**Универзитет у Београду**

**Електротехнички факултет**

Београд, Булевар краља Александра 73

**Пројекат из предмета Системски Софтвер**

**ДВОПРОЛАЗНИ АСЕМБЛЕР**

Професор: Студент:

Саша Стојановић Малешевић Ђорђе, 0399/2015

Београд, јун 2019. год.

С А Д Р Ж А Ј

**Страна**

[1. Задатак 3](#_Toc522799741)

[2. Анализа проблема 4](#_Toc522799742)

[3. Решење 5](#_Toc522799743)

[Табела симбола 6](#_Toc522799744)

[Табела секција 7](#_Toc522799745)

[Асемблер 8](#_Toc522799746)

[Релокациони запис 9](#_Toc522799747)

[Инструкције 10](#_Toc522799748)

[4. Тестирање Пројекта 13](#_Toc522799749)

[Улаз/излаз 13](#_Toc522799750)

# 1. Задатак

Napisati dvoprolazni asembler za procesor opisan u prilogu. Svi potrebni argumenti asemblera zadaju se kroz komandnu liniju prilikom pokretanja. Ulaz asemblera je tekstualni fajl. Za potrebe leksičke analize ulaznog tekstualnog fajla dozvoljeno je koristiti generatore leksera. Izlaz asemblera treba da bude predmetni program zapisan u tekstualnom fajlu. Za potrebe učitavanja u emulator dozvoljeno je generisati i binarni fajl pored tekstualnog. Format predmetnog programa bazirati na školskoj varijanti ELF formata (tekstualni fajl kakav je korišćen u zadatku 9 u prezentaciji V3\_Konstrukcija asemblera.ppt) i predložiti izmene u formatu u skladu sa potrebama ciljne arhitekture (nove sekcije, novi tipovi zapisa o relokacijama, dodatna polja u postojećim tipovima zapisa, novi podaci o predmetnom programu i slično). Prilikom generisanja izlaza asemblera voditi se principima koje koristi GNU asembler. Sve sekcije smeštaju se počev od nulte adrese.

# 2. Анализа проблема

С обзиром да се начин обраде улазног програма заснива на принципу двопролазног асемблера, потребно је поделити који део обраде ће се радити у ком пролазу.

У првом пролазу, асемблер најчешће има задатак да изгради табелу симбола и да у њу упише све битне информације до којих се може доћи у првом пролазу. Такође, као помоћна структура, генерише се и табела секција, која садржи податке о локацији и величини сваке секције улазног програма.

Други пролаз служи да се на основу типа адресирања и симбола који се адресирају, генеришу релокациони записи за све она адресна поља која се не могу разрешити у току асемблирања. На основу типова адресирања, инструкција, услова, вредности поља итд., у другом пролазу асемблер генерише и садржај у меморији. Тај садржај, заједно са табелом симбола и секција, али и релокационим записима, биће довољан повезиваоцу и пуниоцу да успешно генеришу извршни програм.

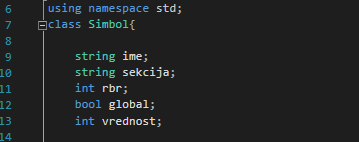
# 3. Решење

Програм за обраду улазне датотеке и генерисање излазне реализован је у језику С++. За сам развој софтвера је коршћено интегрисано развојно окружење Visual Studio 2017, због много лакшег тестирања и дебаговања. Након што је програм прорадио у том окружењу на оперативном систему Microsoft Windows 10, пребачени су фајлови са изворним кодом на систем Linux (x86 архитектура).

