Implementační dokumentace k 2. úloze do IPP 2021/2022
Jméno a příjmení: Vladislav Mikheda
Login: xmikhe00

1 Implementace skriptu interpret.py

1.1 Popis

Skript se načítá ze souboru nebo ze standardního vstupu, kód v jazyce `IPPcode22`je reprezentovaný ve XML formátu. Výstupem je provádění načteného programu nebo chybné hlášení.

1.2 Struktura

Po spuštění skriptu `interpret.py` bude provedena kontrola vstupních argumentů, po kontrole bud' skript bude ukončen chybným hlášením, nebo bude provedeno načtení a analyzovaní XML vstupu.
Načtení bude provedeno ve třídě `Parse.py` s využitím modelu `xml.etree.ElementTree` a uložené do pole ze kterého pak budou postupně přebírané jednotlivé instrukce s argumenty. Takže ve třídě pomocí metody `sort` bude prováděno třídění jednotlivých instrukcí podle `opkodu`, a také začáteční sémantická kontrola, například kontrola opakování `labelu`, nebo záporného `opcodu`.
Dál bude spuštěn hlavní cyklus programu který standardně se ukončí po dvou průchodech. V prvním proběhne kontrolování redefinice proměnných a ostatní základní sémantické kontroly, ve druhem proběhne vyplňování pole které obsahují: název instrukce a její argumenty. Dál pole bude předáno do třídy `Executor.py`. Cyklus muže mít víc než dva běhy kvůli skokovým instrukcím.
Ve třídě `Executor.py` bude prováděné vykonání jednotlivých instrukcí. Ve třídě jsou dvě základních metody:jedna pro instrukce se skokem, druhá pro instrukce bez skoku. Metody pomáhají zjistit které metody pro tuto instrukci musí být zavolaný. Pak už prochází volání jednotlivých metod.

1.3 Datová model

Pro uchování dat byly zavedené třídy: `DataElement` která reprezentuje nejednoduší strukturu datové položky, a třída `Variables` která reprezentuje proměnu. Rámci byli reprezentovaný pomoci pole.

1.4 Skokový instrukce

Při implementaci skokových instrukcí bude znovu sestaveno pole ze kterého čerpáme instrukci, na začátku kterého bude instrukce která obsahuje label na které byl prováděn skok. A hlavní cyklus bude prodloužen ještě na jeden běh.

1.5 Ukončeni programu

Když interpretace programu dojde do poslední instrukce a to bude druhý nebo další běh hlavního cyklu programu, program bude končit s návratovým kódem: `0`.
Takže program muže být ukončen během interpretace, a vrátí chybný návratový kód a nějaké hlášení aby bylo možné zjistit kde a jaká byla chýba kvůli které program spadl.

2 Implementace skriptu test.php

2.1 Popis

Skript provádí testování skriptů `parse.php` a `interpret.php`. Testování může byt prováděno jeden po druhém nebo zároveň. Výstupem je HTML kód který obsahuje výsledky jednotlivých testů.

2.2 Struktura

Ve třídě `ArgumentParse.php` se prohází analýza zadaných argumentu. Po analýze a získání adresářů ve kterých jsou testovací soubory bude prováděno zjištění cesty k jednotlivým souborům v daném adresáři, nebo rekurzivní ctění pod adresáři a také zjištění cesty. Pro rekurzivní prohlíženi byla využita třída `RecursiveIteratorIterato` pro nerekurzivní načtení `IteratorIterator`. Dál byla provedena filtrace souboru podle názvu, odfiltrujeme soubory které končí `.src` pomoci třídy `RegexIterator`.
Dále bude provedena kontrola dostupnosti a existence dalších souborů pro testy případně jejich vygenerovaní. A cesty do `.src` souborů budou uloženy do pole a budou setříděny podle abecedy.
Ve třídě `TestParserInter.php` je prováděna kontrola cest k souborům skriptů a pomocných souborů. A bude spuštěn cyklus ve kterém bude z pole načtená jednotlivá cesta k souboru který bude otestován skriptem parse.php nebo interpret.py případně dvěma. A jednotlivé výsledky pomocí třídy `HTMLwriter` zapsané v HTML formátu do bufferu.

2.3 Testovaní jednotlivých skriptu

Pro testování jen jednoho skriptu ze dvou na začátku bude spuštěn skript na vstupu kterého bude testovací soubor, zatím budou porovnany jen výsledné kódy, když kódy jsou nulové bude porovnán výsledný soubor s výstupem ze skriptu a testový soubor `.out` pak výsledky porovnaní zapsaný v HTML formáte.

2.4 Postupné tetovaní

Pro testování dvou skriptu postupně, na začátku bude spuštěn skript parse.php a výsledek bude uchován do souboru, bude porovnán jen kód zakončení a už pak bude spuštěn skript interpret.php na vstupu kterého bude soubor s výsledným kódem, vystup interpreta bude otestován pomocí unixového nástroje `diff`.

2.5 HTML

Po skončení testování bude vygenerován kód ve formátu HTML ve kterém je představen celkový počet testů, počet úspěchů a počet neúspěchu, a také procento kolik testu dopadli úspěšně. a jednotlivé výsledky každého testu.