Implementační dokumentace k 1. úloze do IPP 2022/2023

Jméno a příjmení: Ondřej Koumar

Login: xkouma02

Filozofie řešení

Moje řešení první části projektu spočívá v používání regulárních výrazů bez řízení konečným automatem. Pomocí RV v kódu řeším vše, co je zapotřebí, a tedy lexikální i syntaktickou správnost a jednoduché sémantické kontroly. Ze sémantických kontrol sem patří počet operandů instrukce a kontrola typu operandů. Kontrola typů v mém kódu je vypracovaná dle instrukční sady, kde pracuji se symbolem (proměnná/konstanta libovolného datového typu), návěštím, proměnnou a typem. Ovšem tyto sémantické kontroly se v mém kódu dají velmi jednoduše zpřísnit, a to prohozením jedné proměnné v regulárním výrazu. Například u RV k instrukci strlen je definován jako

```
^\s*(?i)strlen(?-i)\s+$variable\s+$symbol\s*$comment\s*$,
```

kde `$symbol` je `($variable|$intConstant|$boolConstant|$stringLiteral|$nil)`, z čehož například `$intConstant` je `(?-i)int@[-+]?\d+(?i)`. A je vidět, že `$symbol` by se dalo velmi jednoduše nahradit `($stringLiteral|$variable)` a tím tedy podporovat pouze řetězcové literály, případně proměnné v kombinaci s uvedenou instrukcí. Nakonec jsem se ale rozhodl pro benevolentnější řešení, protože hlavní funkce skriptu parse.php má být lexikální a syntaktická analýza, ne ta sémantická.

Implementační detaily

Parsing

Zásadní funkcí skriptu parse.php je funkce `parse($in)`. Ta provádí lexikální, syntaktické i sémantické kontroly. Všechny RV jsou uloženy v poli `$templates`, které slouží jako „šablony“, podle kterých se kontroluje zdrojový kód řádek po řádku. Na počátku vnějšího cyklu, který načítá soubor, dokud nenarazí na EOF, se kontroluje, zda není řádek prázdný, či na něm není pouze komentář. Pokud ano, načítá se další. Pokud ne, tak celá analýza řádku spočívá v jednom foreach cyklu, kde se pro každý načtený řádek projede pole šablon a kontroluje se, jestli daný řádek splňuje vzor daných šablon. Pokud ano, nastává druhá část skriptu, která jen zapisuje do výstupního XML.

Zapisování do XML

Hlavním mozkem této části skriptu je funkce `addToXML($line)`. V této funkci se nejdříve nahrazuje jakýkoli komentář na řádku (je-li přítomen) prázdným řetězcem. Poté se řádek rozdělí na několik částí oddělených bílými znaky. Ty části jsou instrukce a jejich operandy. Všechny tyto části jsou uloženy v poli `$output`, na prvním indexu je opcode a těch dalších, má-li je daná instrukce, operandy. První se do XML zapisuje opcode instrukce a její pořadové číslo (řešeno globální proměnnou a postupnou inkrementací ve funkci `parse($in)`).

Pro získání typu a hodnoty se používají dvě funkce. `myGetType($symb)` řeší typ operandu, který byl této funkci předán jako string. Buď vrátí rovnou tu samou hodnotu zpátky, pokud se jedná o návěští nebo typ, jinak rozdělí tento string na dvě části oddělené symbolem @. V případě výskytu rámců vrací typ proměnná, v případě klíčového slova pro datový typ vrací to klíčové slovo. `getValue($symb)` vrací hodnotu operandu. V případě proměnné, návěští nebo typu „type“ vrací rovnou předaný parametr zpátky, v případě konstanty si rozdělí string na dvě části stejně, jako funkce `myGetType($symb)` a vrátí hodnotu za symbolem @.