|  |  |
| --- | --- |
| etf_logo_2010_cir | **УНИВЕРЗИТЕТ У БЕОГРАДУ**  **ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ**  Програмски преводиоци 1 – 2019/2020.  Булевар краља Александра 73, ПФ 35-54, 11120 Београд, Србија  телефон: 011/3218-321, [dekanat@etf.bg.ac.rs](mailto:dekanat@etf.bg.ac.rs) |

ПРОЈЕКТНИ ЗАДАТАК

***КОМПАЈЛЕР ЗА МИКРОЈАВУ***

Програмски преводиоци 1

Студент:

Лукић Тања, 2016/0626

Београд, јануар 2020.

Садржај

[О пројектном задатку 3](#_Toc29682049)

[Опис команди за генерисање јава кода алатима, превођење кода компајлером и покретање 3](#_Toc29682050)

[Опис новоуведених класа 4](#_Toc29682051)

[Опис тест примера 4](#_Toc29682052)

# О пројектном задатку

Израда решења пројектног задатка урађена је под оперативним системом Windows у програмском језику Java. Циљ пројектног задатка је рализација компајлера за програмски језик Микројаву. Компајлер омогућава превођење синтаксно и семантички исправних Микројава програма у Микројава бајткод који се извршава на виртуелној машини за Микројаву. Да ли је код синтаксно и семантички исправан одређује се на основу спецификације која је дефинисана поставком пројектног задатка.

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

Лексички анализатор треба да препознаје језичке лексеме и да врати скуп токена изднојених из изворног кода, који се даље разматрају у оквиру синтаксне анализе.

Синтаксни анализатор има задатак да утврди да ли издојени токени могу да формирају граматички исправне сентенце. Као резултат добија се апстаракно синтаксно стабло, на основу граматике која се имплементира у овој фази.

Семантички анализатор се формира на основу добијеног апстарактног синтаксног стабла. Спроводи се имплементацијом метода за посећивање чворова у оквиру који се испитује одговарајућа семантичка правила.

Генератор кода преводи синтаксно и семантички исправне програме у извршни облик за одабрано извршно окружење Микројава ВМ.

# Опис команди за генерисање јава кода алатима, превођење кода компајлером и покретање

За покретање кода користи се класа Run из библиотеке mj-runtime-1.1.jar и као аргумент јој се прослеђује путања до одговарајућег .obj фајла који се тестира. Да би се генерисао потребан .obj фајл користи се додатно имплементирана класа Compiler. Пре покретање наведене класе неоходно је покренути build.xml који ће уз помоћ алата (JFlex, AST-CUP) изгенерисати неопходне јава класе, које уз помоћ осталих, додатно имплементираних класа покрећу цео код.

build.xml се састоји од циљева који међусобно могу али не морају бити зависни, у оквиру којих се позивају претходно наведини алата и дефинише место на којем треба да се смести резултат њиховог извршавања.

# Опис новоуведених класа

Новоуведене класе Compiler, MJParserTest, MJParserTestSintaksa и МЈTest слиже за тестирање генерисања кода, семантичке анализе, синтаксне анализе и лексичке анализе респективно.

Поред њих као нове класе јављају се CodeGenerator и SemanticAnalyzer у оквиру којих се преклапањем visit метода одговарајућих чворава врши генерисање кода и семантичка анализа.

# Опис тест примера

* test2.mj

Користи се при провери синтаксне фазе, у оквиру њега користе се смене које обухватају највећи део граматике, укључујући класе, методе, петље и условна гранања

* test3.mj

Користи се за проверу семантичке фазе, избацује грешке у случају некомпатибилних типова у аритметичким изразима, приликом доделе вредности. Такође проверава типове података у read и print наредбама и не дозвољава извођење из класе која пре тога није дефинисана, не дозвољава инстанцирање апстрактне класе.

* test1.mj

Користи се за проверу фазе генерисања кода. Обухвата смене које су наведе за дати ниво.