Programiranje 1

Beleške sa vežbi

Školska 2008/2009 godina

Matematički fakultet, Beograd

Jelena Graovac (Tomašević)

November 13, 2008

Sadržaj

1 Programski jezik C **5**

1.1 Operatori i izrazi . . . 5

1.1.1 Aritmetički operatori . . . 5

1.1.2 Operatori dodele . . . 7

1.1.3 Operatori uvećana i umanjenja . . . 7

1.1.4 Relacioni i logički operatori . . . 8

1.2 Konverzija . . . 10

1.2.1 Automatska konverzija . . . 10

1.2.2 Eksplicitna konverzija . . . 10

1.3 Kontrola toka — grananja u programu i ciklusi . . . 11

1.3.1 Grananja u programu - if . . . 11

1.3.2 Else-if . . . 12

1.3.3 Ciklusi u programu — petlja while . . . 13

1.3.4 Petlja do-while . . . 14

1.3.5 Petlja for . . . 15

1.3.6 Naredbe break i continue . . . 17

1

Programski jezik C

[1]

1.1 Operatori i izrazi

U C-u postoji veliki broj operatora i oni mogu biti unarni i binarni. Unarni mogu biti prefiksni i sufiksni a binarni su po pravilu infiksni. Operatori imaju svoj prioritet i asocijativnost.

Kombinovanjem promenljivih, konstanti i operatora dobijamo izraze. Svaki izraz ima svoj tip i vrednost.

Tip izraza zavisi od tipova podizraza koji ga čine kao i od operatora kojim se ovi podizrazi povezuju. Ako su operandi neodgovarajućeg tipa onda se vrši implicitna konverzija, ako je to moguće. Tip izraza se može eksplicitno promeniti tzv. `cast` operatorom (ispred izraza se u zagradi navede ime tipa u koji želimo da konvertujemo izraz).

1.1.1 Aritmetički operatori

Operatori `+`, `-`, `*`, `/` i `%` nazivaju se aritmetički operatori. Rezultat aritmetičkih operacija je izraz istog tipa kao i operandi. Ako operandi nisu istog tipa onda se vrši implicitna konverzija užeg u širi tip.

Posebno je zanimljiv operator `/`. On realizuje celobrojno deljenje ako su oba operanda celi brojevi, a realno deljenje ako je bar jedan od operanada realan broj. Da li će se izvršiti celobrojno ili realno deljenje zavisi isključivo od tipa operanada a ne od tipa promenljive u koju se rezultat smešta.

Operator `%` se primenjuje samo na cele brojeve i daje ostatak pri deljenju.

Primer 1 *Program ilustruje neke od aritmetičkih operacija.*

```
#include <stdio.h>
main()
{
        int a, b;
        printf("Unesi prvi broj : ");
        scanf("%d",&a);

        printf("Unesi drugi broj : ");
```

[1] Zasnovano na primerima sa sajtova `http://www.matf.bg.ac.yu/~milan`, `http://www.matf.bg.ac.yu/~filip` i knjige Milana Čabarkape "C - osnovi programiranja".

```
        scanf("%d",&b);

        printf("Zbir a+b je : %d\n",a+b);
        printf("Razlika a-b je : %d\n",a-b);
        printf("Proizvod a*b je : %d\n",a*b);
        printf("Celobrojni kolicnik a/b je : %d\n", a/b);
        printf("Pogresan pokusaj racunanja realnog kolicnika a/b je : %f\n", a/b);
        printf("Realni kolicnik a/b je : %f\n", (float)a/(float)b);
        printf("I ovo je realni kolicnik a/b: %f\n", (float)a/b);
        printf("Ostatak pri deljenju a/b je : %d\n", a%b);

}
Ulaz:
Unesi prvi broj : 2 <enter>
Unesi drugi broj : 3 <enter>
Izlaz:
Zbir a+b je : 5
Razlika a-b je : -1
Proizvod a*b je : 6
Celobrojni kolicnik a/b je : 0
Progresan pokusaj racunanja realnog kolicnika a/b je : 0.000000
Realni kolicnik a/b je : 0.666667
I ovo je realni kolicnik a/b: 0.666667
Ostatak pri deljenju a/b je : 2
```

