Implementační dokumentace k 2. úloze do IPP 2020/2021

Jméno a příjmení: Roman Stepaniuk

Login: xstepa64

Popis Interpret.py

- Práce s soubory a parse

Program začíná svou práce z kontroly argumentů pomocí třídy `agrparse` a otevírá soubor pro input, taky pomocí knihovny `xml.etree.ElementTree` reprezentuje soubor –source ve stromovém tvaru. V připadě špatných argumentů program vrací chybu 10. V případě nevalidných souboru program se ukončí chybou 11.

- Syntaktická analýza

Program začíná syntaktickou analýzu z kontroly validnosti a celistvosti XML souboru. V připadě poškození XML program se ukončí chybou 31. Pak se začinají provadět syntaktické kontroly na dodřování podminek jazyka IPPcode21. Syntaktické kontroly se provádí pomoci `if else` podminek, ciklů a pomocí funkce `checkChild()`, `varCheck()`, `symbCheck()`, `typeCheck()` a `labelCheck()`. Špatný syntax se ukončuje chybovym kodem 32.

- Vytvoření instrukce

Pro další pohodlnou interpretací bylo rozhodnuto použit třídu `Instruction`, která má obsahuje jmeno, id a taky obsahuje seznam Argumentů. Argument reprezentuje literal v programu, který umišťuje hodnoty `value` a `type`. Takže reprezentace samotných argumentů je implementovana pomoci třidy `Argument`. Program prochází XML stromem a přidává instrukce do seznamu pro další implementace. Během zápisu instrukce se provádí funkce `remove_escape_sequences()`, která transformuje escape sekvencí ve stringu do ASCII symbolů.

- Reprezentace programu jazyka IPPcode21

Pro pohodlnější a srozumitelnou reprezentace programu byla vytvořena třida `Program`, která simuluje strukturu programu jazyka IPPcode21. Toto je základní třida pro interpret, ona má tří druhů ramců: GF, TF, LF. Přičemž jeden GF, jeden TF a seznam LF. Pro simulace samotného ramce byla vytvořena třida `Frame`, která v sobě umišťuje seznam proměnných. Proměnnu reprezentuje třída Variable. Každa proměnna má jméno, hodnotu a typ. Každá z těchto tříd má implementované metody pro práce s proměnnými, `setVarValue`(), `getVarValue`() a t.d., třídá `Program` má metody pro práce s ramcy pro simulace instrukci CREATEFRAME, PUSHFRAME, POPFRAME a taky podporuje simulaci DEFVAR, MOVE, takže simuluje celý ten datový model programu.
Do jakého ramce bude zapsana proměnna program rozhoduje podle jmena proměnné (GF@ a t.d.) pomocí regulárních výrazů.

- Interpretovaní instrukce

Hlavní rozhodující třída programu je `Interpret`, do které byl zapsán seznam instrukci a která pomoci využití třídy `Program` interpretuje všechny instrukce. Interpret taký obsahuje dva

zasobnika: datový zasobnik a zásobník volání. Po dokončení zápisu instrukci se vyvolá metoda `checkLabelsandJumps`(), která kontroluje, jestli všechny skoky a navěšti v programu jsou validné. Metoda `run`() mistí ciklus a řadu `if else` podminek a vyvolává ostatní funkci třídy `Interpret`, které ty samotné instrukci interpretují. Instukci v metodě run se přepínají pomoci funkce `getNextOperation`(). Při chybách interpretaci program se ukončuje odpovídajícím chybným kodem. V případě dokončení interpretace poslední instrukci program se uspěšně ukončuje kodem 0.