Operatori i izrazi

Slide 1

Aritmetički operatori

Operator	Značenje
`+`	zbrajanje
`-`	oduzimanje
`*`	množenje
`/`	dijeljenje
`%`	modulo

Aritmetički operatori djeluju na numeričkim operandima tj. cjelobrojnim, realnim i znakovnim operandima (znakovna konstanta ima cjelobrojni tip).

Slide 2

Programiranje (C) Ak. god. 2003/04

Cjelobrojno dijeljenje

Operacije dijeljenja u slučaju cjelobrojnih operanada ima značenje *cjelobrojnog dijeljenje*. Rezultatu dijeljenja se odsjecaju decimalne znamenke kako bi se dobio cjelobrojni rezultat. To je tzv. zaokruživanje prema nuli.

```
  3/2 = 1
3.0/2 = 1.5
 -3/2 = -1
```

Slide 3

Modulo

Operacija modulo (`%`) djeluje na cjelobrojnim operandima i kao rezultat daje ostatak pri cjelobrojnom dijeljenju operanada. Na primjer,

```
x=10, y=3  ==>   x / y =3   i   x % y = 1
```

Uvijek vrijedi (osim za `y=0`)

```
(x / y) * y + x % y  == x
```

Slide 4

M. Jurak pred2.tex

Konverzije

Kada u aritmetičkom izrazu sudjeluju operandi različitih tipova onda prije izračunavanja izraza dolazi konverzije operanada u najširi tip prisutan u izrazu; rezultat aritmetičkog izraza bit će tog tipa.

```
double x=2.0;
float y=3.0;
int z;
........
z = x + y;
```

Ovdje se `y` prije zbrajanja pretvara u `double`. Rezultat zbrajanja `x+y` je tipa `double`.

Pravila konverzije:

Slide 5

- `short` i `char` (s predznakom ili bez) automatski se konvertiraju u `int` ili `unsigned int` ako je to potrebno. Naime, ako je `short` isto što i `int`, onda će `unsigned short` biti širi od `int` pa će operandi biti konvertirani u `unsigned int`.
- U svakoj operaciji koja uključuje operanda različitih tipova prije izvršenja operacije vrši se konverzija operanada u najširi tip.
- Tipovi su prema širini poredani na sljedeći način (od najšireg prema najužem): `long double`, `double`, `float`, `unsigned long long`, `long long`, `unsigned long`, `long`, `unsigned int` i `int`. Jedina iznimka je kad su `long` i `int` isti. Tada je `unsigned int` širi od `long`. Uži tipovi se ne pojavljuju jer se oni automatski konvertiraju u `int` ili `unsigned int` prema prvom pravilu.

Slide 6

Programiranje (C) Ak. god. 2003/04

Konverzije se dešavaju:

- U aritmetičkim izrazima s operandima različitih tipova;
- U izrazu pridruživanja ako tip lijeve strane nije isti kao tip desne strane;
- Pri prijenosu argumenata funkciji ako se stvarni i formalni argmenti razlikuju;
- Pri prijenosu vrijednosti iz funkcije u pozivni program.

Slide 7

Konverzije pri prijenosu argumenata:

- Ako funkcija nema prototipa, onda se svaki stvarni argument funkcije tipa `char` i `short` transformira u `int`, a `float` u `double`.
- Ukoliko funkcija ima prototip, onda se svi stvarni argumenti pri pozivu konvertiraju (ako je to potrebno) u tipove deklarirane u prototipu.

U sljedećem primjeru dolazi do konverzije iz `doubla` u `float` i zbog toga do gubitka preciznosti:

```
void f(float);
........
f(2.0); /* konstanta je tipa double */
```

Slide 8

M. Jurak pred2.tex

Eksplicitne konverzije (cast operator)

Sintaksa konverzije je

```
(tip_podatka) izraz;
```

U primjeru

```
double x;
float y;
........
x= (double) y;
```

`y` je eksplicitno konvertiran u tip `double` prije pridruživanja.

Primjer:

```
int i;
double x;
((int) (i + x)) % 2;
```

Slide 9

Prioriteti operacija

Svi su operatori hijerhijski grupirani prema svom prioritetu. Npr.

```
x=2+4/2
```

daje `x=4` dok

```
x=(2+4)/2
```

daje `x=3`.

Kategorija	Operatori	asocijativnost
aritmetički op.	`* / %`	L → D
aritmetički op.	`+ -`	L → D
op. pridruživanja	`=`	D → L

Slide 10

Asocijativnost

Zagrade koristimo kako bismo grupirali operande oko operatora. Ukoliko ih nema, onda se grupiranje vrši oko operatora najvišeg prioriteta. Kada više operatora ima isti, najviši, prioritet, onda način izračunavanja izraza ovisi o asocijativnosti. Asocijativnost može biti slijeva na desno ili zdesna na lijevo. Ako imamo, na primjer, dva operatora istog prioriteta, čija je asocijativnost slijeva na desno, onda će se operandi prvo grupirati oko lijevog operanda. Na primjer,

```
a - b + c    je ekvivalntno s  (a - b) + c
```