Primer 2 Program ilustruje celobrojno i realno deljenje.

```
#include <stdio.h>

main()
{
    int a = 5;
    int b = 2;
    int d = 5/2;    /* Celobrojno deljenje - rezultat je 2 */
    float c = a/b;  /* Iako je c float, vrsi se celobrojno deljenje jer su i a i b celi */

    /* Neocekivani rezultat 2.000000 */
    printf("c = %f\n",c);

    printf("Uzrok problema : 5/2 = %f\n", 5/2);

    printf("Popravljeno : 5.0/2.0 = %f\n", 5.0/2.0);

    printf("Moze i : 5/2.0 = %f i 5.0/2 = %f \n", 5/2.0, 5.0/2);

    printf("Za promenjive mora kastovanje : %f\n", (float)a/(float)b);

}

Izlaz iz programa:
c = 2.000000
Uzrok problema : 5/2 = 2.000000
```

```
Popravljeno : 5.0/2.0 = 2.500000
Moze i : 5/2.0 = 2.500000 i 5.0/2 = 2.500000
Za promenljive mora kastovanje : 2.500000
```

1.1.2 Operatori dodele

Operator proste dodele je operator `=`. Levi operand ovog operatora je leva vrednost (ime promenljive) a desni operand je proizvoljan izraz. Najpre se izračuna izraz na desnoj strani, njegova vrednost se po potrebi konvertuje u tip promenljive na levoj strani i nakon toga se ta vrednost dodeljuje promenljivoj na levoj strani. Bitno je napomenuti da izraz dodele (izraz koga čini operacija dodele) ima svoj tip (tip promenljive) i vrednost (vrednost dodeljena promenljivoj).

Pošto izraz dodele ima svoju vrednost, to se operacija dodeljivanja može koristiti na sledeći način:

```
int x,y,z;
z=2*(x=3)+4*(y=5/2);
```

Dakle, izraz `(x=3)` ima vrednost `3` koja je dodeljena promenljivoj `x`, a izraz `(y=5/2)` ima vrednost `2` (celobrojno deljenje) koja je dodeljena promenljivoj `y`. Prema tome `z` se dodeljuje vrednost `2*3+4*2`, odnosno `14`.

Izraz kao što je

$$i = i + 2;$$

u kojima se promenljiva na levoj strani odmah ponavlja na desnoj, mogu se pisati u skraćenom obliku kao

$$i+ = 2;$$

Operator += se naziva *operator složene dodele*. Ovo važi za većinu binarnih operatora (`+ - * / %`). Dakle, `izraz1 op = izraz2` je ekvivalnetno sa `izraz1 = (izraz1) op (izraz2)`.

Na primer $x* = y + 1$ je ekvivalento sa $x = x * (y + 1)$.

1.1.3 Operatori uvećana i umanjenja

Operator `++` dodaje vrednost 1 svom operandu a `--` oduzima 1. Ovo su unarni operatori i oni se mogu koristiti kao prefiksni (ispred promenljive, npr. ++n) i kao postfiksni (iza promenljive, npr. n++). U oba slučaja vrši se uvećanje promenljive za 1 ali izraz ++n uvećava promenljivu n *pre* nego što se njena vrednost koristi, dok n++ uvećava n *nakon* što se njena vrednost koristi. Tako se `x=++n;` razlikuje od `x=n++;`.

Primer 3 *Ilustracija prefiksnog i postfiksnog operatora ++*

```
#include <stdio.h>
main()
{
    int x, y;
    int a = 0, b = 0;

    printf("Na pocetku : \na = %d\nb = %d\n", a, b);

    /* Ukoliko se vrednost izraza ne koristi, prefiksni i
       postfiksni operator se ne razlikuju */
    a++;
    ++b;
```

```
    printf("Posle : a++; ++b; \na = %d\nb = %d\n", a, b);

    /* Prefiksni operator uvecava promenjivu, i rezultat
       je uvecana vrednost */
    x = ++a;

    /* Postfiksni operator uvecava promenjivu, i rezultat je
    stara (neuvecana) vrednost */
    y = b++;

    printf("Posle : x = ++a; \na = %d\nx = %d\n", a, x);
    printf("Posle : y = b++; \nb = %d\ny = %d\n", b, y);
}

Izlaz iz programa:
Na pocetku:
a = 0
b = 0
Posle : a++; ++b;
a = 1
b = 1
Posle : x = ++a;
a = 2
x = 2
Posle : y = b++;
b = 2
y = 1
```

1.1.4 Relacioni i logički operatori

Relacioni operatori su: `>` `>=` `<` `<=` . Svi oni imaju isti prioritet. Niži prioritet imaju operatori poređenja na jednakost `==` `!=`. Tako je `a < 5 != 1` ekvivalentno sa `(a < 5)!=1`. Svi relacioni operatori imaju niži prioritet od aritmetičkih operatora. Tako je `i < n-1` ekvivalentno sa `i < (n-1)`.