**Решење** чини 7 класа:

* Asembler
* TabelaSekcija
* TabelaSimbola
* RelokacioniZapis
* Instrukcija
  + Direktiva
  + Naredba

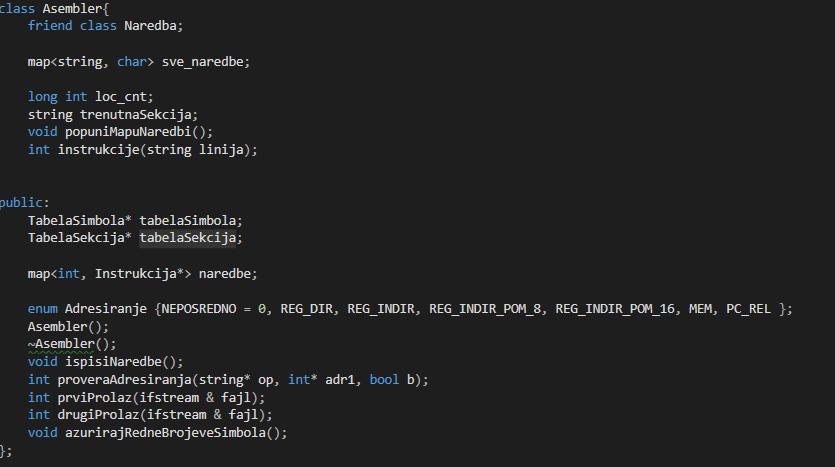
# 3.*1. Tabela simbola*



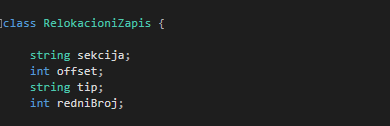
# 3.*2. Tabela sekcija*

# 

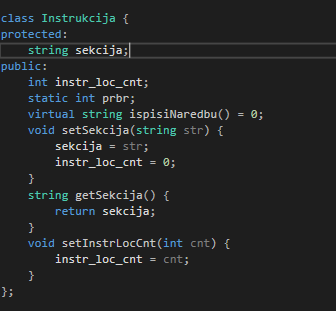
# 3.*2*. *Asembler*

****

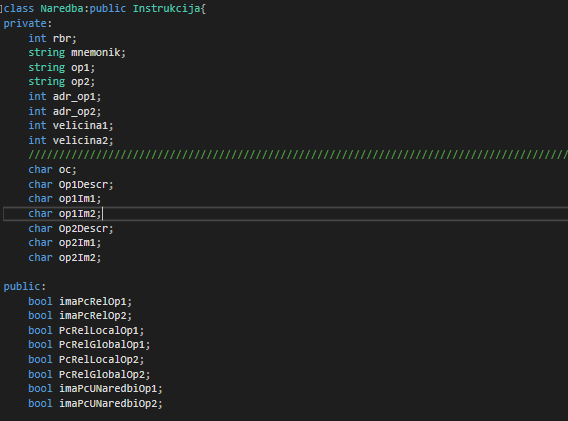
# 3.*4.RELOKACIONI ZAPIS*

****

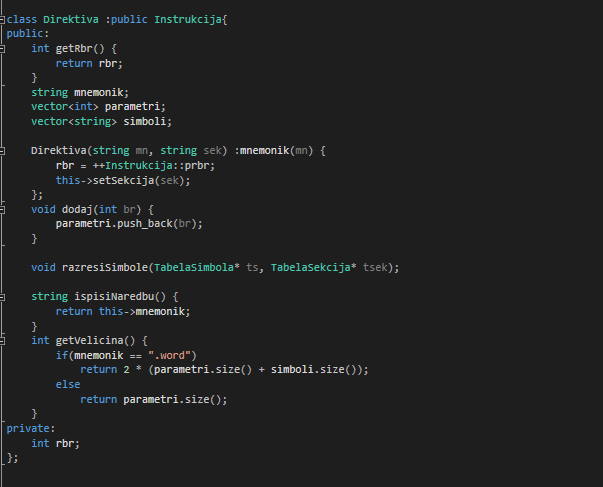
# 3.*5.Instrukcije*



# 3.*6.Naredbe*



# 3.*7.Direktiva*



# 4. Улаз/излаз

# Улаз

# 

# ИЗЛАЗ

