Implementační dokumentace k 2.úloze do IPP2022/2023

Jméno a příjmení: Martin Priessnitz

Login: xpries01

Implementace skriptu interpret.py

Skript *interpret.py* čte ze souboru nebo ze standardního vstupu. Výstupem je provedení samotného programu, případně vypsaní na standardní výstup. Chybová hlášení jsou vypisována na standardní výstup.

Struktura programu

Úlohu jsem řešil pomocí OOP. Vytvořil jsem si třidy na hlavní části programu (class Interpret, class Program, class Instruction, class Argument, class Frame a class Memory).

Class Program

Class Program definuje vnitřní reprezentaci načteného zdrojového kódu. Konstruktor vyžaduje kořenový element zdrojové XML struktury. Obsahuje jedinou proměnnou self.ins, což je seznam instrukcí programu. Provadí nezbytný preprocesing např.: kontrolu orderu a seřazení atd. . Parsování vnitřních XML elementů deleguje na třídu Instruction. Je to instance třídy Instruction.

Class Instruction

Class Instruction reprezentuje jednu instrukci ze zdrojového kódu. Ověřuje a načítá atributy elementu instruction. Parsuje vnitřní elementy a deleguje je na třídu Argument. Obsahuje self.order, opcode(všechni možné typy instrukcí) a arg(což je seznam argumentů). Je to instance třídy Argument

Class Argument

Class Argument kontroluje typy argumentů.

Class Frame

Class Frame reprezentuje jeden frame v paměti. Proměnné si ukládá formou slovníku, kde klíč je název proměnné a hodnota je hodnota proměnné.

Class Memory

*Class Mem*ory reprezentuje paměťový model program. Spolu s instancí třídy *Interpret* jednoznačně určuje aktuální stav výpočtu. Obsahuje všechny framy(*GF*, *TF* a zásobník *LF*).

Class Interpret

Top-level třída, která definuje vnější rozhraní interpretu. V konstruktoru projde program a načte všechna návěští. Interpretace se spouští skrz funkci *run()*.

Interpret

inputFile labels : dict memory pc : int program

add(ins) andd(ins) breakk(ins) call(ins) concat(ins) createFrame(ins) defVar(ins) dprint(ins)
eexit(ins) eq(ins) getchar(ins) gt(ins) idiv(ins) int2char(ins) jump(ins) jumpifeq(ins) jumpifneq(ins) label(ins) lt(ins) move(ins) mul(ins) nott(ins) orr(ins)
popFrame(ins)
pops(ins)
pushFrame(ins) pushs(ins) read(ins) returnn(ins) run()
runInstruction() setchar(ins) stri2int(ins) strlen(ins) sub(ins) typee(ins) write(ins)

memory

Memory

callStack : list dataStack : list globalFrame localFrames : list

temporaryFrame : NoneType

defVar(varName) get(varName) set(varName, value)

globalFrame temporaryFrame

Argument

argNum : int name type types : list

Frame

vars : dict

defVar(name)
exists(name)
get(name)
set(name, value)

Instruction

arg : list opcode opcodes : list order : int

Program ins : list