IMPLEMENTAČNÁ DOKUMENTÁCIA KU 2. PROJEKTU DO IPP (interpret.php) 2023/2024

Meno a Priezvisko: **Boris Vícena**
Login: **xvicen10**

Úvod

Táto dokumentácia poskytuje stručný prehľad implementácie skriptu `interpret.php`, ktorý funguje ako interpret pre imperatívny jazyk IPPcode24. Interpret, načíta zdrojový kód vo formáte XML, vykoná jeho spracovanie a interpretáciu a vypíše výsledky na štandartný výstup. Tento interpret dokáže spracovať a vykonávať rôzne inštrukcie definované v špecifikácií IPPcode24, čím umožňuje interpretáciu programov v danom jazyku.

Návrh a interná reprezentácia

Cieľom bolo vytvoriť interpret, ktorý bude schopný efektívne spracovať a interpretovať kód jazyka IPPcode24. Hlavný tok interpretácie je riadený pomocou triedy `Interpreter`, ktorá zabezpečuje správne načítanie vstupných argumentov, spracovanie súborov, inicializáciu potrebných prostriedkov a volanie potrebných funkcií na interpretáciu programu. Trieda `Runner` slúži ako sprostredkovateľ medzi triedami `Instructions` a `Interpreter`, pričom jej úlohou je načítať a triediť inštrukcie zo zdrojového kódu (XML súbor). V metóde `prerun()` prebehne príprava a následne sa vykonáva interpretácia inštrukcií v metóde `run()`.

Spôsob riešenia

Spracovanie argumentov: Interpret spracováva predané argumenty a nastavuje prostredie pre beh programu.
Načítanie a validácia XML súboru: XML súbor je načítaný a validovaný pre zabezpečenie správnej štruktúry.
Načítanie inštrukcií: Inštrukcie sú načítané zo vstupného XML súboru a zoradené podľa požadovaného poradia order.
Pre-run pre značky (LABELS): Pred vykonaním inštrukcií sa spracujú značky(*Labels*), ktoré sú následne využité pri vykonávaní inštrukcií.
Vykonávanie inštrukcií: Inštrukcie sú postupne vykonávanie pomocou triedy `Runner`, ktorá následne volá triedu `Instructions` pre spracovanie jednotlivých inštrukcií podľa operačného kódu (*opcode*).

Trieda Interpreter

Táto trieda je hlavnou súčasťou interpretátora a je zodpovedná za vykonanie interpretácie zdrojového programu. Slúži na spracovanie argumentov, validáciu XML súboru a následne spustenie interpretačného behu.

- **Spracovanie argumentov:** Vytvára inštanciu objektu `Settings` a spracováva argumenty programu.
- **Získavanie zdrojového kódu a vstupných údajov:** Pomocou objektu `Settings` získava zdrojový kód, vstupné údaje, štandartný výstup (*stdout*) a štandartný chybový výstup (*stderr*).
- **Ošetrenie prípadu nápovedy:** Kontroluje, či bol pri spustení použitý argument o zobrazenie nápovedy a v prípade, že bol, vypisuje nápovedu na štandartný výstup a ukončuje interpretáciu.
- **Validácia XML štruktúry:** Vytvára inštanciu objektu `XMLValidator` a overuje správnosť štruktúry XML dokumentu.
- **Spustenie interpretačného behu:** Vytvára inštanciu objektu `Runner` a spúšťa vykonávanie inštrukcií.

Trieda Runner

Táto trieda riadi beh vykonávania inštrukcií zo zdrojového kódu. Slúži na načítanie a zoradenie inštrukcií podľa ich poradia a následné spustenie interpretácie.

- **Príprava inštrukcií:** Inicializuje objekt `Instructions` s využitím vstupných a výstupných objektov a ukazovateľa na inštrukcie (*instructionPointer*).
- **Načítanie a zoradenia inštrukcií z XML súboru:** Získa koreňový element DOM dokumentu a následne načíta všetky inštrukcie. Následne inštrukcie zoradí podľa atribútu `order`.
- **Predbežné spracovanie značiek (Prerun pre LABELS):** Vykoná predbežné spracovanie značiek a získa značky z DOM dokumentu a pridá ich do zoznamu značiek (*Labels*).
- **Vykonanie inštrukcií:** Postupne vykonáva všetky inštrukcie zoradené podľa ich poradia. Pri každom kroku získava a vykonáva aktuálnu inštrukciu z objektu `Instructions` až do konca zoznamu inštrukcií.