Logički operatori su:
`!` — unarna negacija,
`&&` — logičko i,
`||` — logičko ili.

Prioritet logičkih operatora je niži od prioriteta relacionih operatora i operatora jednakosti. Operator `!` je višeg prioriteta u odnosu na `&&` a on je višeg u odnosu na `||`.

Izrazi povezani logičkim operatorima izračunavaju se sleva na desno.

Napomena: U C-u ne postoji logički tip! U tu svrhu se koristi celobrojni tip pri čemu se svaki ceo broj različit od nule smatra da ima logičku vrednost tačno, a nula ima vrednost netačno.

Primeri:
`5 && 4` — vrednost je tačno,
`10 || 0` — vrednost je tačno,
`0 && 5` — vrednost je 0,
`!1` — vrednost je 0,
`!9` — vrednost je 0,
`!0` — vrednost je 1,

`!(2>3)` — vrednost je 1.

`a>b && b>c || b>d` je isto što i `((a>b) && (b>c)) || (b>d)`. Koja je vrednost ovog izraza ako je a=10, b=5, c=1, d=15?

Primer 4 *Ilustracija logičkih vrednosti (0 - netačno, različito od 0 - tačno).*

```
#include <stdio.h>

main()
{
    int a;

    printf("Unesi ceo broj : ");
    scanf("%d", &a);
    if (a)
        printf("Logicka vrednost broja je : tacno\n");
    else
        printf("Logicka vrednost broja je : netacno\n");
}

Ulaz:
Unesi ceo broj : 3 <enter>
Izlaz:
Logicka vrednost broja je : tacno

Ulaz:
Unesi ceo broj : 0 <enter>
Izlaz:
Logicka vrednost broja je : netacno
```

Primer 5 *Ilustracija logičkih i relacijskih operatora.*

```
#include <stdio.h>

main()
{
    int a = 5<3,  /* manje */
    b = 5>3,  /* vece */
    c = 3==5, /* jednako */
    d = 3!=5; /* razlicito */

    printf("5<3 - %d\n5>3 - %d\n3==5 - %d\n3!=5 - %d\n", a, b, c, d);

    printf("Konjunkcija : 3>5 && 5>3 - %d\n", a && b);
    printf("Disjunkcija : 3>5 || 5>3 - %d\n", a || b);
    printf("Negacija : !(3>5) - %d\n", !a);

}

Izlaz iz programa:
```

```
5<3 - 0
5>3 - 1
3==5 - 0
3!=5 - 1
Konjunkcija : 3>5 && 5>3 - 0
Disjunkcija : 3>5 || 5>3 - 1
Negacija : !(3>5) - 1
```

Operatori dodele su najnižeg prioriteta.

Ako dva operatora imaju isti prioritet onda se u obzir uzima asocijativnost koja može biti s leva na desno ili s desna na levo. Prioritet operatora može se promeniti korišćenjem zagrada.

1.2 Konverzija

1.2.1 Automatska konverzija

Ako se u nekom aritmetičkom izrazu pojave operandi raznih tipova tada se jedan konvertuje tako da odgovara tipu drugog operanda, u smeru manjeg ka većem tipu:

```
char<int<long<float<double
```

Korišćenjem rezervisane reči `unsigned` povišava se hijararhijski rang odgovarajućeg tipa.

Naredba dodele:

```
int i=5;
float f=2.3;
f=i; /* f ce imati vrednost 5.0*/

obrnuto:

int i=5;
float f=2.3;
i=f; /* i ce imati vrednost 2*/
```

1.2.2 Eksplicitna konverzija

```
(tip)<izraz>

float x;
x=2.3+4.2;          /* x ce imati vrednost 6.5 */
x=(int)2.3+(int)4.2; /* x ce imati vrednost 6 */
x=(int)2.3*4.5;     /* x ce imati vrednost 9.0 jer zbog prioriteta
                    operatora konverzije prvo ce biti izvrsena
                    konverzija broja 2.3 u 2 pa tek onda izvrseno
                    mnozenje. */
x=(int)(2.3*4.5)    /* x ce imati vrednost 10.0 */
```

Primer 6 *Kako izbeći celobrojno deljenje*

```
int a,b;
float c;
a = 5;
b = 2;
```

```
c = a/b; /* Celobrojno deljenje, c=2*/
c = (1.0*a)/b;  /* Implicitna konverzija: 1.0*a je realan
                broj pa priliko deljenja sa b dobija se
                realan rezultat c=2.5*/
c = (0.0+a)/b;  /* Implicitna konverzija: (0.0+a) je realan
                broj pa priliko deljenja sa b dobija se
                realan rezultat c=2.5*/
c = (float)a/(float)b; /* Eksplicitna konverzija*/
```

1.3 Kontrola toka — grananja u programu i ciklusi

1.3.1 Grananja u programu - if

```
if (izraz)
    naredba1
else
    naredba2
```

Naredba može biti prosta naredba a može biti i složena naredba (blok) koja se dobije kada se više prostih naredbi grupišu navođenjem vitičastih zagrada.

