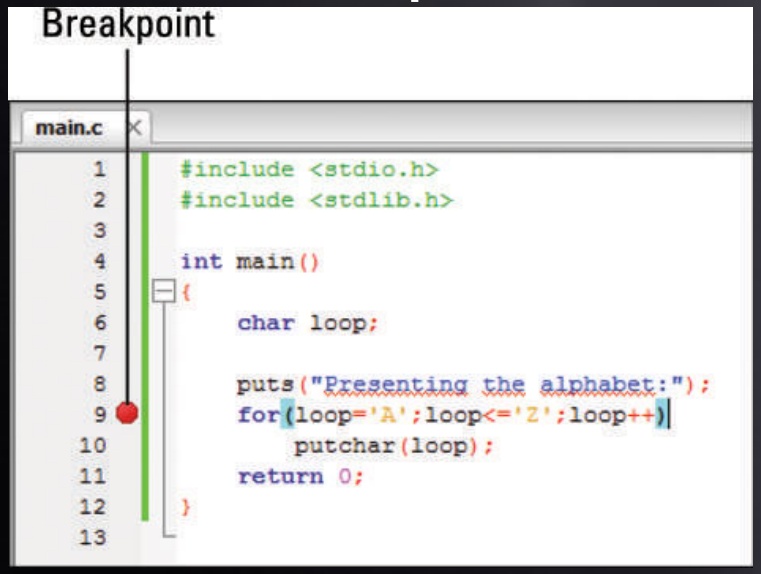
Programiranje 2
Programski jezik C

Kompilacija iz više izvornih datoteka
Bitski operatori



PRIMER BIBLIOTEKE
- polinomi -

Napisati program koji radi sa polinomima:

(a) Definisati strukturu Polinom koja predstavlja polinom (stepena najviše 20). Struktura sadrži stepen i niz koeficijenata. (razmisliti o redosledu navođenja koeficijenata u nizu)

(b) Napisati funkciju koja ispisuje polinom na standardni izlaz u što lepšem obliku.

(c) Napisati funkciju koja učitava polinom sa standardnog ulaza.

(d) Napisati funkciju za izračunavanje vrednosti polinoma u datoj tački (prednost Hornerovog algoritma).

(e) Napisati funkciju koja sabira dva polinoma.

(f) Napisati funkciju koja množi dva polinoma.

PRIMER BIBLIOTEKE
- polinomi -

Paralelno dok pišemo funkcije, pišemo i main funkciju koja testira napisane funkcije.

Želimo da napišemo funkcionalnu 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 fajla, pravimo dva, jedan koji će sadržati main funkciju (test-polinom.c), i drugi koji će sadržati sve ostalo (polinom.c). Da bi preveli ceo naš program, treba kompajlirati ova dva fajla. To činimo sledećom naredbom:

```
gcc -Wall -o polinom polinom.c test-polinom.c
```

Prevođenje ne prolazi.

PRIMER BIBLIOTEKE
- polinomi -

Prevođenje nije prošlo jer prevodilac u fajlu test-polinom.c nije znao što je tip Polinom, šta je MAX_STEPEN, itd... Ovo rešavamo tako što u test-polinom.c prepišemo typedef iz fajla polinom.c, prepišemo define direktivu, i stavimo deklaracije funkcija koje smo definisali u polinom.c. Ovo će rešiti prethodne probleme. Kompilaciju možemo uraditi i na sledeći način:

```
gcc -Wall -c -o polinom.o polinom.c
gcc -Wall -c -o test-polinom.o test-polinom.c
gcc -o polinom polinom.o test-polinom.o
```

PRIMER BIBLIOTEKE
- polinomi -

Prva naredba poziva prevodilac za fajl polinom.c sa opcijama -Wall (štampaj upozorenja prevodioca), -o (fajl koji prevodilac generiše imenuj sa polinom.o) i -c (ne vrši prevođenje do izvršnog fajla, već samo do objektnog).

Rezultat ovoga je objektni fajl polinom.o. Taj program je na mašinskom jeziku, ali nije izvršni, jer nije izvršeno linkovanje biblioteka koje su u njemu korišćene, i ostalih objektnih fajlova koji se koriste sa njim.

Druga naredba radi analogno za test-polinom.c.

PRIMER BIBLIOTEKE
- polinomi -

Ova dva objektna fajla treba ulinkovati međusobno, i ulinkovati 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.

Problem koji se sada javio je sledeći:

Ako hoćemo da pravimo neke ispravke u strukturi Polinom, ili da promenimo prototip neke funkcije iz polinom.c, moraćemo da menjamo obe .c datoteke.

Rešenje je da deklaracije iz obe datoteke izdvojimo u header-fajl polinom.h, i da isti uključimo u obe .c datoteke. #include “polinom.h”

PRIMER BIBLIOTEKE
- polinomi -

Možemo primetiti da se header-fajlovi koje sami pišemo navode između navodnika za razliku od standardnih header-fajlova. Ovde se podrazumeva da se polinom.h nalazi u tekućem direktorijumu (istom u kome se nalaze i .c fajlovi).