Slide 11

Operatori inkrementiranja i dekrementiranja

Operator inkrementiranja `++` povećava vrijednost varijable za jedan. Izraz

```
x++;
```

ekvivalentan je izrazu

```
x=x+1;
```

Operator dekrementiranja `--` smanjuje vrijednost varijable za jedan. Izraz

```
x--;
```

ekvivalentan je izrazu

```
x=x-1;
```

Slide 12

Prefiks i postfiks forma

Operatore inkrementiranja/dekrementiranja moguće je pisati ispred i iza varijable:

```
++x;  x++; --x;  x--;
```

- U prefiks notaciji (`++x, --x`) varijabla će biti promijenjena prije no što će njena vrijednost biti iskorišten u složenom izrazu;
- U postfiks notaciji (`x++, x--`) varijabla će biti promijenjena nakon što će njena vrijednost biti iskorišten u složenom izrazu.

Slide 13

Primjeri

```
x=3;
y=++x;   /* daje y=4, x=4  */
y=x++;   /* daje y=4, x=5  */
```

```
i=7;
printf("i= %d\n", --i);  /* ispisuje i=6  */
printf("i= %d\n", i--);  /* ispisuje i=6  */
printf("i= %d\n", i  );  /* ispisuje i=5  */
```

Slide 14

Programiranje (C) Ak. god. 2003/04

Operator sizeof

`sizeof` je operator koji daje veličinu svog operanda u bajtovima. Operand može biti izraz ili tip podatka.

```
int i;
float x;
printf("Velicina tipa int  = %d\n", sizeof(i));
printf("Velicina tipa float  = %d\n", sizeof(x));
```

Isti efekt postižemo ako `sizeof` primijenimo na tip podatka:

```
printf("Velicina tipa int = %d\n", sizeof(int));
printf("Velicina tipa float = %d\n", sizeof(float));
```

`sizeof(char)` je uvijek jednak 1.

Slide 15

Primjena na nizove

```
char tekst[]="Dalmacija";

printf("Broj bajtova u varijabli tekst =%d\n",
                                sizeof(tekst));
```

daje

```
Broj bajtova u varijabli tekst =10
```

Operator `sizeof` vraća cjelobrojnu vrijednost bez predznaka koja ovisi o implementaciji. Taj je tip definiran u datoteci zaglavlja `<stddef.h>` i zove se `size_t`.

Slide 16

M. Jurak pred2.tex

Tablica prioriteta (nepotpuna)

Kategorija	Operatori	asocijativnost
unarni op.	`++ -- - * & (type) sizeof`	D $\rightarrow$ L
aritmetički op.	`* / %`	L $\rightarrow$ D
aritmetički op.	`+ -`	L $\rightarrow$ D
op. pridruživanja	`=`	D $\rightarrow$ L

Slide 17

Relacijski operatori

Relacijski operatori:

Operator	Značenje
$<$	strogo manje
$<=$	manje ili jednako
$>$	strogo veće
$>=$	veće ili jednako
$==$	jednako
$!=$	različito

Slide 18

Programiranje (C) Ak. god. 2003/04

Logički izrazi

Pomoću ovih šest operatora formiraju se logički izrazi. Njihova vrijednost je istina (=1) ili laž (=0).

Za `i=1, j=2, k=4` imamo

Izraz	Istinitost	Vrijednost
`i<j`	istinito	1
`(i+j)>=k`	neistinito	0
`i==2`	neistinito	0
`k!=i`	istinito	1

Slide 19

Prioritet logičkih operatora je niži od prioriteta aritmetičkih operatora.

Izraz

```
i >= 'A'-'a'+1
```

je ekvivalentan s

```
i >= ('A'-'a'+1)
```

Slide 20

M. Jurak pred2.tex

Logički operatori

Složeniji logički izrazi formiraju se pomoću logičkih operatora:

<table>
<tr><th>Operator</th><th>Značenje</th></tr>
<tr><td>&&</td><td>logičko I</td></tr>
<tr><td>||</td><td>logičko ILI</td></tr>
<tr><td>!</td><td>logička negacija (unarno)</td></tr>
</table>

Operandi logičkih operatora su logičke vrijednosti (najčešće logički izrazi) s tim da se svaka cjelobrojna vrijednost različita od nule interpretira kao istina, a nula se interpretira kao laž. Vrijednost složenog logičkog izraza bit će 0 (laž) ili 1 (istina).

Slide 21

Primjer. Ako je izraz `i>1` istinit, izraz `c=='t'` istinit, a izraz `j<6` lažan, onda imamo:

<table>
<tr><th>Izraz</th><th>Istinitost</th><th>Vrijednost</th></tr>
<tr><td><code>i>1 || j<6</code></td><td>istinito</td><td>1</td></tr>
<tr><td><code>i>1 && j<6</code></td><td>neistinito</td><td>0</td></tr>
<tr><td><code>!(i>1)</code></td><td>neistinito</td><td>0</td></tr>
<tr><td><code>i>1 || (j<6 && c!='t')</code></td><td>istinito</td><td>1</td></tr>
</table>

Slide 22

Programiranje (C) Ak. god. 2003/04

Tablica prioriteta (nepotpuna)

