Operatori

Srednja škola fra Slavka Barbarića Čitluk
Izborna nastava

Mario Miletić

Operatori

- Operatori su simboli koji predstavljaju (zamjenjuju) određen funkcije.
- U C-u je definirano nekoliko skupina operatora:
  - Aritmetički opetarori
  - Logički operatori
  - Operatori usporedbe
  - Ostali operatori

Aritmetički operatori

- Aritmetički operatori mogu biti:
  - Binarni (djeluju na dvije varijable)
  - Unarni (djeluju na samo jednu varijablu)

Binarni aritmetički operatori

- U tablici su prikazani binarni operatori za izvođenje osnovnih aritmetičkih funkcija.

+	zbrajanje
–	oduzimanje
*	množenje
/	dijeljenje
%	modulo rezultat je ostatak dijeljenja dvaju cijelih brojeva

Unarni aritmetički operatori

- Unarni operatori djeluju samo na jednu varijablu.
- Razlikujemo sljedeće unarne operatore:
  - za promjenu predznaka
  - za uvećavanje (inkrementiranje)
  - za umanjivanje (dekrementiranje)

Unarni aritmetički operatori

– a	Unarni minus	Mijenja predznak broja
a++	Operator za uvećavanje	Uvećava broj za 1 (nakon što se varijabla dobavi iz memorije)
a – –	Operator za umanjivanje	Umanjuje broj za 1 (nakon što se varijabla dobavi iz memorije)
++a	Operator za uvećavanje	Uvećava broj za 1 (prije nego se varijabla dobavi iz memorije)
– – a	Operator za umanjivanje	Umanjuje broj za 1 (prije nego se varijabla dobavi iz memorije)

Unarni aritmetički operatori

- Pri uporabi unarnih operatora važno je paziti na položaj operatora (da li se nalazi prije ili poslije varijable) jer se njegovo djelovanje u jednom i drugom slučaju razlikuje.

Unarni operatori prije varijable

- Ako je operator ispred varijable (npr. ++a) tada se u izrazu računa s uvećanom/umanjenom vrijednošću varijable. U primjeru:

```
a=1;

b=++a +5;
```

- Po izvršenju naredbi, sadržaj varijabli je:

  a=2, b=7.

  (Prvo se povećava vrijednost varijable a za 1, a zatim se računa vrijednost izraza.)

Unarni operatori nakon varijable

- Ako je operator iza varijable (npr. a++) tada se vrijednost varijable uveća/umanji tek nakon izračunavanja izraza. U primjeru:

```
a=1;

b=a++ +5;
```

- Nakon izvršenja naredbi, sadržaj varijabli je:

  a=2, b=6.

  (Prvo se računa vrijednost izraza, a nakon toga se povećava vrijednost varijable a za 1.)

Primjer 1

- Zadatak:
  - Potrebno je izračunati zbroj, razliku, umnožak i kvocijent dvaju realnih brojeva.
  - Ispis neka bude oblika:

```
Unesi prvi broj:
unesi drugi broj:
.... + .... = ....
.... - .... = ....
.... * .... = ....
.... / .... = ....
```

Primjer 1

- Napomene:
  - Na početku deklarirati dvije realne varijable.
  - Aritmetički operatori i znak = ne mogu se unijeti kao znakovni nizovi.

Primjer 1

Rješenje:

```c
#include<stdio.h>

void main(void) {

	float a,b;

	printf("Unesi prvi broj:");
	scanf("%f", &a);

	printf("Unesi drugi broj:");
	scanf("%f", &b);

	printf("%f + %f = %f\n", a, b, a+b);
	printf("%f * %f = %f\n", a, b, a*b);
	printf("%f - %f = %f\n", a, b, a-b);
	printf("%f / %f = %f\n", a, b, a/b);

}
```

```
Unesi prvi broj:1
Unesi drugi broj:2
1.000000 + 2.000000 = 3.000000
1.000000 * 2.000000 = 2.000000
1.000000 - 2.000000 = -1.000000
1.000000 / 2.000000 = 0.500000
Press any key to continue . . .
```

