Objektno orjentisano programiranje – C++

Tipovi podataka i operatori

Teme

- Identifikatori
- Specijalni znakovi
- Ugrađeni tipovi podataka
- Operatori
- Konverzija tipova
- Simboličke konstante

Identifikatori

- Identifikator je ime (promenljive, konstante, klase, ...)
- Identifikator može da bude sastavljen od slova engleskog alfabeta (A-Z, a-z), brojeva (0-9) i donje crte (_)
- Prvi znak mora da bude slovo ili donja crta
- Identifikator ne sme da bude neka ključna reč

Ključne reči jezika C++

asm	do	if	return	typedef
auto	double	inline	short	typeid
bool	dynamic_cast	int	signed	typename
break	else	long	sizeof	union
case	enum	mutable	static	unsigned
catch	explicit	namespace	static_cast	using
char	export	new	struct	virtual
class	extern	operator	switch	void
const	false	private	template	volatile
const_cast	float	protected	this	wchar_t
continue	for	public	throw	while
default	friend	register	true	
delete	goto	reinterpret_cast	try	

Imena (identifikatori)

- Izbegavati nazive koji počinju znakom (_) jer se tako obično označavaju sistemske promenljive
- Nazivi koji počinju velikim slovom po pravilu su rezervisani za klase, a oni koji počinju malim za promenljive
  - nije dozvoljena primena znakova (č, š, ž, ć i sl.) u identifikatorima
  - ne postoji ograničenje u broju znakova identifikatora, ali ne treba preterivati
- Mada **main** nije ključna reč, ponašati se kao da jeste

Ne postoji ograničenje u dužini identifikatora??

Identifikatore:

- Snjeguljica_i_7_patuljaka
- Snjeguljica_i_sedam_patuljaka

kompajler će (ukoliko prepoznaje samo prvih 14 znakova) interpretirati kao iste.

Ako kompajler prepoznaje samo prvih 14 znakova																											
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
S	n	j	e	g	u	l	j	i	c	a	_	i	_	7	_	p	a	t	u	l	j	a	k	a				
S	n	j	e	g	u	l	j	i	c	a	_	i	_	s	e	d	a	m	_	p	a	t	u	l	j	a	k	a

Identifikatori

- Jezik C++ razlikuje mala i velika slova (engl. case sensitive)
  - maliNarodiVelikeIdeje
  - MaliNarodiVelikeIdeje
  - maliNarodiVELIKEIDEJE
- Primeri loših imena:
  - 7Up, Ana-Marija, Ana Marija
  - Snezana i 7 Patuljaka
  - MojaPrivatnaPromenljiva
  - _system

Specijalni znaci

Specijalni znak	Naziv	Opis
\n	newline	Pomera kursor u sledeći red
\t	tab	Pomera kursor na sledeći tabulator
\a	alarm	Računar se oglašava
\b	backspace	Vraća kursor za jednu poziciju unazad
\r	carriage return	Pomera kursor na početak tekućeg reda
\\	backslash	Prikazuje obrnutu kosu crtu
\'	single quote	Prikazuje apostrof
\''	double quote	Prikazuje navodnik
\?	question mark	Prikazuje znak pitanja

Tipovi podataka

- Tip objekta određuje i raspored bitova prema kojem je objekat sačuvan u memoriji
- C++ nudi ugrađene tipove podataka koji odgovaraju:
  - celim brojevima,
  - realnim brojevima u pokretnom zarezu (floating-point)
  - logičkim vrednostima
- C++ omogućava konstruisanje mnogo složenijih tipova: nabrajanja, strukture i klase

Osnovni ugrađeni tipovi

Tip	Značenje
char	Karakter ili znak
w_char	Široki karakter, više od jednog bajta
int	Celi broj
float	Broj sa pomičnom tačkom
double	Decimalni broj u dvostrukoj preciznosti
bool	Logička vrednost
void	Bez vrednosti

Osnovni tipovi (2)