<table>
<tr><th>Kategorija</th><th>Operatori</th><th>asocijativnost</th></tr>
<tr><td>unarni op.</td><td><code>! ++ -- - * & (type) sizeof</code></td><td>D → L</td></tr>
<tr><td>aritmetički op.</td><td><code>* / %</code></td><td>L → D</td></tr>
<tr><td>aritmetički op.</td><td><code>+ -</code></td><td>L → D</td></tr>
<tr><td>relacijski op.</td><td><code>< <= > >=</code></td><td>L → D</td></tr>
<tr><td>op. jednakosti</td><td><code>== !=</code></td><td>L → D</td></tr>
<tr><td>logičko I</td><td><code>&&</code></td><td>L → D</td></tr>
<tr><td>logičko ILI</td><td><code>||</code></td><td>L → D</td></tr>
<tr><td>op. pridruživanja</td><td><code>=</code></td><td>D → L</td></tr>
</table>

Slide 23

Izračunavanje logičkih izraza

Logički izrazi se izračunavaju slijeva na desno i prestaju se izračunavati kad je vrijednost izraza poznata. Primjer:

```
char x[128];
for(i=0; i<128 && x[i] !='a'; ++i) {
....
}
```

neće doći do ispitivanja `x[i] !='a'` ako je `i=128`. S druge strane, kôd

```
for(i=0; x[i] !='a' && i<128; ++i) { /* GRESKA */
....
}
```

bi eventualno ispitivao da li je `x[128]` različito od `'a'`.

Slide 24

M. Jurak pred2.tex

Operatori pridruživanja

Osnovni operator pridruživanja je `=`. Naredba pridruživanja je oblika

```
varijabla = izraz;
```

Na primjer,

```
i=2;
a=3.17;
c='m';
```

Pridruživanje `varijabla = izraz` je ustvari izraz i stoga može biti dio kompleksnijeg izraza.

Slide 25

Izraz `varijabla = izraz` ima vrijednost varijable na lijevoj strani nakon što se izvrši pridruživanje. Stoga je moguće koristiti pridruživanje unutar drugih naredbi kao u primjeru

```
while((a=x[i])!=0){
....
++i;
}
```

gdje se u testu `while` narebe prvo `x[i]` pridruži varijabli `a`, a zatim se testira da li je dobiveni izraz, a to je vrijednost varijable `a`, različit od nule.

Slide 26

Programiranje (C) Ak. god. 2003/04

Operatore pridruživanja moguće je ulančati i pisati

```
varijabla1 = varijabla2 = varijabla3 = ... = izraz;
```

jer je asocijativnost operatora pridruživanja `D->L`. To znači da se `izraz` jednom izračuna i zatim se redom pridružuje varijablama `.. varijabla3, varijabla2, varijabla1`. Na primjer,

```
x = y = cos(3.22);
```

je ekvivalentno s

```
x = (y = cos(3.22));
```

odnosno

```
y = cos(3.22);
x=y;
```

Slide 27

Složeni operatori pridruživanja

```
+=   -=   *=   /=   %=
```

Općenito izraz oblika

```
izraz1 op= izraz2;
```

gdje je `op` jedna od operacija `+, -, *, /, %`, ekvivalentan je s

```
izraz1 = izraz1 op izraz2;
```

Ovi operatori spadaju u istu prioritetnu grupu s operatorom `=` i izračunavaju se zdesna na lijevo. Prioritet im je manji od prioriteta aritmetičkih i logičkih operatora.

Slide 28

M. Jurak pred2.tex

Primjeri

Izraz	Ekvivalentan izraz
`i +=5`	`i=i+5`
`i -=j`	`i=i-j`
`i *=j+1`	`i=i*(j+1)`
`i /=4`	`i=i/4`
`i %=2`	`i=i%2`

Slide 29

Uvjetni operator : ?

Uvjetni izraz je izraz oblika

```
izraz1 ? izraz2 : izraz3;
```

U njemu se prvo izračunava `izraz1`. Ako je on istinit (različit od nule) onda se izračunava `izraz2` i on postaje vrijednost čitavog uvjetnog izraza; ako je `izraz1` lažan (jednak nuli) onda se izračunava `izraz3` i on postaje vrijednost čitavog uvjetnog izraza.

Na primjer,

```
double a,b,min;
 ........
min=(a<b) ? a : b;
```

Uvjetni operator ima nizak prioritet tako da zagrade oko prvog, drugog i trećeg izraza najčešće nisu potrebne.

Slide 30

Programiranje (C) Ak. god. 2003/04

Tablica prioriteta (nepotpuna)

Kategorija	Operatori	asocijativnost
unarni op.	`! ++ -- - * & (type) sizeof`	D → L
aritmetički op.	`* / %`	L → D
aritmetički op.	`+ -`	L → D
relacijski op.	`< <= > >=`	L → D
op. jednakosti	`== !=`	L → D
logičko I	`&&`	L → D
logičko ILI	`||`	L → D
uvjetni ? : op.	`? :`	D → L
op. pridruživanja	`= += -= *= /= %=`	D → L

Slide 31

M. Jurak pred2.tex