Primjer 2

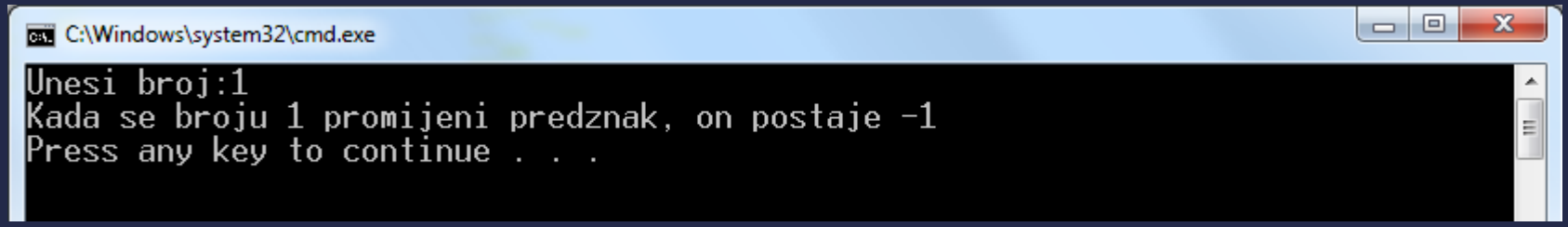
- Zadatak (unarni operator za promjenu predznaka):
  - Potrebno je unijeti cijeli broj, a zatim mu unarnim operatorom promijeniti predznak.
  - Ispis neka bude oblika:

```
Unesi broj:

Kada se broju .... promijeni
predznak, on postaje ....
```

Primjer 2

Rješenje:

```c
#include<stdio.h>

void main(void) {

	int a;

	printf("Unesi broj:");
	scanf("%d", &a);

	printf("Kada se broju %d promijeni predznak, on postaje %d\n", a, -a);

}
```



Komentari

- U izvornom programu (kôdu) korisno je opisati što program radi, što su argumenti, objasniti deklaraciju varijabli i sl.
- Takvi pomoćni opisi se nazivaju komentari.
- Temeljna namjena komentara je olakšati razumijevanje programa.

Komentari

- Komentar može biti napisan u istom redu s naredbom ili u zasebnom redu.
- Komentar je tekst koji započinje s /*, a završava s */.

```
int a,b; /*deklariram dvije cjelobrojne varijable a i b*/
float c;
```

- Komentari su obično prikazani u drugoj boji (zbog preglednosti – u C-u zelenom bojom).

Komentari

- Pri prevođenju izvornog kôda komentar se ne prevodi.
- Osim za opis programa komentar se ponekad rabi za privremeno isključivanje dijelova izvornog kôda.

Logički operatori

- Logičke funkcije se izvode uporabom logičkih operatora.
- Logički operatori mogu biti:
  - unarni
  - binarni

Logički operatori

!	Negacija (unarni operator koji 1 pretvara u 0 i obratno)
&&	Logički I (engl. *AND*)
||	Logički ILI (engl. *OR*)

- Logički se operatori uglavnom rabe u naredbama za grananje programa.

Logički operatori

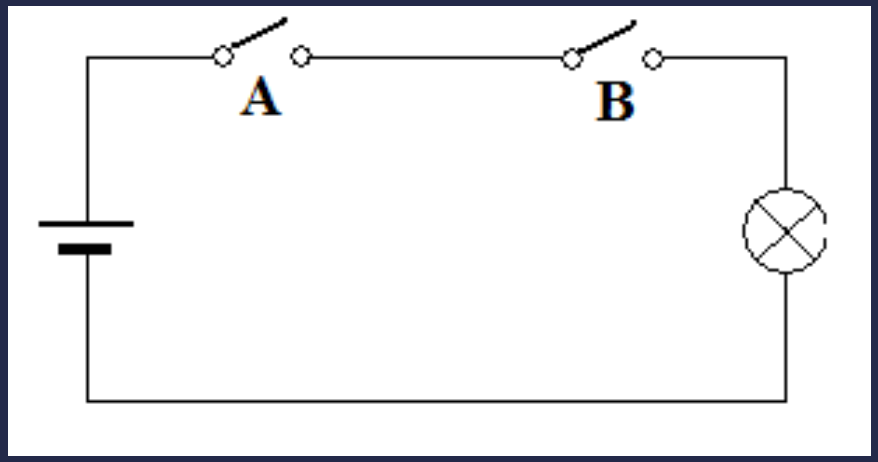
- Prilikom rješavanja logičkih izraza potrebno je obratiti pozornost na sljedeće:
  - Logički izrazi se rješavaju s desna na lijevo (vrijedi za **&&** i **||**)
  - Operator negacije (!) ima isti prioritet kao i predznak broja
  - Logički operatori **&&** i **||** imaju manji prioritet u odnosu na ostale operatore

