupu.

Kontrola řádků

Kontrola řádků začíná tím, že si každý řádek rozdělím na jednotlivé elementy pomocí `preg_split("\s+//", $lines[$i])` a následně je podle prvního elementu (identifikátoru instrukce) rozděluji pomocí switche do skupin podle kombinace operandů instrukce (těchto skupin je 8). Tento první element si ale nejdřív převedu pomocí `strtoupper()` na velká písmena, abych je mohl ve switchi lépe třídit. Pokud se stane, že první element řádku nebude patřit ani do jedné ze skupin, nejedná se o správně zadanou instrukci a skript končí s chybou 22. Pokud instrukce "zapadne" do nějaké ze skupin, zapíše se již do formátu XML pomocí funkce `instructionXML()`, která mi umožňuje snáze zapisovat instrukce do XML.
Dále probíhá kontrola operandů instrukce. Každý typ operandu se vyhodnocuje svou vlastní funkcí. <symb> se vyhodnocuje funkcí `checkSymb()`, <var> pomocí `checkVar()`, <const> funkcí `checkConst()`, <label> `checkLabel()` a nakonec <type> pomocí funkce `checkType()`. Všechny tyto funkce operují na podobném principu. Ověří se pomocí regulárního výrazu (ten je pro každý typ operandu jiný), který pokud nesouhlasí s operandem, volá se chyba 23. Například funkce `checkLabel()` ověřuje správnost zápisu návěští pomocí `preg_match("/^[a-zA-Z_\-$&%*!?][\w\-$&%*!?]*$/", $arg[$pos])`. Pokud je daný operand napsán správně, zapíše se do XML formátu k příslušné instrukci pomocí funkce `operandXML()`, která funguje prakticky stejně, jako již zmíněná funkce `instructionXML()`, jenom se zapisuje pořadí argumentu a jeho typ, protože instrukce může mít více argumentů a typů.

Tímto způsobem se u každého typu instrukce ověří správnost všech jejích operandů a jejich pořadí. Pokud byla instrukce přečtena správně, v XML dojde k jejímu ukončení (uzavření elementu `instruction`) a pokračuje se na zpracování dalšího řádku v dalším běhu hlavní smyčky.

Ukončení skriptu

Pokud již nejsou žádné další řádky, uzavře se XML element `program`, uzavře se dokument a obsah XML se vypíše na výstup pomocí `file_put_contents()`. Pokud se nepodaří výpis na standardní výstup, program končí s chybou 12. V případě úspěšného výpisu program končí s nulou.