Programiranje 2
Programski jezik C

Kompilacija iz više izvornih datoteka

19. februar 2015

Primer biblioteke -kompleksni brojevi-

Napisati program koji koristi kompleksne brojeve:

a) Definisati strukturu KompleksanBroj koja predstavlja kompleksan broj i sadrži realan i imaginaran deo kompleksnog broja.

b) Napisati funkciju koja
   - ispisuje kompleksan broj na standardni izlaz, što lepše.
   - učitava kompleksan broj sa standardnog ulaza.
   - računa vrednosti realnog (imaginarnog) dela broja.
   - sabira (oduzima, množi) dva kompleksna broja.
   - računa konjugovano-kompleksni broj svog argumenta.
   - računa moduo kompleksnog broja.
   - računa argument kompleksnog broja.

Postupak

- Paralelno dok pišemo funkcije, pišemo i main funkciju koja testira napisane funkcije.
- Želimo da napišemo biblioteku koju ćemo moći da koristimo u nekim narednim programima. Zato treba izdvojiti deo koji se odnosi na "korisne" funkcije, od main funkcije koja nam je potrebna samo za trenutni program. Od jednog .c dokumanta sa izvornim kodom, pravimo dva. Jedan će sadržati main funkciju (main.c), a drugi će sadržati sve ostalo (complex.c). Kako nam je izvorni kod izdeljen, da bi preveli ceo program, treba kompajlirati ova dva .c dokumenta. To činimo sledećom naredbom:

  gcc -Wall -lm –o complex complex.c main.c

- Prevođenje ne prolazi.

U čemu je problem?

- Prevođenje nije prošlo jer prevodilac u fajlu main.c nije znao što je tip KompleksanBroj, šta je real, itd.
- Ovo rešavamo tako što u main.c prepišemo typedef deo iz complex.c, prepišemo define direktivu, i stavimo deklaracije funkcija koje smo definisali u complex.c.
- Preostao nam je još jedan problem.
- Ako hoćemo da pravimo neke ispravke u strukturi KompleksanBroj, ili da promenimo prototip neke funkcije iz complex.c, moraćemo da iste izmene uradimo u obe .c datoteke. Rešenje je da deklaracije iz obe datoteke izdvojimo u zaglavlje (header) complex.h, i da isti uključimo u obe .c datoteke sa:

  #include "complex.h"

Complex.h

- Zaglavlje koje sami pišemo navodimo između navodnika za razliku od zaglavlja standardne biblioteke.
- Ovde se podrazumeva da se complex.h nalazi u tekućem direktorijumu (istom u kome se nalaze i .c datoteke). Ako se on nalazi negde drugde u fajlsistemu, onda se pod navodnicima mora navesti apsolutna ili relativna putanja do zaglavlja.
- Prevodilac gcc standardna zaglavlja traži na nekim lokacijama koje su unapred određene. U našem slučaju mi moramo da mu naznačimo gde se nalazi naše zaglavlje, na prethodno opisan način, ili opcijama prevodioca.

Pravila lepog kodiranja

- Korisničko zaglavlje se navodi posle bibliotečkih zaglavlja.
- Pored deklaracija funckija, definicija korisničkih tipova, makroa itd. zaglavlje treba da se zaključava pretprocesorskim direktivama i time spreči višestruko uključivanje.

#ifndef _COMPLEX_H_

#define _COMLEX_H

Tekst koji se inače nalazi unutar zaglavlja

#endif

- Izazvati problem dodavanjem novog zaglavlja koje se takođe uključuje u main.c, a u sebi sadrži uključivanje complex.h

Drugi način kompilacije

Kompilaciju možemo uraditi i na sledeći način:

gcc -Wall -c -o complex.o complex.c

gcc -Wall -c -o main.o main.c

gcc -lm -o complex complex.o main.o

Objašnjenje naredbi

gcc -Wall -c -o complex.o complex.c

- Poziva prevodilac za kod compex.c sa opcijama:
  - –Wall (štampaj upozorenja prevodioca),
  - -lm (za linkovanje sa math.h bibiliotekom),
  - -o (fajl koji prevodilac generiše imenuj sa complex.o)
  - -c (ne vrši prevođenje do izvršnog programa, već samo do objektnog koda).
- Rezultat ovoga je objektni fajl complex.o, koji sadrži program na mašinskom jeziku. Još uvek nije izvršni program, jer nije urađeno uvezivanje (linkovanje) biblioteka koje su u njemu korišćene, i ostalih objektnih fajlova koji se koriste sa njim.
- Druga naredba radi analogno za main.c.

Objašnjenje naredbi

- Ova dva objektna fajla treba ulinkovati međusobno, i sa objektnim kodom standardne biblioteke. To se radi trećom naredbom. Prevodilac gcc prepoznaje da su njegovi argumenti objektni fajlovi i da ne treba da ih prevodi, već samo da ih ulinkuje ispravno.

gcc -lm -o complex complex.o main.o

Automatizacija prevođenja

- Kako se za prevođenje našeg programa sada koriste 3 komande, da bi se ovaj proces kompilacije automatizovao postoji make alat.
- Potrebno je da u datoteci koja se zove makefile popišemo postupak kojim se od izvornih datoteka dobija željeni izvršni program, a make alat će za nas izvrši naznačene komande.
- Detaljnije uputstvo o make alatu, gcc prevodiocu i još nekim korisnim alatima možete naći u skripti „GNU alati“ Ace Samardžića. http://poincare.matf.bg.ac.rs/~asamardzic/gnu.pdf