Logičko I i logičko ILI

- Način rada logičkih operatora I i ILI najjednostavnije je prikazati strujnim krugom u kome se nalaze dvije sklopke.
- Stanje otvorene sklopke može se označiti sa 0, a zatvorene sa 1.
- Stanje u kome žaruljica svijetli može se označiti sa 1, a kada ne svijetli sa 0.

Logičko I

- Način rada logičkog I može se prikazati sa sklopkama koje su spojene serijski.
- Žaruljica će svijetliti samo kada su obje sklopke zatvorene.



A	B	Ž
0	0	0
1	0	0
0	1	0
1	1	1

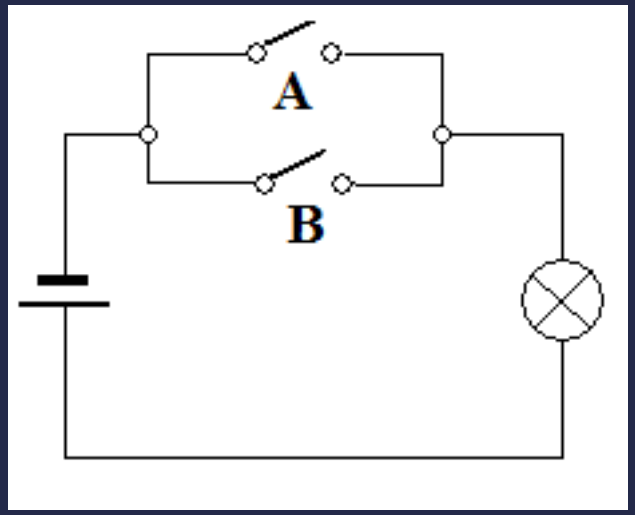
Logičko I

- Kao što pokazuje tablica stanja, logički operator I vraća jedinicu samo ako su oba uvjeta **true** (ispunjen uvjet, istina, 1).
- U ostalim stanjima rezultat je 0.

A	B	(A&&B)
0	0	**0**
1	0	**0**
0	1	**0**
1	1	**1**

Logičko ILI

- Način rada logičkog ILI može se prikazati sa sklopkama koje su spojene paralelno.
- Žaruljica će svijetliti ako je bilo koja (ili obje) sklopke zatvorene.



A	B	Ž
0	0	0
1	0	1
0	1	1
1	1	1

Logičko ILI

- Kao što pokazuje tablica stanja, logički operator ILI vraća jedinicu ako je ispunjen samo jedan od uvjeta (**true**, 1) ili ako su ispunjena oba uvjeta.
- U situaciji u kojoj nije ispunjen niti jedan od uvjeta, rezultat rada operatora je 0.
- Napomena: Operator ILI (||) zapisuje s kombinacijom tipki Alt Gr + W

A	B	(A||B)
0	0	**0**
1	0	**1**
0	1	**1**
1	1	**1**

Primjer 3

- Primjer:

```
!i && j || j && !j
```

Početne vrijednosti: **i = 1 , j = 0.**

```
1. !i = !1 = 0, !j = !0 =1
2. !i && j = 0 && 0 = 0
3. (!i && j) || j = 0 || 0 = 0
4. (!i && j || j ) && !j = 0 && 1 = 0
```

Operatori usporedbe

- Dva se broja mogu uspoređivati, a rezultat usporedbe određena je vrijednošću 1 ili 0.
- Ako je napisani izraz istinit, rezultat usporedbe će biti 1 (**true**), a ako nije rezultat će biti 0 (**false**).
- Uspoređuje se uporabom operatora usporedbe.

