Domaći zadatak  
iz predmeta

Sistemski softver

Student  
Vučković Marina 0230/2015

**Uputstvo** i glavni delovi projekta su klase Symbol, Section, SymbolTable, AuxiliaryClass, FirstPass i SecondPass koje su realizovane kroz odgovarajuće deklaracije i definicije u vidu .h i .cpp fajlova.   
 Klasa Symbol je predstava simbola, dok je klasa Section reprezentacija jedne sekcije. Zajedničko za ove klase je ime i id objekta. Simbol dodatno evidentira njegovu lokalnost/ globalnost, kao i kojoj sekciji pripada, dok sama sekcija ima i dodatne informacije o veličini kao i koje od operacija čitanja, upisa i izvršavanja su dozvoljene u okviru sekcije.  
 Svi svimboli i sekcije koje se nalaze u ulaznom fajlu čuvaju se u okviru klase SymbolTable, koja ima podršku za realizaciju osnovnih funkcionalnosti sa simbolima i sekcijama, u vidu dodadavanja, pretraživanja i ispisa istih u izlazni fajl.   
 Dalje je predstavljen AuxiliaryClass koji je sačinjen iz deklaracija i definicija kroz .h i . cpp fajl čija je namena da realizuje osnovne i pomoćne metode u podršci parsiranja ulaznog tekstualnog fajla, neophodnih za rad prvog i drugog prolaza.  
 Prvi prolaz realizovan je klasom FirstPass, koja ima svoju spostvenu tabelu simbola (SymbolTable). Neke od funkcionalnosti prvog prolaza su čitanje ulaznog fajl, sakupljanje simbola nailaskom na direktive .global, .equ, pojava simbola u vidu labela, sakupljanje sekcija dodavajući ih u tabelu simbola, i definisanje veličine samih sekcije koji se realizuju prepozanavanjem instrukcija i odredjivanjem njihovih veličina kao i prepoznavanjem direktiva .byte, .word, .skip, .align i uvećanja sekcije za odgovrajuću veličinu.  
 Drugi prolaz realizovan klasom SecondPass genereše mašinski kod i tabelu relokacionih zapisa. Mašinski kod generiše se nailaskom na direktive .byte, .word, .skip i .align, kao i nailaskom na instrukcije, i u zavisnosti od argumenata direktiva odn. operanada instrukcija generišu se relokacioni zapisi. Drugi prolaz se kreira na kraju prvog, prosledjuje mu se tabela simbola prvog simbola, kao i linija prvog prolaza koje treba obraditi u drugom. Time je fizički i logički razdvojeno šta se radi u prvom a šta u drugom prolazu.

**Prevodjenje** se radi pomoću makefile-a čiji je sadržaj dat u nastavku.

all: prog

prog: main.o AuxiliaryClass.o FirstPass.o SecondPass.o Section.o Symbol.o SymbolTable.o

g++ main.o AuxiliaryClass.o FirstPass.o SecondPass.o Section.o Symbol.o SymbolTable.o -o prog

main.o: main.cpp

g++ -c main.cpp

AuxiliaryClass.o: AuxiliaryClass.cpp AuxiliaryClass.h

g++ -c AuxiliaryClass.cpp

FirstPass.o: FirstPass.cpp FirstPass.h

g++ -c FirstPass.cpp

SecondPass.o: SecondPass.cpp SecondPass.h

g++ -c SecondPass.cpp

Section.o: Section.cpp Section.h

g++ -c Section.cpp

Symbol.o: Symbol.cpp Symbol.h

g++ -c Symbol.cpp

SymbolTable.o: SymbolTable.cpp SymbolTable.h

g++ -c SymbolTable.cpp

clean:

rm -rf \*o prog

Pre samog prevodjenja neophodno je instanlirati g++ pomoću komande   
sudo apt-get install g+++

Generisanje izvršnog programa se radi komandom   
make all

**Pokretanje** programa radi se iz komandne linije. Ukuloko se na primer traži predmetni program od fajla tekst1.s, sadržaj komandne linije bi bio  
 ./prog asembler -o tekst1.o tekst1.s

U slučaju da makefile ne radi, može se instalirati g++-6 sledećim komandama  
 sudo apt-get install g++