Trieda Frames
Táto trieda je zodpovedná za správu rámcov (*Frames*), v ktorých sa ukladajú premenné počas interpretácie programu.

- **Pridávanie premenných:** Poskytuje metódu `add()`, ktorá umožňuje pridávať premenné do jednotlivých rámcov (*globalFrame, localFrame, tempFrame*).
- **Nastavovanie hodnôt premenných:** Metóda `set()` umožňuje nastavovať hodnoty premenných v jednotlivých rámcov podľa ich názvu.
- **Získavanie hodnôt premenných:** Pomocou metódy `get()` je možné získať *hodnotu* alebo *typ* premennej z daného rámca.
- **Manipulácia s rámcom:** Poskytuje metódy na manipuláciu s rámcom vrátane pushovania nového rámca pomocou metódy `pushFrame()` a popovania rámca pomocou metódy `popFrame()`.

Trieda Labels
Táto trieda sa zaoberá prácou so značkami (*Labels*), ktoré sa používajú na riadenie toku programu počas jeho interpretácie.

- **Pridávanie značiek:** Poskytuje metódu `add()`, ktorá umožňuje pridávať značky do zoznamu značiek.
- **Hľadanie indexu značky:** Obsahuje metódu `findLabelIndex()`, ktorá nájde index značky v poli inštrukcií podľa jej názvu.
- **Riešenie značiek:** Metóda `resolveLabels()` sa používa na získanie poľa značiek v zadanom DOM dokumente pre ďalšie použitie pri vykonávaní jednotlivých inštrukcií.

Trieda Instructions
Táto trieda je *srdce* interpretátora a je zodpovedná za vykonávanie jednotlivých inštrukcií podľa operačného kódu (*opcode*) a príslušných argumentov. Je zodpovedná za správne prevedenie kódu a manipuláciu s dátami v súlade s ich vykonávaním.

- **Interpretácia inštrukcií:** Poskytuje metódu `run()`, ktorá vykonáva aktuálnu inštrukciu podľa operačného kódu (*opcode*) a príslušných argumentov inštrukcie.
- **Prístup k prostriedkom interpretácie:** Má prístup k vstupu, výstupu, ukazovateľu inštrukcií, zásobníku dát a skokov, ktoré využíva pri vykonávaní jednotlivých inštrukcií.
- **Definície jednotlivých inštrukcií:** Trieda `Instructions` obsahuje metódy pre každú podporovanú inštrukciu. Tieto metódy sú zodpovedné za správne vykonanie danej inštrukcie na základe jej operačného kódu a argumentov. Každá metóda implementuje logiku konkrétnej inštrukcie a zabezpečuje, že je vykonaná správne a s požadovanými efektmi.

Chybové stavy a zachytávanie výnimiek v triede Interpreter
Zachytávanie výnimiek: V triede `Interpreter` sú zachytávané rôzne typy výnimiek, ktoré môžu nastať počas behu interpretácie.
Vyhodenie výnimky: V prípade zachytenia výnimky je tá istá výnimka opätovne vyvolaná pomocou *throw*. Toto zabezpečuje preposlanie presnej chybovej správy aj s kontextom vykonávania programu.
Chybové stavy a ich správy:

- *InvalidStructure:* Výnimka vyvolaná pri neplatnej štruktúre XML dokumentu.
- *ValueError:* Výnimka vyvolaná pri neplatnej hodnote premennej alebo argumentu inštrukcie.
- *VariableAccessError:* Výnimka vyvolaná pri pokuse o prístup k neexistujúcej premennej alebo pri neplatnom prístupe ku premennej.
- *FrameAccessError:* Výnimka vyvolaná v prípade chyby prístupu k rámcom.
- *StringOperationError:* Výnimka vyvolaná pri chybách operácií s reťazcami.
- *SemanticError:* Výnimka vyvolaná pri sémantických chybách v programe.
- *OperandStructure:* Výnimka vyvolaná pri chybách štruktúry operandov.
- *OperandValue:* Výnimka vyvolaná pri chybách hodnoty operandov.

UML Diagram tried