Operatori usporedbe

<	manje
<=	manje ili jednako
>	veće
>=	veće ili jednako
==	jednako
!=	različito

- Operatori usporedbe se najčešće rabe u naredbama za grananje.

Primjer 4

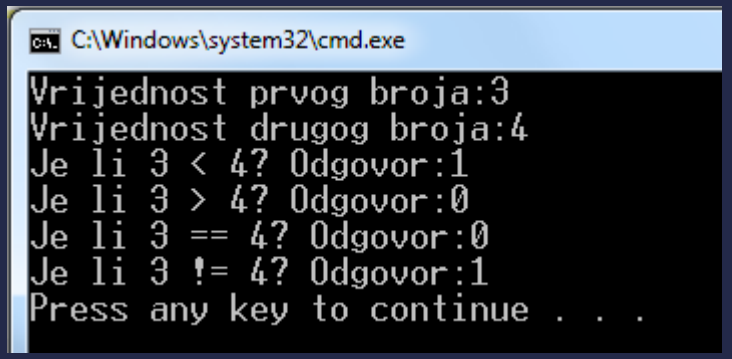
- Zadatak (operatori usporedbe):
  - Potrebno je unijeti dva cijela broja. Nakon toga se ti brojevi uspoređuju (<, >, ==, !=) i ispisuje se rezultat usporedbe.
  - Ispis neka bude oblika:

```
Vrijednost prvog broja=
Vrijednost drugog broja=
Je li.... < .... odgovor: ....
Je li.... > .... odgovor: ....
Je li.... == .... odgovor: ....
Je li.... != .... odgovor: ....
```

Primjer 4

Rješenje:

```c
#include<stdio.h>

void main(void) {

	int a,b;

	printf("Vrijednost prvog broja:");
	scanf("%d", &a);

	printf("Vrijednost drugog broja:");
	scanf("%d", &b);

	printf("Je li %d < %d? Odgovor:%d\n", a, b, a<b);
	printf("Je li %d > %d? Odgovor:%d\n", a, b, a>b);
	printf("Je li %d == %d? Odgovor:%d\n", a, b, a==b);
	printf("Je li %d != %d? Odgovor:%d\n", a, b, a!=b);
}
```



Operatori obnavljajućeg pridruživanja

- Operatori obnavljajućeg pridruživanja omogućavaju kraći zapis aritmetičkih izraza.
- Sastoje se od znaka jednakosti i odgovarajućeg aritmetičkog operatora.

  Npr.

  izraz a=a+8, može se zapisati kao a+=8.

Operatori obnavljajućeg pridruživanja

- Neki od operatora obnavljajućeg pridruživanja:

+ =	– =	* =	/ =	% =
a = a + ...	a = a –	a = a *	a = a /	a = a % ....

Primjer 5

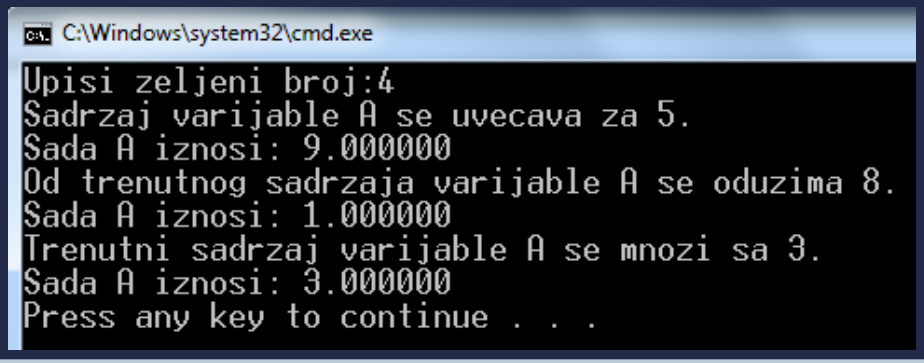
- Zadatak (operatori obnavljajućeg pridruživanja):
  - Potrebno je unijeti realni broj i pridružiti ga varijabli A. Sadržaj varijable A prvo treba uvećati za 5, pa umanjiti za 8, na kraju pomnožiti sa 3. Koristiti operatore obnavljajućeg pridruživanja. Ispis neka bude oblika:

```
Upisi zeljeni broj:
Sadrzaj varijable A se uvecava za 5.
Sada A iznosi: ....
Od trenutnog sadrzaja varijable A se
oduzima 8. Sada A iznosi: ....
Trenutni sadrzaj varijable A se mnozi
sa 3. Sada A iznosi: ....
```