Primer 7 *Program ilustruje if i ispisuje ukoliko je uneti ceo broj negativan.*

```
#include <stdio.h>

int main()
{
    int b;
    printf("Unesi ceo broj:");
    scanf("%d", &b);
    if (b < 0)
        printf("Broj je negativan\n"); //prosta naredba
    return 0;
}
```

```
Ulaz:
Unesi ceo broj:-5
Izlaz:
Broj je negativan

Ulaz:
Unesi ceo broj:5
Izlaz:
```

`Else` se odnosi na prvi neuparen `if`. Ako želimo drugačije moramo da navedemo vitičaste zagrade.

```
if (izraz) //prvo if
    if (izraz1) naredba1 //drugo if
else naredba2
```

Ovo else se odnosi na drugo if a ne na prvo if!

```
if (izraz)
    {
    if (izraz1) naredba1
    }
else naredba2
```

Tek sada se else odnosi na prvo if!!!

1.3.2 Else-if

```
if (izraz1)
    iskaz1
else if (izraz2)
    iskaz2
else if (izraz3)
    iskaz3
else if (izraz4)
    iskaz4
else iskaz

npr if (a<5)
    printf("A je manje od 5\n");
else if (a==5)
    printf("A je jednako 5\n");
else if (a>10)
    printf("A je vece od 10\n");
else if (a==10)
    printf("A je jednako 10\n");
else printf("A je vece od pet i manje od 10\n");
```

Primer 8 *Program ilustruje if-else konstrukciju i ispituje znak broja.*

```
#include <stdio.h>

int main()
{
    int b;
    printf("Unesi ceo broj : ");
    scanf("%d", &b);
    if (b < 0)
        printf("Broj je negativan\n");
    else if (b == 0)
        printf("Broj je nula\n");
    else
        printf("Broj je pozitivan\n");
    return 0;
}

Ulaz:
Unesi ceo broj:-5
Izlaz:
Broj je negativan
```

```
Ulaz:
Unesi ceo broj:5
Izlaz:
Broj je pozitivan
```

Primer 9 *Pogresan program sa dodelom = umesto poredjenja ==.*

```
#include <stdio.h>

int main()
{
    int b;
    printf("Unesi ceo broj : ");
    scanf("%d", &b);

    /* Obratiti paznju na = umesto == Analizirati rad programa*/
    if (b = 0)
        printf("Broj je nula\n");
    else if (b < 0)
        printf("Broj je negativan\n");
    else
        printf("Broj je pozitivan\n");
    return 0;
}

Ulaz:
Unesi ceo broj:-5
Izlaz:
Broj je pozitivan
```

Napomena: Voditi računa o tome da je `=` operator dodele a `==` je operator poređenja na jednakost. Ne mešati ta dva operatora!

1.3.3 Ciklusi u programu — petlja while

`while(izraz) naredba`
Uslov u zagradi se testira i ako je ispunjen telo petlje (naredba) se izvršava. Zatim se uslov ponovo testira i ako je ispunjen ponovo se izvršava telo petlje. I tako sve dok uslov ne postane neispunjen. Tada se izlazi iz petlje i nastavlja sa prvom sledećom naredbom u programu.

Napomena: Voditi računa o tome da li je naredba koja čini telo petlje prosta ili složena! Ako nema vitičastih zagrada onda se prva naredba iza while(uslov) tretira kao telo while petlje. Na primer, u sledećem fragmentu koda

```
while (i<j)
    i=2*i;
    j++;
```

samo naredba `i=2*i;` se ponavlja u okviru `while` petlje dok se naredba `j++;` izvršava tek nakon izlaženja iz `while` petlje.