- C++ dozvoljava proširivanje osnovnih tipova navođenjem modifikatora ispred naziva tipa podataka
- Postoje 4 modifikatora tipa:
  - *signed,*
  - *unsigned,*
  - *long,*
  - *short*

Osnovni tipovi (3)

Osnovni tipovi mogu biti:

- char, signed char, unsigned char
- int:
  - short, int, long
- int:
  - signed int, unsigned int
- float:
  - double, long double
- void
- nabrajanje (engl. enumerated)

Osnovni (ugrađeni) tipovi podataka

Tip	Veličina u bajtovima	Opseg vrednosti
bool	1	true ili false
char	1	-128 do +127
unsigned char	1	0 do 255
short	2	-32 768 do +32767
unsigned short	2	0 do 65 535
int	4	-2 147 483 648 do 2 147 483 647
unsigned int	4	0 do 294 967 295
long	4	-2 147 483 648 do 2 147 483 647
unsigned long	4	0 do 294 967 295
float	4	$\pm 3{,}4 \times 10^{\pm 38}$ sa tačnošću od približno 7 cifara
double	8	$\pm 1{,}7 \times 10^{\pm 308}$ sa tačnošću od približno 15 cifara
long double	8	$\pm 1{,}7 \times 10^{\pm 308}$ sa tačnošću od približno 15 cifara

Opseg osnovnih tipova (ANSI/ISO standard)

Tip	Veličina u bajtovima	Opseg definisan ANSI/ISO standardom
bool	1	true ili false
char	1	-128 do +127
unsigned char	1	0 do 255
signed char	1	-128 do +127
short int	2	-32 768 do +32767
unsigned short int	2	0 do 65 535
signed short int	2	isto kao short int
int	4	-2 147 483 648 do 2 147 483 647
unsigned int	4	0 do 4 294 967 295
signed int	4	isto kao int
long int	4	-2 147 483 648 do 2 147 483 647
signed long int	4	isto kao long int
unsigned long int	4	0 do 4 294 967 295
float	4	$\pm 3{,}4 \times 10^{\pm 38}$ sa tačnošću od približno 7 cifara
double	8	$\pm 1{,}7 \times 10^{\pm 308}$ sa tačnošću od približno 15 cifara
long double	8	isto kao double

header <cfloat> (float.h)

- Karakteristike tipa broja sa pomičnom tačkom (engl. floating-point)
- Zaglavlje cfloat opisuje karakteristike tipova sa pomičnom tačkom za određeni sistem i implemetaciju prevodioca.

header <climits> (limits.h)

- Veličine osnovnih tipova

Macro constants

name	expresses	value*
CHAR_BIT	Number of bits in a char object (byte)	8 or greater*
SCHAR_MIN	Minimum value for an object of type signed char	-127 (-2^7+1) or less*
SCHAR_MAX	Maximum value for an object of type signed char	127 (2^7-1) or greater*
UCHAR_MAX	Maximum value for an object of type unsigned char	255 (2^8-1) or greater*
CHAR_MIN	Minimum value for an object of type char	either SCHAR_MIN or 0
CHAR_MAX	Maximum value for an object of type char	either SCHAR_MAX or UCHAR_MAX
MB_LEN_MAX	Maximum number of bytes in a multibyte character, for any locale	1 or greater*
SHRT_MIN	Minimum value for an object of type short int	-32767 ($-2^{15}+1$) or less*
SHRT_MAX	Maximum value for an object of type short int	32767 ($2^{15}-1$) or greater*
USHRT_MAX	Maximum value for an object of type unsigned short int	65535 ($2^{16}-1$) or greater*
INT_MIN	Minimum value for an object of type int	-32767 ($-2^{15}+1$) or less*
INT_MAX	Maximum value for an object of type int	32767 ($2^{15}-1$) or greater*
UINT_MAX	Maximum value for an object of type unsigned int	65535 ($2^{16}-1$) or greater*
LONG_MIN	Minimum value for an object of type long int	-2147483647 ($-2^{31}+1$) or less*
LONG_MAX	Maximum value for an object of type long int	2147483647 ($2^{31}-1$) or greater*
ULONG_MAX	Maximum value for an object of type unsigned long int	4294967295 ($2^{32}-1$) or greater*
LLONG_MIN	Minimum value for an object of type long long int	-9223372036854775807 ($-2^{63}+1$) or less*
LLONG_MAX	Maximum value for an object of type long long int	9223372036854775807 ($2^{63}-1$) or greater*
ULLONG_MAX	Maximum value for an object of type unsigned long long int	18446744073709551615 ($2^{64}-1$) or greater*

