Implementačná dokumentácia k 1. úlohe do IPP 2021/2022
Meno a priezvisko: Peter Ďurica
Login: xduric05

1 Návrh riešenia

Parser postupne načítava jednotlivé riadky zo zdrojového súboru. Prvý riadok by mal vždy obsahovať hlavičku s názvom jazyka, v ktorom je napísaný kód. Ak je hlavička správne zapísaná, parser začína analyzovať zvyšok programu. Ak sa v riadku nachádza komentár, tak ho odstráni a pracuje len s nezakomentovanou časťou. Túto časť rozdelí na jednotlivé tokeny. Prvý token v riadku by mal byť operačný kód príkazu a teda ho porovnáva so známymi operačnými kódmi. Ak nenájde zhodu tak zdrojový kód obsahuje chybu a nemôže ho analyzovať. Ak je zhoda nájdená skontroluje argumenty príkazu s očakávanými argumentami podľa pravidiel jazyka. Po úspešnej kontrole si parser inštrukciu zaznamená a prechádza na ďalší riadok. Inštrukcie sú následne vypísané až po úspešnej analýze celého zdrojového kódu.

2 Pomocné funkcie

2.1 Funkcie na upravenie kódu

Riadok kódu najprv rozdelím na dve časti podľa prvého výskytu znaku “#“. Časť po mriežke zahodím a pracujem ďalej len s prvou časťou.

Pri ďalšom kroku rozdelím riadok bez komentárov podľa znaku medzery “ “. Výsledné pole reťazcov následne pošlem cez funkciu, ktorá odstráni prázdne polia a potom cez ďalšiu, ktorá polia nanovo preindexuje. S takto upraveným riadkom môžem začať analýzu kódu.

2.2 Funkcie na kontrolu správnosti zadaných argumentov

Argumenty zväčša kontrolujem oproti regulárnym výrazom. Funkcie na porovnanie premennej, typu, návestia len porovnajú kontrolovaný reťazec a vrátia logickú jednotku alebo nulu ak sedia s regulárnym výrazom.

Funkcia na kontrolu symbolov má však viac návratových typov a teda viem zistiť či je symbol typu “string“, “bool“, “int“, “nill“ alebo je to názov premennej.

3 Generovanie XML kódu

3.1 Použité nástroje

Na generovanie XML kódu som použil objekt DOMDocument, ktorý umožňuje postupné priraďovanie elementov do elementového stromu a využíva sa v ňom objektové programovanie. Zaisťuje správne formátovanie výstupného XML kódu a prišiel mi ako správne riešenie otázky generovania výstupu. XML štruktúra nevypisuje počas behu programu ale ukladá sa do objektu dokumentu, ktorý viem následne na konci programu vytlačiť do výstupu.

3.2 Funkcie na automatické generovanie inštrukcií a argumentov

Vytvoril som si funkcie na generovanie inštrukcií a argumentov pre XML kód. Funkcia na generovanie inštrukcie berie názov operačného kódu a jeho poradové číslo a vygeneruje element DOCDocumentu, s ktorým môžem následne pracovať.

Funkcia na generovanie argumentu vytvára element DOCDocumentu podľa chceného typu. Ak argument operačného kódu je nesprávne zapísaný podľa pravidiel jazyka, tak funkcia hádže chybu. Takto vytvorený argument viem pridať k inštrukcii a tú následne pridať do programového stromu.

4 Kontrola operačných kódov

Na kontrolu operačných kódov slúži switch do ktorého je posielané prvé slovo z riadku, ak riadok neobsahuje slovo tak sa preskakuje. Prvé slovo je ešte predtým poslané do funkcie, ktorá ho prepíše len na veľké písmená. Tak sa zaistí nerozlišovanie veľkosti písma. Prípady výsledkov, ktoré majú rovnaký počet a typ argumentov sú zlúčené spolu pre lepší prehľad a skrátenie kódu. Pre každý správne napísaný operačný kód sa vygeneruje inštrukcia aj s prípadnými argumentmi. Ak je poslaný neznámy operačný kód parser hlási chybu v zdrojovom kóde.