Ako se on nalazi negde drugde u fajl-sistemu, onda se pod navodnicima mora navesti apsolutna ili relativna putanja od header-fajla.

Prevodilac gcc standardne .h fajlove traži na nekim lokacijama koje su unapred određene. U našem slučaju mi moramo da mu specificiramo gde se nalazi naše zaglavlje, na prethodno opisan način, ili nekim opcijama prevodioca.

PRIMER BIBLIOTEKE
- polinomi -

Kako su za prevođenje našeg programa potrebne 3 komande, da bi se ovaj proces kompilacije automatizovao, postoji make alat. Mi specificiramo kako dolazimo do željenog izvršnog fajla u datoteci koja se zove makefile, a make alat za nas izvrši tražene komande.

Ovaj teskt o biblioteci sa polinomima je propraćen primerima koje možete naći mojoj stanici.

Detaljnije uputstvo o make-alatu, gcc prevodiocu I još nekim korisnim alatima možete nađi u sledećoj skripti:

http://poincare.matf.bg.ac.rs/~asamardzic/gnu.pdf

Bitski operatori

Bitski operatori u C-u su:

- Bitsko AND: &
- Bitsko OR: |
- Bitsko ekskluzivno OR: ^
- Levo pomeranje: <<
- Desno pomeranje: >>
- Komplement: ~

Ne treba ih mešati sa logičkim operatorima.

Bitski operatori - primer

```
#include <stdio.h>
int main() {
   printf("%o %o\n",255,15); /*255 i 15 se ispisuju u oktalnom zapisu*/
   printf( "255 & 15 = %d\n", 255 & 15 );
   printf( "255 | 15 = %d\n", 255 | 15 );
   printf( "255 ^ 15 = %d\n", 255 ^ 15 );
   printf( "2 << 2   = %d\n", 2 << 2 );
   printf( "16 >> 2 = %d\n", 16 >> 2 );
   return 0;
}
```

Bitski operatori - primer

Izlaz iz programa je:

377 17

255 & 15 = 15

255 | 15 = 255

255 ^ 15 = 240

2 << 2 = 8

16 >> 2 = 4

Bitski operatori - zadaci

1. Napisati funkciju koja ispisuje sve bitove datog celog broja (označeni i neoznačeni slučaj).

2. Napisati funkciju koja broji binarne jedinice u zapisu datog neoznačenog broja.

   (a) Prikazati varijantu sa pomeranjem maske (linearna složenost).

   (b) Prikazati varijantu sa pomeranjem broja (linearna složenost).

3. (a) Napisati funkciju koja određuje najveći broj koji se može zapisati istim binarnim ciframa kao dati broj.

   (b) Napisati funkciju koja određuje najmanji broj koji se može zapisati istim binarnim ciframa kao dati broj.

Bitovski operatori - zadaci

4. (a) Napisati funkciju koja određuje broj koji se dobija kada se n bitova datog broja, počeši od pozicije p postave na 0.

(b) Napisati funkciju koja određuje broj koji se dobija kada se n bitova datog broja, počeši od pozicije p postave na 1.

(c) Napisati funkciju koja izdvaja n bitova datog neoznačenog broja, počevši od pozicije p i vraća ih kao bitove najmanje težine rezultata. Npr. 7 bitova broja 1001011101110110, počevši od pozicije 9 su 1101110, pa rezultat treba da bude 0000000001101110.

(d) Napisati funkciju koja vraća broj koji se dobija upisivanjem poslednjih n bitova broja y u broj x, počevši od pozicije p.

(e) Napisati funkciju koja vraća broj koji se dobija invertovanjem n bitova broja x počevši od pozicije p.

Bitovski operatori - zadaci

5. (a) Napisati funkciju koja određuje broj koji se dobija rotiranjem ulevo datog celog broja.

(b) Napisati funkciju koja određuje broj koji se dobija rotiranjem udesno datog celog broja (neoznačena i označena varijanta).

Bitovki operatori samostalni rad

1. Napisati funkciju koja broji binarne jedinice u zapisu datog neoznačenog broja.

(a) U kojoj se u svakom koraku uklanja poslednja jedinica. (linearna složenost - broj koraka jednak broju jedinica)

(b) Prikazati paralelnu varijantu za 32 bita (logaritamska složenost). Napomena: sabrati 16 parova jednobitnih brojeva i dobiti 16 dvobitnih brojeva, zatim sabrati 8 parova dvobitnih brojeva i dobiti 8 četvorobitnih brojeva, zatim sabrati 4 para četvorobitnih brojeva i dobiti četiri osmobitna broja, zatim sabrati dva para osmobitnih brojeva i dobiti dva šesnaestobitna broja čijim se sabiranjem dobija jedan tridesetdvobitni broj koji predstavlja konačnu traženu sumu jedinica. Npr:

Bitovki operatori samostalni rad

```
01011011001010100110110101011011
01010110000101010101100101010110
00100011000100100010001100100011
00000101000000110000010100000101
00000000000010000000000000001010
00000000000000000000000000010010
```