Primer 10 *Program ilustruje petlju - while.*

```
#include <stdio.h>

int main()
{
    int x;

    x = 1;
    while (x<10)
    {
        printf("x = %d\n",x);
        x++; /* x++ je isto kao i x=x+1 */
    }

}

Izlaz:
x = 1
x = 2
x = 3
x = 4
x = 5
x = 6
x = 7
x = 8
x = 9
```

1.3.4 Petlja do-while

Ovo je slično repeat-until izrazu u Pascal-u.

```
do naredba while (izraz);
```

Primer 11 *Program ilustruje petlju do-while.*

```
#include <stdio.h>

int main()
{
    int x;

    x = 1;
    do
    {
        printf("x = %d\n",x);
        x++; /* x++ je isto kao i x=x+1 */
    }
    while (x<10);

}
```

```
Izlaz:
x = 1
x = 2
x = 3
x = 4
x = 5
x = 6
x = 7
x = 8
x = 9
```

1.3.5 Petlja for

```
for (izraz1; izraz2; izraz3) naredba
```

Ovo je ekvivalentno kodu:

```
izraz1;
while (izraz2)
{
    naredba
    izraz3;
}
```

Primer 12 *Program ilustruje for petlju.*

```
#include <stdio.h>

int main()
{
    int x;

    for (x = 1; x < 10; x++)
        printf("x = %d\n",x);

}

Izlaz:
x = 1
x = 2
x = 3
x = 4
x = 5
x = 6
x = 7
x = 8
x = 9
```

Napomena: `izraz1`, `izraz2`, `izraz3` i `naredba` mogu biti izostavljeni. Ako je `izraz2` izostavljen podrazumeva se da je stalno tačan.
`for( ; ; );` pretstavlja "beskonačnu" `for` petlju.

Primer 13 *Konverzija centimetara u inče - while petlja.*

```
#include <stdio.h>

/* Definicija simbolickih konstanti preko #define direktiva */
/* U fazi pretprocesiranja se vrsi doslovna zamena konstanti
   njihovim vrednostima */

#define POCETAK 0
#define KRAJ 20
#define KORAK 10

int main()
{
    int a;
    a = POCETAK;
    while (a <= KRAJ)
    {
        printf("%d cm = %f in\n", a, a/2.54);
        a += KORAK; /* isto sto i a = a + KORAK; */
    }
    return 0;
}

Izlaz:
0 cm = 0.000000 in
10 cm = 3.937008 in
20 cm = 7.874016 in
```

Primer 14 *Konverzija centimetara u inče - for petlja.*

```
#include <stdio.h>
#define POCETAK 0
#define KRAJ 20
#define KORAK 10

int main()
{
    int a;
    for  (a = POCETAK; a <= KRAJ; a += KORAK)
        printf("%d cm = %f in\n", a, a/2.54);

    return 0;
}

Izlaz:
0 cm = 0.000000 in
10 cm = 3.937008 in
20 cm = 7.874016 in
```

1.3.6 Naredbe break i continue

Naredba break omogućava prevremeni izlazak iz petlje a continue omogućava izlazak iz tekuće iteracije u petlji i nastavak izvršenja petlje počev od sledeće iteracije.

Primer 15 *Ilustracija naredbe break*

```
#include <stdio.h>

int main()
{
    int i;
    for(i=1; i<=5; i++)
    {
        if(i==3) break;
        printf("i = %d\n", i);
    }

    return 0;
}

Izlaz:
i = 1
i = 2
```

Primer 16 *Ilustracija naredbe continue*

```
#include <stdio.h>

int main()
{
    int i;
    for(i=1; i<=5; i++)
    {
        if(i==3) continue;
        printf("i = %d\n", i);
    }

    return 0;
}

Izlaz:
i = 1
i = 2
i = 4
i = 5
```

Zadaci za praktikum:

Zadatak 1 *Napisati program koji izračunava zbir recipročnih vrednosti prvih 10 brojeva.*

Zadatak 2 *Izvršiti štampanje parnih brojeva od 1 do 100 (for, while i do-while).*

Zadatak 3 *Napisati program koji izračunava sumu i maksimum brojeva koji se unose na standardni ulaz pri čemu je poslednji uneti broj 0 (for, while).*

Zadatak 4 *Napisati program koji ispisuje kvadrate svih brojeva od 5 do 35. Nakon svakog petog kvadrata odštampati znak za novi red (for, while).*

Zadatak 5 *Napisati program koji izračunava koliki je realni deo kompleksnog broja* $(1 + i)^{21}$ *(rešenje je -1024).*

Zadatak 6 *Napisati program koji sabira pozitivne brojeve niza cifara koji završava nulom i koji se unose sa standardnog ulaza.*

Primer 17 *Napisati program koji računa zbir* $1 + x + \frac{x^2}{2} + \ldots + \frac{x^n}{n!}$

Primer 18 *Napisati program koji računa sumu* $1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \ldots + (-1)^n \frac{x^{2n}}{(2n)!}$