Opseg vrednosti osnovnih tipova

```
#include <iostream>
#include <climits>
#include <cfloat>
int main() {
   using namespace std;  → aktivira se imenik standardnih funkcija
   cout<<"Najmanji int "<<INT_MIN<<endl;
   cout<<"Najveci int "<<INT_MAX<<endl;
   cout<< "Najmanji long "<<LONG_MIN<<endl;
   cout<< "Najveci long "<<LONG_MAX<<endl;
   cout<< "Najmanji float "<<FLT_MIN<<endl;
   cout<< "Najveci int "<<FLT_MIN<<endl;
   cout<< "Najmanji double "<<DBL_MIN<<endl;
   cout<< "Najveći double "<<DBL_MAX<<endl;
   return 0;
}
```

Opseg vrednosti osnovnih tipova

```
#include <iostream>
#include <climits>
#include <cfloat>
int main() {
   using namespace std;  → aktivira se imenik standardnih funkcija
   cout<<Najmanji int -2147483648
   cout<<Najveći int 2147483647
   cout<<Najmanji long -2147483648
   cout<<Najveci long 2147483647
   cout<<Najmanji float 1.17549e-038
   cout<<Najveci int 1.17549e-038
   cout<<Najmanji double 2.22507e-308
   cout<<Najveci double 1.79769e+308
   return 0;
}
```

Tipovi short, int, long

```
#include <iostream>
int main()
{
  using namespace std;
  cout << "The size of an int is:" << sizeof(int)<< " bytes.\n";
  cout << "The size of a short int is:" << sizeof(short)<<" bytes.\n";
  cout << "The size of a long int is:"<< sizeof(long) <<" bytes.\n";
  return 0;
 }
```

```
The size of an int is:4 bytes.
The size of a short int is:2 bytes.
The size of a long int is:4 bytes.
```

Osnovni (ugrađeni) tipovi podataka

- Svi podaci i promenljive u programu moraju biti definisanog tipa (C++ je “strogo tipiziran” jezik)
- Svi osnovni tipovi su standardno označeni (signed)
- Veličina nekog osnovnog tipa u bajtovima može se saznati primenom operatora **sizeof**
- Deklaracije promenljivih se mogu nalaziti bilo gde u programu ali pre prvog korištenja promenljive
- Promenljiva se deklariše navođenjem imena i tipa: *int vreme;*

Primer tipova podataka

int i; → *deklaracija promenljive*

int j=1, k=2; → *deklaracija i inicijalizacija promenljive*

float f1=3.4, f2=0.0;

double PI=3.14;

char a='a', nul='0';

Tipovi int, float i double

```
#include <iostream>
using namespace std;
int main()
{
   cout<<(float)1/3<<endl;
   cout<<1/3.f<<endl;
   cout<<1/3<<endl;
   return 0;
}
```

```
0.333333
0.333333
0
```

Bez .f broj se interpretira kao int, 1/3 = (int)1/(int)3 => (int)0 umesto da bude (float) 0.333333.
Ovde .f kaže prevodiocu da interpretira literal kao float broj.

