Implementační dokumentace k 1. úloze do IPP 2023/2024

Jméno a příjmení: Maksim Samusevich
Login: xsamus00

Úvod

Tento skript je implementací parseru pro jazyk IPPcode24 v jazyce Python. Skript načítá instrukce IPPcode24 ze standardního vstupu, kontroluje syntaxi a generuje odpovídající XML reprezentaci instrukcí na standardní výstup.

Implementace

Nejprve program provádí kontrolu argumentů předaných skrze příkazový řádek a pokud byl poskytnut aspoň jeden argument, zjišťuje se, zda tento argument je `"--help"`. Pokud je argument `"--help"`, program zavolá funkci `print_help()`, která vypíše nápovědu, a následně ukončí své vykonávání s návratovým kódem 0.

Dále nasleduje zpracování vstupního obsahu ze standardního vstupu a rozdělení ho na jednotlivé řádky. Při zpracování vstupního obsahu program ignoruje prázdné řádky a komentáře začínající znakem # a jakmile je nalezen neprázdný a nekomentovaný řádek, probíhá kontrola hlavičky.

Každý řádek je rozdělen na jednotlivé tokeny podle mezer, první token z každého řádku se ukládá do proměnné `instruction`, na základe které probíhá kontrola syntaxe na tomto řádku. Zkontroluje se, zda má aktuální instrukce správný počet tokenů a zda tyto tokeny odpovídají správnému formátu.

V programu jsou použity dvě třídy `Parser` a `XMLWriter`, což zlepšuje čitelnost a snadnost údržby kódu.

Parser

Třída Parser poskytuje sadu metod pro analýzu a ověření syntaxe instrukcí a tokenů v jazyce IPPcode23. Zde je popis jednotlivých částí třídy:

`check_tokens()`: Statická metoda, která ověřuje, zda je správný počet argumentů pro instrukci.

`check_var()`: Statická metoda, která ověřuje správnost syntaxe proměnného symbolu pomocí regulárního výrazu. Kontroluje, zda proměnný symbol odpovídá definovanému formátu, který zahrnuje cestu k proměnnému rámci (LF, TF, GF) a následně název proměnné.

`check_label()`: Metoda, která využívá regulární výraz k ověření syntaxe značky (label). Tento regulární výraz kontroluje, zda značka obsahuje pouze povolené znaky, jako jsou písmena, číslice a určité speciální znaky.

`check_symbol()`: Metoda, která kontroluje platnost a syntaxi symbolu, který je vstupním argumentem. Rozděluje symbol na typ a hodnotu a poté provádí specifické kontroly pro různé typy symbolů, včetně celočíselných, booleovských, řetězcových a nilových hodnot. Pokud symbol nevyhovuje požadavkům syntaxe nebo platnosti, metoda vyvolá chybu a ukončí běh programu.

`check_type()`: Metoda, která kontroluje platnost typu symbolu. Pokud symbol není jedním z povolených typů (”int”, ”bool”, ”string”, ”nil”), metoda vyvolá chybu a ukončí běh programu.

XMLWriter (NVP rozšíření)

Třída XMLWriter slouží k vytváření XML reprezentace instrukcí programu a byla udělána podle návrhového vzoru `továrna` . Zde je popis jednotlivých metod:

`__init__(self)()`: Konstruktor třídy XMLWriter, který inicializuje kořenový element root s názvem ”program” a jazykem ”IPPcode24”.

`start_instruction()`: Tovární metoda, která začíná vytvářet novou instrukci v XML.

`finish_instruction()`: Tovární metoda, která dokončuje vytváření jedné instrukce v XML.

`write_argument()`: Tovární metoda, která přidává argument do vytvářeného XML. Rozhoduje o typu argumentu na základě jeho obsahu a názvu instrukce, a poté vytváří odpovídající XML element s příslušnými vlastnostmi a hodnotou argumentu. Nakonec přidá tento element jako součást aktuální instrukce v XML struktuře.

`finish_xml()`: Tovární metoda, která ukončí XML dokument.

`prettify()`: Statická metoda, která formátuje XML reprezentaci tak, aby byl lépe čitelná.