Primjer 5

Rješenje:

```c
#include<stdio.h>

void main(void) {

	float a;

	printf("Upisi zeljeni broj:");
	scanf("%f", &a);

	a+=5;
	printf("Sadrzaj varijable A se uvecava za 5.\nSada A iznosi: %f\n", a );

	a-=8;
	printf("Od trenutnog sadrzaja varijable A se oduzima 8.\nSada A iznosi: %f\n", a );

	a*=3;
	printf("Trenutni sadrzaj varijable A se mnozi sa 3.\nSada A iznosi: %f\n", a );

}
```



Tip podatka operanada i rezultata

- Tip rezultata aritmetičkog izraza ovisi o tipovima operanada iz izraza.

  (Ako su operandi u izrazu tipa float i rezultat aritmetičkog izraza je također tog tipa.)

Tip podatka operanada i rezultata

- Kada se u izrazu nađe više različitih tipova operanada, tip rezultata aritmetičkog izraza ovisi o definiranim pravilima pretvorbe.
- Podaci se prvo svode na zajednički tip, prije zadane operacije.
- Pravila pretvorbe različitih tipova podataka usmjerena su prema višem tipu podataka.

Tip podatka operanada i rezultata

- Primjer pokazuje moguću grešku:

```
int a;
float b = 3.5;
float c = 5.0;
int d = 2;
a = b * c / d;
```

- Podaci se prvo svode na zajednički tip i to viši, float. Rezultat izraza je 8.75
- Pošto se rezultat pohranjuje u varijablu **a** koja je određena kao cjelobrojna (int) bit će pohranjena samo vrijednost 8.

Tip podatka operanada i rezultata

- Da bi se izbjegla moguća greška i neočekivani rezultati treba nastojati ne miješati varijable različitih tipova.

Primjer 6

- Zadatak (svođenje rezultata na zajednički tip s operandima):
  - Treba izračunati kvocijent dvaju cijelih brojeva i spremiti ga u realnu varijablu.
  - Ispis neka bude oblika:

```
Unesi prvi broj:
unesi drugi broj:
kvocijent iznosi: ....
```

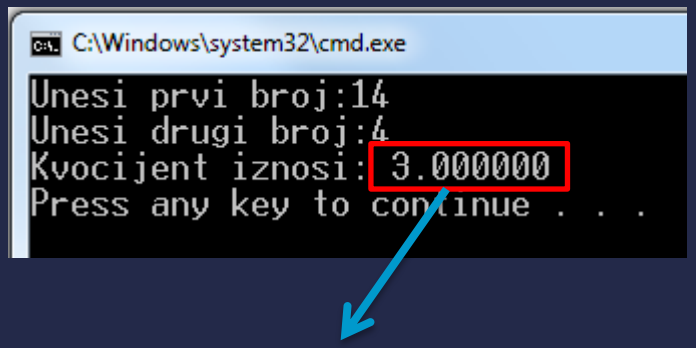
Primjer 6

- Napomene:
  - Varijable a i b deklarirati kao cjelobrojne (**int**).
  - Deklarirati varijablu kvocijent (tip **float**) za pohranu rezultata dijeljenja.

Primjer 6

Rješenje:

```c
#include<stdio.h>

void main(void) {

    int a, b;
    float c;

    printf("Unesi prvi broj:");
    scanf("%d", &a);

    printf("Unesi drugi broj:");
    scanf("%d", &b);

    c=a/b;

    printf("Kvocijent iznosi: %f\n", c );

}
```



Očekivan rezultat bi bio:
14/4=3,5

Budući da su i a i b tipa int konačan rezultat se zaokružuje na 3.
Iako je varijabla c tipa float, u ovom slučaju samo je zapis drugačiji (3.000000).