Tip char

- Promenljive tipa char sadrže 8-bitne ASCII znakove, koji se navode pod apostrofima
- Tip char se može koristiti i kao “mali” int opsega od -128 do +127
- Postoji i tip w_char (wide character) koji se koristi za Unikod (engl. Unicode) i internacionalizaciju

char a = `a`;

cout << a;

```
char slovoa = 'a';
cout << 'b';
```

Tip string

```
#include <iostream>
#include <string>
using namespace std;
int main ()
{
    string mystring;
    mystring = "Ovo je string";
    cout << mystring<<endl;
    return 0;
}
```

```
Ovo je string
```

Tip nabrajanja (engl. enumerated type)

- Ideja za tip nabrajanja je kreiranje novih tipova podataka koji imaju ograničen broj vrednosti
- Takođe, ove vrednosti su konstante, a ne brojevi.

```
enum wind_directions_t {NO_WIND,
NORTH_WIND, SOUTH_WIND, EAST_WIND,
WEST_WIND};
```

- Imena konstanti trebaju biti dovoljna za poređenje vrednosti...

Tip nabrajanja: Primer

```
#include <iostream>
using namespace std;
int main ()
{
   enum wind_directions_t {NO_WIND, NORTH_WIND,
   SOUTH_WIND, EAST_WIND, WEST_WIND};
   wind_directions_t wind_direction = WEST_WIND;
   cout << static_cast<int>( wind_direction )<<endl;
   return 0;
}
```

Rezultat
4

Tip bool

- C++ definiše dve konstante, true i false
- Bilo koja vrednost različita od nule tumači se kao true, a vrednost 0 tumači se kao false;
  - ovo svojstvo je naročito korisno u logičkim izrazima
- Kada se koristi u izrazima koji nisu logički, važi obrnuto: true se konvertuje u 1, a false u 0

Tip bool (2)

```
#include <iostream>
int main() {
    using namespace std;
    bool b=false;
    cout<<"b je:" <<b<<endl;
    b=true;
    cout<<"b je:"<<b<<endl;
    return 0;
}
```

```
b je:0
b je:1
```

Komentari

- Olakšavaju razumevanje programa, ali nisu obavezni
- Dve kose crte označavaju jednolinijski komentar, koji se proteže od znaka // do kraja reda
- Sve što počinje sa /* i završava sa */ je komentar

```
// ovo je jednolinijski komentar

/* a ovo je komentar u C-stilu
   koji se proteze u dva reda */
```

Promenljive (varijable)

- Osnovna jedinica za čuvanje podataka u programu
- Mesto u memoriji za zapis neke proste vrednosti ili pokazivača na objekat
- Naziva se promenljiva zato što tokom izvršavanja programa to mesto može da sadrži različite vrednosti (istog tipa)

Promenljive (2)

- Svaka promenljiva se mora deklarisati pre prve upotrebe, bilo gde u programu
- Deklaracija promenljive: tip ime*;*
  tip ime = početna vrednost;
- Promenljiva sadrži vrednost ili je pokazivač na objekat.
- Ako sadrži vrednost, ta vrednost je literal (primitivni tip).
- Dodeljivanje početne vrednosti zove se inicijalizacija
- Definicija promenljive podrazumeva rezervisanje prostora za promenljivu u memoriji

Konstante

- Vrednosti koje predstavljaju same sebe, tj. fiksne vrednosti bilo kog tipa

Tip	Primeri literala
char	’A’, ’Z’, ’8’, ’*’
int	-77, 65, 0x9FE
unsigned int	10U, 64000U
long	-77L, 65L, 12345L
unsigned long	5UL
float	3.14f
double	1.414
bool	true, false

Konstante (2)

Konstante mogu biti literali i simbolične konstante.

- **Literalne konstante**

int b =25,

- **Simbolične konstante**

brojStudenata = brojUcionica * 30;

