Електротехнички факултет - Универзитет у Београду

A gold and red emblem with a red circle

Description automatically generated

Програмски преводиоци 1

Компајлер за Микројаву

Студент:

Манојло Пековић

2019/0504

Професор:

проф. Др Драган Бојић

Београд, јун 2024.

**Кратак опис поставке задатка**

Циљ пројектног задатка је реализација компајлера за програмски језик Микројава. Компајлер омогућава превођење синтаксно и семантички исправних Микројава програма у Микројава бајткод који се извршава на виртуелној машини за Микројаву. Уз синтаксно и семантички исправан код, омогућена је и исправка грешака у унапред одређеним ситуацијама. Синтаксно и семантички исправни Микројава програми су дефинисани спецификацијом коју је могуће пронаћи на сајту предмета у одељку Домаћи задатак.

Програмски преводилац за Микројаву има четири основне функционалности: лексичку анализу, синтаксну анализу, семантичку анализу и генерисање кода.

**Лексички анализатор** треба да препознаје језичке лексеме и врати скуп токена издвојених из изворног кода, који се даље разматрају у оквиру синтаксне анализе. Уколико се током лексичке анализе детектује лексичка грешка, потребно је исписати одговарајућу поруку на излаз.

**Синтаксни анализатор** има задатак да утврди да ли издвојени токени из изворног кода програма могу формирају граматички исправне сентенце. Током парсирања Микројава програма потребно је на одговарајући начин омогућити и праћење самог процеса парсирања на начин који ће бити у наставку документа детаљно описан. Након парсирања синтаксно исправних Микројава програма потребно је обавестити корисника о успешности парсирања. Уколико изворни код има синтаксне грешке, потребно је издати адекватно објашњење о детектованој синтаксној грешци, извршити опоравак и наставити парсирање.

**Семантички анализатор** се формира на основу апстрактног синтаксног стабла које је настало као резултат синтаксне анализе. Семантичка анализа се спроводи имплементацијом метода за посећивање чворова апстрактног синтаксног стабла. Стабло је формирано на основу граматике имплементиране у претходној фази. Уколико изворни код има семантичке грешке, потребно је приказати адекватну поруку о детектованој семантичкој грешци.

**Генератор кода** преводи синтаксно и семантички исправне програме у извршни облик за одабрано извршно окружење Микројава ВМ. Генерисање кода се имплементира на сличан начин као и семантичка анализа, имплементацијом метода које посећују чворове.

**Покретање програма**

Преко билд фајла:

* Таргет за брисање: Брише претходно генерисане фајлове приликом новог покретања компајлера.
* Генератор лексера: Користећи библиотеку JFlex.jar и улазни фајл lexer.lex креира лексер(YYlex.java)
* Генератор парсера: Користећи cup\_v10k.jar и parser.cup фајл, генерише парсер помоћу генератора синтаксних аналижатора АСТ-ЦУП.
* Препакивање: Преправљање назива пакета
* Компајлирање: Покретање процеса компајлирања
* Дисасемблирање, дебуг и рун: Користе се ради провере извршавања кода на виртуелној машини.

Из командне линије:

1. Генерисање лексера:
   * java -cp ../lib/JFlex.jar JFlex.Main -d rs/ac/bg/etf/pp1 ../spec/mjlexer.flex
2. Генерисање парсера:
   * java -cp ../lib/cup\_v10k.jar java\_cup.Main -destdir rs/ac/bg/etf/pp1 -ast rs.ac.bg.etf.pp1.ast -parser MJParser -dump\_states -buildtree ../spec/mjparser.cup
3. Компајлирање компајлера:
   * javac -cp ../lib/\*;. rs/ac/bg/etf/pp1/ast/\*.java rs/ac/bg/etf/pp1/util/\*.java rs/ac/bg/etf/pp1/\*.java
4. Компајлирање микроЈава програма:
   * java -cp "../lib/\*;../bin/" rs.ac.bg.etf.pp1.Compiler ../test/test301.mj ../test/program.obj
5. Декомпајлирање микроЈава програма:
   * java -cp ../lib/mj-runtime-1.1.jar rs.etf.pp1.mj.runtime.disasm ../test/program.obj
6. Извршавање објектног фајла са преусмеравањем стандардног излаза и излаза грешке:
   * java -cp ../lib/mj-runtime-1.1.jar rs.etf.pp1.mj.runtime.Run < ../input.txt ../test/program.obj >../test/program.out 2>../test/program.err

**Опис тест примера:**

* **Error\_handling:** Тест у ком су изазване све грешке које треба да се обраде у парсеру.
* **Test\_if\_else:** Тест у ком се позива више if\_else конструкција и врши се испис у свакој од њих. Такође су дати примери семантичких грешака у вези ових конструкција
* **Test\_list\_comp:** Тест у ком се позивају методе везане за копирање листе, као и семантичке грешке у вези овог конструкта.
* **Test\_print\_meth:** Тест који позива принтове низа и метода. Такође даје примере семантичких грешака везаних за испис и методе.