Implementační dokumentace k 2. úloze do IPP 2020/2021

Jméno a příjmení: Jaroslav Kvasnička
Login: xkvasn14

Interpret.py

Interpret se skládá z několi hlavních a pomocných funkcí. Mezi hlavní funkce patří `xmlFunc`, `instrCheck` a `Interpretation` a k pomocných funkcím náleží `getVar` a `setVar`.

Pomocné funkce

Tyto pomocné funkce prohledávají jednotlivé rámce podle jmen proměnných a patřičně reagují na chybové stavy. Pokud je u `getVar` vše v pořádku a proměnná existuje, vrátí její hodnotu i s typem. Funkce `setVar` uskuteční vyhledání proměnné, a po úspěšném nalezení do ní zapíše hodnotu s typem.

Hlavní funkce

Tělo programu začíná načtením argumentů z příkazové řádky a určením jejich pravdivosti. Zjistí se také jestli cesty k souborům jsou skutečné a pokud ano, načtou se z nich data.

Pomocí `xml.etree.ElementTree` dostaneme jednotlivé Instrukce, které kontrolujeme v `xmlFunc`. Po detailní kontrole hlavičky, správnosti argumentů a typů, se instrukce seřadí a přiřadí se jim nové očíslování, pro lepší zpracovatelnost. Funkce `instrCheck` slouží jako druhá kontrola počtu argumentů pro každou funkci. Tím se zjistí jestli každá funkce má přesný počet argumentů pro zpracování. V `Interpretation` se každá funkce převede a provede.

Všechny funkce i návěští jsou číslovány a skoky jsou řešeny rekurzivním voláním funkce `Interpretation` s hodnotami návěští na které se má skok provést.

Globální rámec a dočasný rámec jsou řešeny pomocí `dictionaries` a lokální rámec je seznam. Stejným způsobem jsou postaveny datový stack, stack pro definované návěští, nebo stack pro volané návěští pouze pro funkce call a return.

Seznam instrukcí je velká tabulka s 8-prvkovými seznamy.

Test.php

Je skrip spouštějící zvlášť parse.php a interpret.php a vypisuje statistiky do tabulky ve formě HTML. Pro zjišťování cest k souborům byly použity funkce `real.path()` a `RecursiveDirectoryIterator` a `Recursiveregexiterator`. Tyto funkce slouží k zjištění všech podadresářů s jejich soubory, které jsou uloženy do pole, které se postupně parsuje a zpracovává

Tester se dělí na 3 základní části a to jsou Parser argumentů, printer tabulky a samotný tester. Parser argumentů si rozděluje jednotlivé argumenty a podle toho zařídí průběh zpracovávání testů.

Printer tabulky vytiskne šablonu, do které se postupně naplňují jednotlivé testy po jejich dokončení.

V testeru dochází k rozhodování mezi parse.php nebo interpret.py nebo jejich kombinací. Nejprve si ošetří všechny nalezené a nenalezené soubory, poté spustí test, a nakonec ho vyhodnotí a výsledek zapíše v html do tabulky.

Dodatek

Kód v test.php by se určitě dal implementovat pomocí funkcí a tím jej výrazně zmenšit. Také by se daly použít jiné prohledávací metody jako například `scandir()` a další. Základ tabulky byl generován na stránce:
`https://www.quackit.com/html/html_table_generator.cfm`