Definisanje konstanti

- **Korišćenjem “#define”**

```
#define brojStudenata 30;
```

- **Korišćenjem “const”**

```
const  unsigned short  brojStudenata = 30;
```

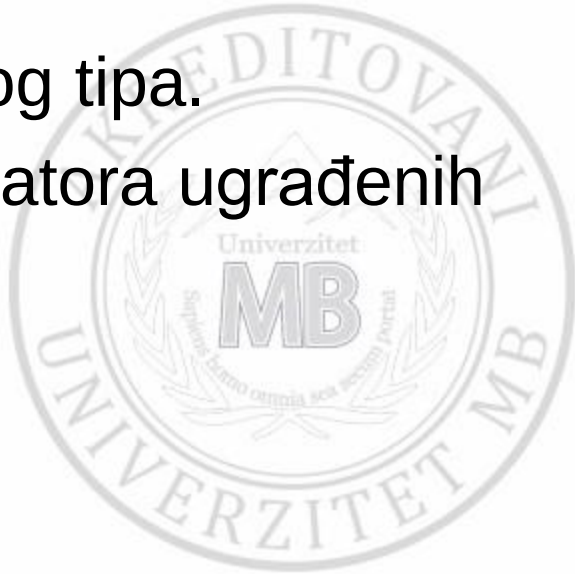
- Vrednost konstante mora biti inicijalizovana tokom deklaracije

Konstante (3)

- Heksadecimalne konstante počinju sa 0x
- Sve konstante koje počinju sa 0 prevodilac tretira kao oktalne brojeve
  int broj1 = 0x0C; //12 decimalno
  int broj2= 010; //8 decimalno
- Ako tip literala nije zadat, C++ kompajler standardno celoj vrednosti dodeljuje najmanji celobrojni tip u koji ona može da stane, dok su realne konstante standardno tipa double

Izrazi

- Izraz je iskaz u programu koji sadrži operande (objekte, funkcije ili literale nekog tipa) i operacije nad tim operandima
  - generiše rezultat tačno definisanog tipa.
  - operacije se zadaju pomoću operatora ugrađenih u jezik.



Operatori

- Operator dodele
- Aritmetički operatori
- Relacioni i logički operatori
- Operator nabrajanja (razdvajanja)
- Operator *sizeof*

Operator dodele

vrijednost = izraz;

- Moguća je i lančana dodela

```
int x,y,z;
x=y=z=10;
```

- Izraz tipa *vrednost = vrednost + izraz* može se napisati kao *vrednost += izraz*;
- Treba zapamtiti suštinsku razliku između jednostrukog znaka jednakosti (=) koji je simbol za dodelu, i dvostrukog znaka jednakosti (==) koji je simbol za poređenje

Aritmetički operatori

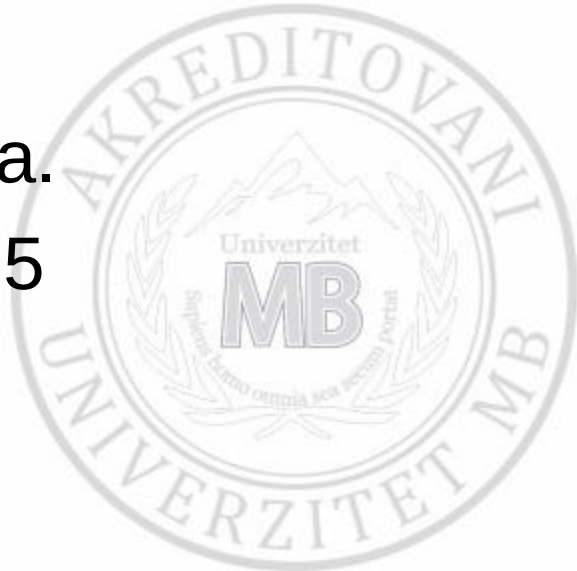
- Operator moduo (modulo) (%) može se primenjivati isključivo na celobrojne promenljive

Operator	Značenje
+, -, *, /	sabiranje, oduzimanje, množenje, deljenje
%	modulo (ostatak celobrojnog deljenja)
++	inkrement (uvećava za 1)
--	dekrement (umanjuje za 1)

Operacija deljenja

- Da bi se izbegle nedoumice oko tipa rezultata, preoručljivo je pisati floating point konstante sa tačkama, čak i kada nemaju decimalnih mesta

```
float rezultat = 3 / 4;   //rezultat je nula.
rezultat =3. / 4. ;       //rezultat je 0.75
```



Inkrement i dekrement

- Ove operatore treba koristiti kad god je moguće (generišu efikasan kod)