sudo add-apt-repository ppa:ubuntu-toolchain-r/test

sudo apt-get update

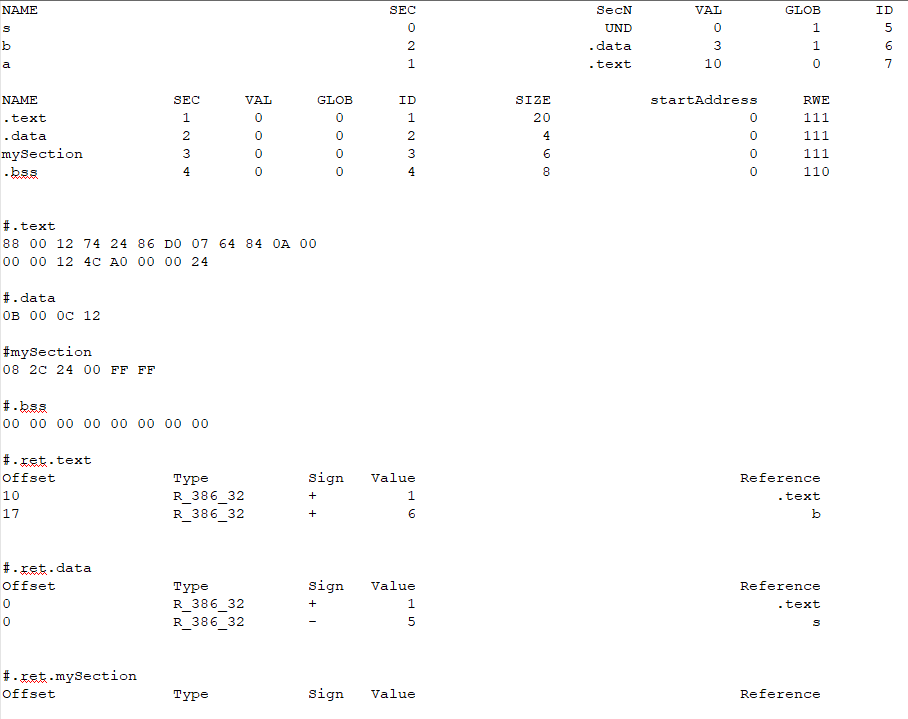
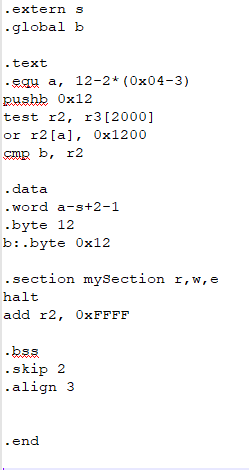
sudo apt-get install g++-6

U tom slučaju generisanje izvršnog koda je iz komande linije bez makefile-a i terba navesti sve .h i .cpp fajlove klasa, kao i main.

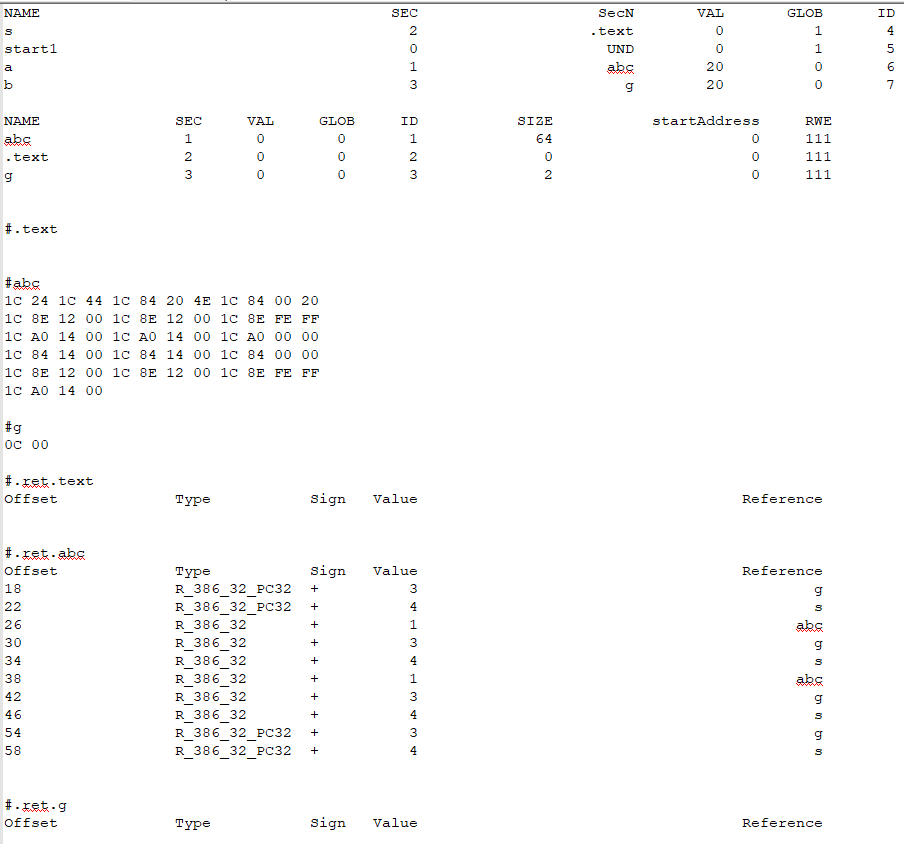
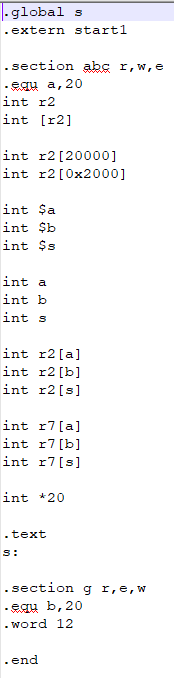
g++-6 -o prog AuxiliaryClass.h AuxiliarzClass.c Emulator.h Emulator.cpp FirstPass.h FirstPass.cpp LinkerLoader.h LinkerLoader.cpp SecondPass.h SeconPass.cpp Section.h Section.cpp Symbol.h Symbol.cpp SymbolTable.h SymbolTable.cpp main.cpp

Pokretanje programa je opet ./prog asembler -o tekst1.o tekst1.s

Primer 1.



Primer 2.



Primer 3.