Primjer 6

- Alternativno rješenje:
  a) Deklarirati jednu od zadanih varijabli (a ili b) kao float, kako bi rezultat dijeljenja također bio realan broj (float).
  b) Koristiti cast operator u naredbi
- Cast operator
  - Cast operator se koristi kako bi neki podatak ”zamaskirali” drugim tipom podatka.
  - Npr.:

```
int a;
a = (float) a;
```

Primjer 6a

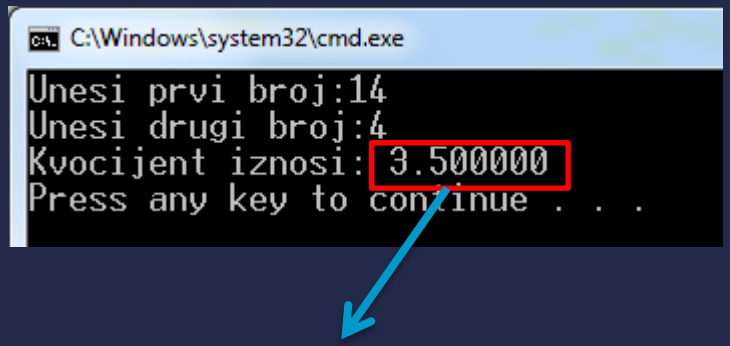
- Zadatak (svođenje operanada na zajednički tip):
  - Treba izračunati kvocijent dva broja od kojih je jedan cijeli, a drugi realan. Rezultat spremiti u realnu varijablu.
  - Ispis neka bude oblika:

```
Unesi prvi broj:
unesi drugi broj:
Kvocijent iznosi: …
```

Primjer 6a

Rješenje:

```c
#include<stdio.h>

void main(void) {

    int a;
    float b;
    float c;

    printf("Unesi prvi broj:");
    scanf("%d", &a);

    printf("Unesi drugi broj:");
    scanf("%f", &b);

    c=a/b;

    printf("Kvocijent iznosi: %f\n", c );

}
```



Očekivan rezultat bi bio:
14/4=3,5

Budući da smo deklarirali a kao int, a b kao float, konačan rezultat će biti također tipa float (3,500000).

Primjer 6b

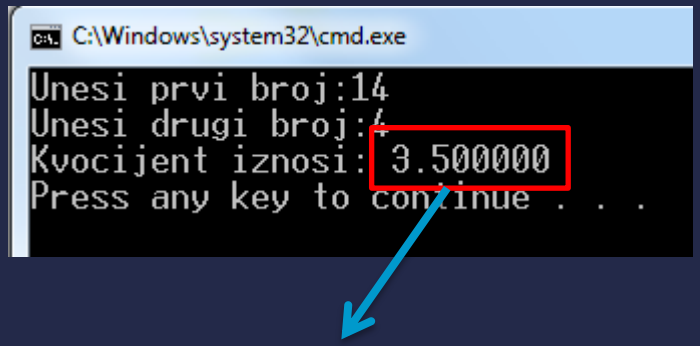
- Zadatak (korištenje cast operatora):
  - Treba izračunati kvocijent dva cijela broja od kojih je jedan cijeli, a drugi realan. Rezultat spremiti u cjelobrojnu varijablu.
  - Ispis neka bude oblika:

```
Unesi prvi broj:
unesi drugi broj:
Kvocijent iznosi: …
```

Primjer 6b

Rješenje:

```c
#include<stdio.h>

void main(void) {

    int a, b;
    float c;

    printf("Unesi prvi broj:");
    scanf("%d", &a);

    printf("Unesi drugi broj:");
    scanf("%d", &b);

    c=(float)(a)/b;

    printf("Kvocijent iznosi: %f\n", c );

}
```



Očekivan rezultat bi bio:
14/4=3,5

Budući da smo deklarirali a kao int, a b maskirali preko cast operatora kao float, konačan rezultat će biti također tipa float (3,500000).