Primer

```
int brojac;
brojac = 0;
//sledeće tri naredbe su identične
brojac = brojac +1;
brojac++;
++brojac;
```

Inkrement i dekrement (2)

- Razlikujemo prefiksnu (++i) i postfiksnu (i++) notaciju

```
#include <iostream>
int main() {
    using namespace std;
    int i=100;
    int j=++i;
    cout<<"t=0, j ="<<j<<endl;
    int k=100;
    j =k++;
    cout<<"t=1, j ="<<j<<endl;
    return 0;
}
```

```
t=0, j =101
t=1, j =100
```

Relacioni i logički operatori

Relacioni operator	Značenje
>, >=, <, <=	veće, veće ili jednako, manje, manje ili jednako
==	operator jednakosti
!=	operator nejednakosti

Logički operator	Značenje
&&	AND (i)
||	OR (ili)
!	NOT (negacija)

Operator razdvajanja ili nabrajanja (,)

- Vrednost niza izraza je ona koja je poslednja u nizu.

```
#include <iostream>
int main() {
  using namespace std;
  long l1=1, l2=2, l3=3, l4=4;
  l4= (l1=10, l2=20, l3=30);
  cout<<"l4 = "<<l4<<endl;
  return 0;
}
```

```
l4 = 30
```

Operator *sizeof*

```
#include <iostream>
int main()
{
  using namespace std;
  cout << "The size of an int is:" << sizeof(int)<< " bytes.\n";
  cout << "The size of a short int is:" << sizeof(short)<<" bytes.\n";
  cout << "The size of a long int is:"<< sizeof(long) <<" bytes.\n";
  return 0;
}
```

- Prikazuje dužinu određenog tipa u bajtovima

```
The size of an int is:4 bytes.
The size of a short int is:2 bytes.
The size of a long int is:4 bytes.
```

Prioritet operatora

- Redosled izvršavanja operatora je definisan, ali se uvek može izmeniti pomoću zagrada
- Dobra je navika stavljati zagrade i razmake uvek kada postoji dilema o hijerarhiji operatora

Operator
()
!, + (unarni), - (unarni), ++, --
*, /, %
+, -
<, <=, >, >=
==, !=
&&
||
=, +=, -=, *=, /=

Konverzija tipova u izrazima i dodelama

- Kada u izrazu dodele učestvuju operatori različitih tipova, vrednost desne strane (izraza) biće konvertovana u tip leve strane (tip promenljive)
- Kada u izrazu učestvuju operandi različitih tipova, svi se konvertuju u tip najvećeg operanda (tzv. unapređivanje tipa)

Konverzija tipova

Prioritet operatora (konverzije)	Tipovi operatora
	long double
↑	double
↑	float
↑	unsigned long
↑	long
↑	unsigned int
↑	int
↑	unsigned short
↑	short
↑	unsigned char
↑	char

Konverzija tipa: cast (static_cast)

static_cast <tip> (izraz) ili *(tip)izraz*

```
#include <iostream> int
main() {
  using namespace std;
  float pi=3.1415926;
  cout<<"Cijeli dio broja pi "<<static_cast<int>(pi) <<endl;
  cout<<"Cijeli dio broja pi "<<(int)pi<<endl;
  return 0;
}
```

```
Cijeli dio broja pi 3
Cijeli dio broja pi 3
```

Zaključak

- **Identifikator** je ime (promenljive, konstante, klase, ...): A-Z, a-z, 0-9 i donja crta (_)
- Prvi znak identifikatora mora da bude slovo ili donja crta i ne sme da bude neka ključna reč
- **Tipovi podataka:** char, w_char, int, float, double, bool, void
- **Promenljive (varijable)** čuvaju mesto u memoriji za zapis neke proste vrednosti ili pokazivača na objekat
- **Operatori:** operator dodele, aritmetički operatori, relacioni i logički operatori, operator nabrajanja (razdvajanja), operator *sizeof*
- **Konverzija tipova:** ako se u izrazu nađu operatori različitih tipova

Kraj prezentacije

HVALA NA PAŽNJI!