4. Osnovni podatkovni tipi

1

Osnovni podatkovni tipi

2

- podatkovni tip – vrsta podatka
  - prevajalniku pove:
    - za kakšno vrsto podatka gre
    - koliko pomnilnika potrebuje za shranjevanje tega podatka
- osnovni podatkovni tipi v C++ (te bomo uporabljali mi)
  - bool, int, double, char, wchar_t
- predznačenost številskih tipov lahko spremenimo z določiloma: signed / unsigned

Osnovni **celoštevilski** podatkovni tipi

3

ime tipa	velikost v zlogih (bitih)	možne vrednosti	opis
bool	1 zlog (8 bitov)	false ali true 0 ali 1	logična vrednost, true ali false
char	1 zlog (8 bitov)	(signed) -128 … 127 (unsigned) 0 … 255	predstavlja en znak razširjene ASCII tabele, gledano celoštevilsko pa eno od možnih vrednosti
short	2 zloga (16 bitov)	(signed) -32768 … 32767 (unsigned) 0 … 65536	cela števila v omenjenem razponu
int ali long	4 zlogi (32 bitov)	(signed) -2147483648 … 2147483647 (unsigned) 0 .. 0-4294967296	int je standardni celoštevilski tip na nekaterih arhitekturah je long 64 bitni
long long	8 zlogov (64 bitov)	(signed) -9223372036854775808 … 9223372036854775807 (unsigned) 0 … 18446744073709551616	za zelo velika cela števila

Osnovni **realni** podatkovni tipi

4

ime tipa	velikost v zlogih (bitih)	možne vrednosti	opis
float	4 zlogi (32 bitov)	+/- 1.4023×10^{-45} … $3.4028 \times 10^{+38}$	splošno realno število
double	8 zlogov (64 bitov)	+/- 4.9406×10^{-324} … 1.7977×10^{308}	realno število z večjo natančnostjo
long double	12 zlogov (96 bitov)		za zelo velika realna števila in z večjo natančnostjo

Velikosti podatkovnih tipov

5

- velikost podatkovnih tipov
  - število zlogov, ki so potrebni, da shranimo podatek tega tipa
  - izračunamo ga lahko z operatorjem `sizeof()`
- `PRIMER:`
  - `cout << "Velikost tipa int je " << sizeof(int);`

5. Deklaracija spremenljivk in prirejanje vrednosti

6

Deklaracija spremenljivke

7

- Kaj je deklaracija in kaj definicija?
  - deklaracija … napoved
  - definicija … določi tudi vsebino
- **deklaracija spremenljivke** poveže identifikator (ime) s podatkovnim tipom
  - prevajalniku pove, kaj identifikator predstavlja (kako identifikator obravnava)
- Sintaksa deklaracije spremenljivk
  - `tip ime1, ime2, …, imen;`
  - `tip ime = izraz;`

Sintaksa deklaracije spremenljivk

8

- začetno vrednost spremenljivke lahko določimo v deklaraciji
  - torej je v tem primeru deklaracija hkrati definicija
  - izraz na desni strani enačaja lahko vsebuje poljuben izraz
    - v tem izrazu morajo biti vse uporabljene spremenljivke in funkcije že definirane
- deklaracije so stavki programskega jezika C++
- **že deklarirani** spremenljivki lahko spremenimo vrednost na naslednji način:
  - `ime = izraz;`
- pravimo: spremenljivki priredimo vrednost

Primer


```
#include <iostream>
using namespace std;

int main()
{
    int a = 10;
    char c1 = 'A', c2;
    double x = 0.77, y = x + a;

    cout << "a=" << a << endl;
    cout << "x=" << x << "\ty=" << y << endl;
    cout << "c1=" << c1 << endl;
    cout << "c2=" << c2 << endl;

    return 0;
}
```

6. Vhod / izhod

Vhod / izhod

11

- vhodno / izhodne rutine
  - objekti in ukazi, ki omogočajo vnos in izpis podatkov
  - niso direktno del jezika C++
  - množica tipov in metod v standardni knjižnici `iostream`
- knjižnica `iostream`
  - definira objekte za vnos in izpis podatkov
  - operatorji za delo s temi objekti
- knjižnice iz programskega jezika C
  - v programskem jeziku C++ so knjižnice poimenovane enako, le da je pred njimi pripona **`c`** in ni pripone **`.h`**
  - npr. `stdio.h => cstdio`

Knjižnica `iostream`

12

- definira naslednje objekte za delo
  - cout ... izpis, standardni izhod (konzola, ekran, ...)
  - cin ... vnos, standardni vhod (tipkovnica, ...)
- definira naslednje operatorje za delo
  - << ... "pošlji na" izhodni tok
  - >> ... "pridobi iz" vhodnega toka

nad števili imata ta dva operatorja drug pomen
<< ... pomik bitov v levo
>> ... pomik bitov v desno

Uporaba izhodnih tokov in operatorja <<

13

- sintaksa uporabe za izhodni tok
  - `cout << podatek1 << podatek2 << … << podatekn;`
  - za vsakim operatorjem << mora biti natanko en podatek
  - vsak podatek lahko tudi izpišemo s svojim stavkom:
    - `cout << podatek1;`
    - `cout << podatek2;`
    - …
    - `cout << podatekn;`
- podatki, ki jih lahko na ta način izpišemo so lahko:
  - cela števila, realna števila, znaki, nizi znakov
- sintaksa je enaka za vse izhodne toke (cout, datoteke…)

Uporaba vhodnih tokov in operatorja >>

14

- sintaksa uporabe za vhodni tok
  - `cin >> objekt1 >> objekt2 >> … >> objektn;`
  - za vsakim operatorjem >> je natanko en objekt
  - vsakemu objektu lahko preberemo vrednost s svojim stavkom:
    - `cin >> objekt1;`
    - `cin >> objekt2;`
    - …
    - `cin >> objektn;`
- podatki, ki jih lahko na ta način preberemo so lahko:
  - cela števila, realna števila, znaki, nizi znakov
- sintaksa je enaka za vse vhodne toke (cin, datoteke…)

Primer (1/2)

15

```
#include <iostream>
using namespace std;

int main()
{
    // deklarirajmo dve spremenljivki
    int i;      // celo število
    double x;   // realno število

    // branje vrednosti v spremenljivko x
    cout << "Vnesi realno stevilo: ";
    cin >> x;

    // branje vrednosti v spremenljivko i
    cout << "Vnesi pozitivno celo stevilo: ";
    cin >> i;
```

Primer (2/2)

16

```
    // branje ponavljamo tako dolgo, dokler
    // vnesena vrednost za i ni pozitivno stevilo
    while (i<1) {
        cout << "Napaka, i=" << i << endl;
        cout << "Vnesi pozitivno celo stevilo: ";
        cin >> i;
    }

    cout << "i * x = " << i*x << endl;

    return 0;
}
```