Dokumentace k projektu IFJ a IAL

**Implementace překladače imperativního jazyka IFJ19**

Tým 127, varianta I

**Jiří Žák (xzakji02) 25%**

Ivan Halomi (xhalom) 25%

Adam Ševčík (xsevci) 25%

Martin Hiner (xhiner) 25%

1. Úvod

Cílem tohoto projektu bylo vytvořit překladač, který načte zdrojový kód v jazyce IFJ19, který je zjednodušenou podmnožinou jazyka Python 3, a přeloží ho do mezikódu IFJcode19. Program je konzolová aplikace, která vrací buď výsledný kód, nebo případně odpovídající chybový kód. Program načítá ze standartního vstupu a vrací na standartní výstup.

1. Návrh a implementace

Jednotlivé modely boli paralelne vyvíjané každým členom týmu, podľa rozdelenia.

* 1. Lexikální analyzátor

Implementácia lexikálního analizátora sa nachádza v súbore scanner.c a využíva pomocnú knihovnu string.c pre jednoduchšiu prácu so stringom.

Hlavná funkcia lexikálního analyzátora je get\_token(), ktorá je implementácia konečného automatu podľa grafu vytvořeného na základe zadania. Funkcia vracia ukazatel na objekt typu tToken, ktorý reprezentuje jednu lexému za zadaného kódu. Sú v ňom uložené informácie ako riadok, na ktorom sa token nachádza, obsah a typ tokenu, prípadne podtyp, podla požiadaviek ostatních modulov. Jednotlivé stavy automatu sa nachádzajú v type enum tState, ktorý sa ale pre jednoduchosť zároveň využíva jako označenie typov a podtypov tokenov. Funkcia je implementovaná jako nekonečný switch, ktorý znak po znaku prechádza štandardný vstup a podla zadaných pravidel ho delí na jednotlivé tokeny.

Jazyk IFJ19 využíva odsadenie riadkov na zoskupenie príkazov do sekvencie, preto bolo pre kontrolu nutné využiť pomocný zásobník typu LIFO. Změna odsadenia sa prejaví odoslaním tokenov sIndent alebo s sDedent, ktoré nahradzujú zložené zátvorky známe z jazyka C.

V súbore sa ešte nachádzajú pomocné funkcie init\_token() na inicializáciiu tokenu, unget\_token() na vrátenie tokenu na štandardný vstup a assign\_type() na zistinie, či sa je v tokene uložené klúčové slovo alebo identifikátor. Taktiež sa tu nachádzajú pomocné funkcie pre prácu so zásobníkom.

* 1. Syntaktický analyzátor
  2. Sémantický analyzátor
  3. Generování kódu
  4. Makefile

V požadavcích projektu bylo přiložit soubor Makefile, který přeloží projekt s příkazem make. V souboru jsou nastavená pravidla, podle kterých se má projekt překládat. Projekt se překládá překladačem gcc s nastavenýma flagama. Ze souborů s příponou .c se vytvoří objektové soubory s příponou .o a z těch se poté vytvoří jeden spustitelný soubor s názvem main.

Nástroj Makefile jsme také použili pro testování a mazání dočasných souborů.

1. Speciální použité techniky a algoritmy

Pro projekt jsme implementovali několik speciálních datových struktur.

* 1. Tabulka pomocí binárního vyhledávacího stromu

Vytvořili jsme tabulku pomocí binárního vyhledávacího stromu, která slouží jako tabulka symbolů, což souvisí s předmětem IAL ve kterém jsme se o tom učili. Implementovali jsme nezbytné funkce pro práci s touto tabulkou jako inicializace, přidání nové položky, odstranění určité položky, vyhledávání a odstranění celé tabulky z paměti.

* 1. String
  2. Lifo

1. Práce v týmu
   1. Způsob

Nejdříve jsme si projekt podle částí rozdělili a poté jsme na něm postupně pracovali. Na různých částech pracoval většinou každý sám s případnou pomocí ostatních.

* 1. Git

Jako verzovací systém jsme použili Git a jako vzdálený repositář Github. Každý z nás zde měl svou větev, do které přidával svou část projektu. Poté jsme vše dávali do jedné hlavní a postupně vše sestavovali.

* 1. Komunikace

Komunikace probíhala většinou osobně nebo prostřednictví messengeru. Občas jsme se sešli, abychom řešili problémy na různých částí projektu.

* 1. Rozdělení práce

Práci jsme si rozdělili rovnoměrně s ohledem na složitost. Každý z týmu tedy dostal hodnocení 25%.

Jiří Žák - syntaktický a sémantický analyzátor, testování, dokumentace

Martin Hiner - lexikální analyzátor, testování

Ivan Halomi - syntaktický a sémantický analyzátor, testování

Adam Ševčík - generování kódu, testování

1. Závěr

Projekt byl pro nás určitě zatím nejnáročnější za naši dobu působení na této fakultě. V průběhu vývoje jsme se občas potýkali s nějakými neshodami, které jsme museli řešit laděním a testováním. Tento projekt nám určitě dal hodně znalostí z látek probíraných na přednáškách IFJ a IAL a přinesl nám zkušenost s prací na větších projektech.

Obrázok, na ktorom je mapa, text

Automaticky generovaný popis