Implementační dokumentace k 2. úloze do IPP 2021/2022
Jméno a příjmení: Juraj Hatala
Login: xhatal01

Interpret

Interpret sa skladá z niekoľkých modulov s rôynzmi funkciami.

Pamäťový model

V module `interpret_memory.py` sa nachádza implementácia pamäťového modelu. Konkrétne je tu objekt definovaná trieda `memory_frames`, v ktorej sa nachádzaju premenné reprezuntujúce rámce definované v zadaní a metódy, ktoré s nimi pracujú, ako napr. metóda pre inicializáciu premennej na zadanom rámci, `def_var(frame, name)`, či prístup k jej hodnote, `get_varValue(frame, name)`. V objekte je taktiež implementovaný zásobník z rozšírenia. V `interpret.py` je potom vytvorený objekt `memory_frames` s názvom `memory` do ktorého sa ukladajú všetky dáta interpretu.

XML kontrola

Modul `interpret_XML_handler.py` slúži čisto pre zachytávanie chýb vstupného XML súboru, teda zachytávanie chyby `32`. Náchádza sa tu statický slovník `instr_dic`, ktorý obsahuje informácie o obsahu jednotlivých instrukcí na vstupe. Tento slovník sa následne využíva v jedinej metóde modulu, `check_XML(child)`. Táto metóda kontroluje správnosť štruktúry zadanej inštrukcie teda napr. ci instrukcia obsahuje order a opcode, ci su zadáne správne argumenty pre konkrétnu inštrukciu napr. `JUMPIFNEQ : label, symbol, symbol` a či sú argumenty správne označené. Metóda taktiež argumenty zoradí pre jednoduchšie spracovanie.

Metódy interpretu

`interpret_functions.py` je najobsiahlejší modul projektu a obsahuje metódy využívané v `interpret.py`. Sú tu metódy pracujúce s XML formátom ako `get_symbolValue(instruction,memory)` či metódy kontrolujúce rôzne chybné vstupy napr. `check_symbol_type(symbol, memory, type)`, ktorá kontroluje ci zapisany typ symbolu odpovedá hodnote symbolu a očakávanému typu predaného argumentom. Tieto metódy sa následne využívajú pri samotnej implementácii inštrukcií, napr. v metódach ako `perform_arithmetics(...)` , ktorá podľa mena zadanej inštrukcie vyberie zo slovníka `_operators` operátor odpovedajúci danej inštrukcii, pomocou ktorého spracuje argumenty inštrukcie a výsledok uloží do premennej danej inštrukciou. Podobne fungujú aj metódy

`perform_comparsion(...)`, `perform_logical_operations(...)` a `perform_jumps(...)`, kde každá táto metóda má svoju alternatívu pre operácie na zásobníku z rozšírenia, napr. `perform_jumps_stack(...)`, ktorá pracuje s hodnotami na zásobníku, namiesto hodnôt uložených v rámcoch a XML atribútoch.

Skoky

Skoky sú možné vďaka cyklu, spustenému pred hlavný cyklom vykonávajúcim inštrukcie, ktorý vyhladá všetky náveštia ktoré obsahuje zadaná XML štruktúra a uloží ich názvy a pozície do slovníku `labels`. Z tohoto slovníka majú inštrukcie skokov informácie potrebné pre ich priebeh.

Testovací rámec

Testovanie je riešenév jedinom skripte test.php, ktorý obsahuje všetky potrebné funkie i samotnú implementáciu. Zpracovanie argumentou príkazovej riadky zabezpečuje funcia `handle_arguments()`, ktorá pomocnými funkciami argumenty zparsuje a zkontroluje validitu ich zápisu pripadne spustí help. Podľa argumentov sa následne hladajú potrebné súbory v zadaných adresárových štruktúrach. Scripty a jexam súbory sa ukladajú do priradených premenných a testovacie súbory sa ukladajú do pola `$all_test_files`, v ktorom sa rozdelujú do polí src, in , out a rc. Následne sa vytvoria počítadlá testov a html hlavička a spustí sa hlavný cyklus.

Cyklus prechádza polom `$all_test_files["src"]`, každému `.src` súboru priradí podľa mena opovedajúce `.in` `.out` a `.rc` súbory a uloží ich do pola `$current_test`. Cyklus následne podľa argumentov zistuje či sa jedná o int-onl, alebo parse-only testovanie, alebo o súčastné testovanie oboch scriptov. Každý script sa spúšta funkciou exec, ktorá uloží výstupy do dočasných súborov a návratové hodnoty do premenných, a následne zisťuje či sa jedná o test správnosti výstupu alebo test chybnej návratovej hodnoty. Po vyhodnotení testu sa zavolajú funkcie `test_success` , alebo `test_failed`, ktoré vypisujú výsledky na štandardný výstup v html formáte.

Pri spustení testovania oboch scriptov sa nekontroluje `parse.php` výstup jexam nástrojom, pretože podľa zadania nie je možnosť pre piaty testovací súbor `test.out2` z ktorým by nástroj výstup porovnal, kontroluje sa teda iba výstup `interpret.py`.