Implementační dokumentace k 2. úloze do IPP 2020/2021
Jméno a příjmení: Adam Marhefka
Login: xmarhe01

Implementácia druhej úlohy sa skladá zo skriptu `interpret.py` a skriptu `test.php`. Skript `interpret.py` je napísaný v jazyku python 3.8.5 a skript `test.php` v jazyku php7.4.

Úlohou skriptu je zabezpečiť kontrolu správnosti vstupného xml súboru, ktorý obsahuje xml reprezentáciu kódu v jazyku IPPcode21 odpovedajúcu výstupu skriptu `parse.php` z prvej úlohy. Skript sa začína kontrolou argumentov, tá je uskutočnená pomocou série podmienok a listu `sys.argv` z modulu `sys`, v ktorom sa nachádzajú jednotlivé argumenty zadané skriptu pri spustení cez príkazový riadok. Na načítanie xml súboru bol použitý modul `xml.etree.ElementTree`. Pomocoue funkcie `xml.etree.ElementTree.parse` sa načíta strom xml súboru a zároveň sa kontroluje správnosť formátu, využitím konštrukcie `try-except` (pri chybnom formáte funkcia končí s výnimkou).

Ďalej sa v skriptne kontroluje či štruktúra xml súboru odpovedá zadanej forme ktorú by mal generovať skript `parse.php` z prvej úlohy. Táto kontrola prebieha v cykle a pre kontrolu správnosti formátu jednotlivých literálov a názvov premenných je použitá funkcia na kontrolu regulárnych výrazov `re.match` z modulu `re`. Pri kontrole štruktúry sa do slovníka ukladá ako kľúč poradie inštrukcie a ako hodnota je uložený ďalší vnorený slovník obsahujúci poradie argumentu inštrukcie ako kľúč a samotný argument ako hodnotu. Ďalej sa všetky inštrukcie ukladajú do listu a hodnota `order` sa ukladá do ďalšieho listu. Náveštia sa ukladajú ako zanorený list spolu s ich hodnotou atribútu `order` do listu `labels`. Hodnoty atribútu argumentov `type` sa taktiež ukladajú do listu pre neskoršie porovnanie počas interpretácie. Po kontrole štruktúry dôjde ku usporiadaniu listov a slovníku podľa listu obsahujúceho hodnoty atribútov `order` a následnému prečíslovaniu týchto hodnôt na súvislú postupnosť čísel pre jednoduchú indexáciu v cykle. Ešte predtým dochádza ku viacerým kontrolám, napr. či sú hodnoty atribútov `order` v povolenom rozsahu, kontrola duplicity týchto hodnôt, kontrola duplicity náveští, či odpovedajú dĺžky listov a podobne.

Po preusporiadaní a kontrole pomocných datových štruktúr sa začína samotná interpretácia inštrukcií. Globálny rámec, datový zásobník, zásobník volaní a zásobník rámcov je každí inicializovaný ako prázdny list, zatiaľ čo dočasný a lokálny rámec sú inicializované na hodnotu `None`, ktorá v implementácii reprezentuje nedefinovanosť. Samotná interpretácia potom prebieha v cykle, v ktorom sa najprv pomocou konštrukcie podmienok určí aká inštrukcia sa práve implementuje (inštrukcie sú už zoradené a ich poradie je zmenené pre jednoduchú indexáciu pomocou iteračnej premennej). Premenné sú v implementácii reprezentované listom skládajúcim sa z štyroch hodnôt. Prvá reprezentuje názov premennej, ktorý je uložený bez úvodných znakov reprezentujúcich rámec (tzn. neukladá sa identifikátor rámca `GF@, LF@` a podobne). Druhá hodnota reprezentuje typ premennej alebo literálu vo forme reťazca kde je jednoducho vypísaný daný typ. Tretia hodnota je samotná hodnota premennej alebo literálu vo forme odpovedajúcej ukladanému datovému typu, s vínimkou typu `nil` ktorý je uložený ako reťazec. Štvrtá hodnota je nakoniec reťazec o dĺžke jedného znaku reprezentujúci rámec v ktorom sa premenná nachádza, jeho hodnoty môžu byť G, L a T. Každá z týchto hodnôt je v prípade chýbajúcej hodnoty nastavená na hodnotu `None`, čo výrazne zjednodušuje kontrolu definovanosti premenných a typovú kontrolu. V podmienkach implementujúcich samotné inštrukcie sa potom s argumentom inštrukcie volá funkcia `matchfind` a jej varianty pre rozličné typy literálov. Táto funkcia určí či je argument premenná alebo literál. V prípade literálu vráti vyššie spomínané pole štyroch hodnôt, s názvom a rámcovým identifikátorom inicializovaným na hodnotu `None` a hodnotu reprezentujúcu typ a samotnú hodnotu literálu uloži ako 2. a 3. prvok listu. Na zistenie typu je opäť použitá funkcia `re.match`. V prípade premennej je prehľadaný príslušný rámec, v tejto funkcii taktiež dôjde ku odstráneniu úvodného identifikátoru rámca (`GF@`). Ak nie je premenná v správnom rámci nájdená alebo literál neodpovedá žiadnému regulárnemu výrazu, funkcia vracia v štvorprvkový list hodnôt `None`. Po návrate s funkcie sa pokračuje v interpretácii, kontroluje sa či bola premenná nájdená, správnosť typov atď.. Ak sú splnené všetky formálne požiadavky vykoná sa interpretácia funkcionality inštrukcie. Takto sa pokračuje kým nie sú interpretované všetky inštrukcie, kedy program končí s návratovým kódom 0, alebo kým skript nenarazí na chybu kedy sa ukončí s príslušným chybovým kódom.

Skript `test.php` implementujúci testovací rámec pre skripty `parse.php` a `interpret.py` sa začína kontrolou argumentov skriptu a nastavením príslušných prepínacích premenných typu `bool`. Po kontrole argumentov sa získa pole názvov súborov z funkcie `get_sort_files` prípadne `get_sort_files_rec` pri špecifikovaní argumentu `--recursive`. Obidve funkcie vrátia pole s názvami súborov v špecifikovanom priečinku. Následne sa v cykle prechádza každý súbor a pri nájdení súboru s koncovkou `.src` sa získajú z rovnomenných súborov s koncovkami `.rc` a `.out` príslušné údaje a uložia sa do polí pre očakávané návratové kódy a očakávané výstupy. Následne sa pustí príslušný skript (podľa vstupných konzolových argumentov pre skirpt) kde sa na vstup skriptu ako argument vloží buď súbor so zdrojovým kódom IPPcode21 (v prípade testovania skriptu `parse.php` alebo testovania oboch skriptov postupne) alebo súbor s xml reprezentáciou tohto kódu (pri testovaní s argumentom `--int-only` súbor s koncovkou `.src` a pri testovaní oboch skriptov je použitý výstup zo skriptu `parse.php`).

Výstupy a návratové kódy sú opäť uložené do polí a následne porovnávané s očakávanými návratovímy kódmi a výstupmi. Vyhodnotenia týchto porovnaní sú ukladané do ďalšieho poľa ako hodnota typu `bool` kde hodnota False

reprezentuje zlyhanie testu a hodnota True reprezentuje úspešný test. Nakoniec je v cykle podľa výsledkov testov generovaný html výstup. Ten sa vypisuje priamo na štandardný výstup.