Implementační dokumentace k 2. úloze do IPP 2022/2023

Jméno a příjmení: Adam Kala

Login: xkalaa00

Návrh

Tento projekt byl navržen s využitím objektově-orientovaného programování, které sloužilo jako základ celého programu. Tato metoda byla aplikována na instrukce, argumenty, funkce, kontrolu XML a otevírání vstupních souborů. Kromě toho byla vytvořena třída pro proměnné, která je uložena v jiném souboru, aby bylo možné přistupovat ke všem statickým metodám spojených s argumenty stejným způsobem, například k **setValue()**. Téměř celý projekt se dá pomocí jednotlivých tříd přesunou, upravit, nebo případně dopsat rozšíření.

Skript

Samotný projekt byl strukturován tak, aby byl snadno srozumitelný a nebyly příliš velké skoky při práci s jednotlivými instrukcemi. Na začátku kódu byly inicializovány proměnné a seznamy například pro rámce, s nimiž se poté bude pracovat. Dále následují dvě funkce, jedna pro výpis chybových hlášek a druhá pro ukončení programu příslušným návratovým kódem a vypsání chybové hlášky na **stderr**.

Třída pro všechny funkce byla vytvořena tak, aby na začátku zjistila počet argumentů a prováděla jednotlivé metody na základě této informace. Některé metody byly spojeny dohromady, například **arithmetic(),** která obsahuje instrukce ADD, SUB, MUL a IDIV. Tato metoda je vytvořena za účelem zjednodušení skriptu, protože tyto instrukce mají společné požadavky, až na IDIV, kde musíme počítat s dělením 0, a lze je tedy sdružit.

Zkontrolování argumentů z příkazového řádku probíhá ve třídě **ParseFile**, společně s inciliazací "stromu", provedeno pomocí knihovny **xml.etree.ElementTree.**

Třída **CheckXML()** slouží k ověření správnosti argumentů spojených s XML a zároveň zkontroluje, zda se číslo argumentu neopakuje. Tudíž pokud má například vstup arg1, arg2 a arg2, tato metoda to odhalí a ukončí program chybou.

Pořadí instrukcí bylo seřazeno mezi voláním těchto dvou tříd. Konkrétně přes funkci sorted podle argumentu "order".

Další část kódu obsahuje cyklus for, který prochází všechna návěští a ukládá je, stejně jako argumenty podle názvu instrukce. Také zde kontroluje její počet argumentů, aby odpovídal požadovanému počtu.

Poslední cyklus, který je v kódu umístěn, je while cyklus, jenž prochází všechny uložené instrukce a podle názvu instrukce provádí danou funkci. Tento proces je ukončen ve chvíli, kdy dorazí na poslední instrukci. Pokud se dostaví opcode, který neexistuje, vyvolá se chyba 32. Po dokončení cyklu se program úspěšně ukončí.

Třída Functions

Společně s metodami je vytvořena tak, aby podporovala všechny možné počty argumentů, od žádných až po 3. Po přiřazení těchto hodnot se třída přesune do zadané metody a vykoná jednotlivé úkony. Buď hodnotu uloží přes již zmíněnou funkci **setValue(),** vrátí ji přímo přes return nebo narazí na chybu a ukončí se.

Rozšíření

Ačkoliv rozšíření nebyla implementována, jejich přidání by bylo velmi jednoduché. Do while cyklu na konci skriptu by se dala podmínka například pro JUMPIFEQS a v ní by se program přesunul do funkce přes **funct.JUMPIFEQS(),** kde by vykonal potřebné kroky pro dosažení požadovaného výsledku.

UML Diagram tříd