Osnove programskega jezika C++

1

OSNOVE PROGRAMIRANJA V C++
UPORABA PROGRAMA CODE::BLOCKS

Nekaj osnovnih pojmov (1.)

2

- programski stavek
  - definicija, deklaracija, ukaz
  - sestavljen iz elementov jezika C++
- funkcija
  - samostojna skupina programskih stavkov
  - je poimenovana

Nekaj osnovnih pojmov (2.)

3

- objekt
  - predstavitev neke informacije
  - je poimenovan
  - vrednosti opisujejo lastnosti objekta
  - je sposoben reagirati na sporočila (s pomočjo funkcij/operatorjev)
- program
  - je zbirka definicij, deklaracij in funkcij
  - je lahko razdeljen na več datotek

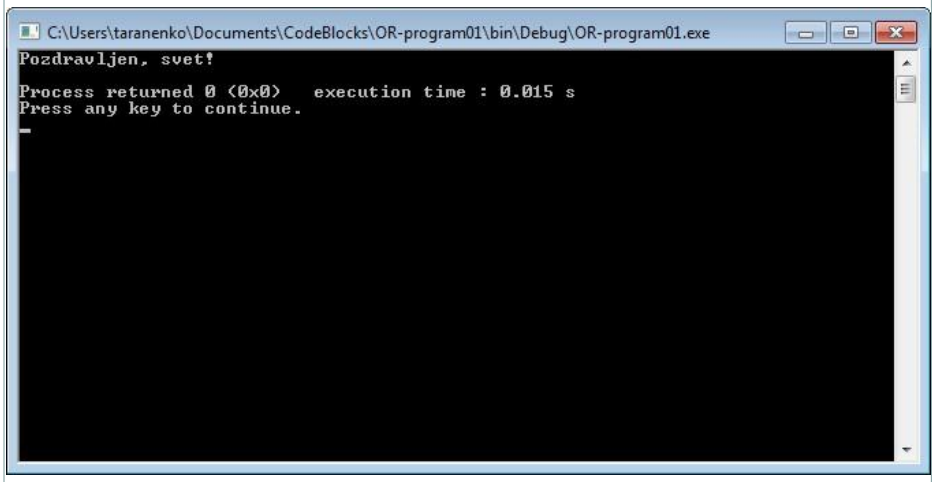
Prvi program v C++

4

```
/*
      Moj prvi program.
      Avtor: Andrej Taranenko.
*/

#include <iostream>
using namespace std;

int main() {
  // Na ekran izpišimo sporočilo
  cout << "Pozdravljen, svet!" << endl;

  return 0;
}
```

Prvi program v C++ ... rezultat

5



Prvi program v C++

6

```
/*
      Moj prvi program.
      Avtor: Andrej Taranenko.
*/
```

→ več vrstični komentar

```
#include <iostream>
```

→ vključujemo zunanje vire

```
using namespace std;
```

→ lažji dostop do nekaterih objektov

```
int main() {
  // Na ekran izpišimo sporočilo
  cout << "Pozdravljen, svet!" << endl;

  return 0;
}
```

→ funkcija z imenom *main()*

→ tukaj se program prične izvajati

→ konča izvajanje funkcije *main()*, s tem zaključi program

Programi v C++

7

- Preprost program v C++ se prične tako:

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

int main() {
```

- in zaključi na naslednji način:

```
  return 0;
}
```

Zapis programa v C++

8

- prevajalnik strogo loči med malimi in velikimi črkami
- sprejme se poljubna oblika zamikov in praznin
- program zapišemo tako, da je enostavno čitljiv
  - { postavimo takoj za stavek, h kateremu pripada
  - } postavimo v svojo vrstico in poravnamo z začetkom stavka, ki ga zaključuje
  - v eno vrstico zapišemo en stavek
  - stavke, ki “spadajo skupaj” zamaknemo za enako št. znakov
- stavki (ne nujno vrstice) se zaključijo s podpičjem

Zapis programa v C++

#include <iostream>

- pove prevajalniku, kje išče informacije, ki jih uporabljamo v programu
- s tem vključujemo knjižnico iostream
- iostream vsebuje definicije objektov cout in cin

using namespace std;

- v katerem imenskem prostoru (predstavljajmo si skupino) naj išče informacije, če jih ne najde prej

Zapis programa v C++

int main() {

- je začetek glavne funkcije
- program se prične izvajati tukaj

return 0;
}

- zaključek funkcije main()
- sporoči operacijskemu sistemu, da pri izvajanju ni bilo napak (vračamo vrednost nič – 0 napak)

Zapis programa v C++

11

```
/*
      Moj prvi program.
      Avtor: Andrej Taranenko.
*/
```

- je komentar, ki zavzema več vrstic
- prične se s simbolom /*
- zaključi se s simbolom */
- vse vmes šteje za komentar
- komentarji niso del izvedljivega programa
- komentarji so namenjeni tistim, ki delajo z izvornim programom

Zapis programa v C++

12

```
// Na ekran izpišimo sporočilo
```

- je enovrstični komentar
- prične se s simbolom //
- zaključi se s skokom v novo vrstico
- vse od simbola // naprej in do konca vrstice šteje za komentar
- običajno z njimi opišemo stavek oz. skupino stavkov, ki sledijo

Zapis programa v C++

13

cout << "Pozdravljen, svet!" << endl;

- z objektom cout prikažemo informacijo na ekranu (konzoli)
- operator << lahko beremo kot “pošlji” v smeri puščic
- tekst v C++ zapišemo znotraj dvojnih narekovajev
- endl predstavlja skok v novo vrstico
- podpičje zaključuje stavek, ki se je pričel s cout

Prevajanje, povezovanje, zagon…

14

- izvorni program oz. izvorno kodo pišemo v navadnem urejevalniku besedil
- shranjen je v .cpp datoteki – navadna tekstovna datoteka
- prevajalnik pretvori izvorno kodo v objektno kodo
- povezovalnik združi vse objektne kode v izvedljivi program

Prevajanje, povezovanje, zagon...

15

- kodo prevedemo in povežemo v izvedljiv program
  - pritisnemo ikono za ustvarjanje izvedljivega programa
  - pritisnemo kombinacijo tipk **Ctrl** in **F9**
  - izberemo meni Build / Build
- odpravimo napake, na katere opozori prevajalnik
  - po odpravljanju napak moramo program znova prevesti iz zgraditi

slovarček:

- build ... zgradi
  - prevede program in zgradi izvedljivi program

Prevajanje, povezovanje, zagon...

16

- izvedljivi program zaženemo
  - pritisnemo ikono za zagon programa
  - pritisnemo kombinacijo tipk **Ctrl** in **F10**
  - izberemo meni Build / Run
- oba koraka lahko združimo z Build and Run
  - pritisnemo ikono za zgradi in zaženi
  - pritisnemo tipko **F9**
  - izberemo meni Build / Build and Run

slovarček:

- run ... zaženi
  - zažene izvedljivi program
- build and run
  - prevede program in tvori izvedljivi program, ki ga nato še zažene

Napake, napake, napake ...

17

- sintaktične napake
  - so napake, ki kršijo pravila sintakse jezika C++
  - prepozna jih prevajalnik
  - prevajalnik jih opiše (pove, kaj ga moti) in pogosto prikaže, kje v izvorni kodi je napaka
- logične napake
  - težko odkriti
  - so napake v algoritmu reševanja
  - računalnik jih ne prepozna kot napake
- napake v času izvajanja programa
  - napake, do katerih pride, ko izvajamo program (npr. napačen tip podatkov)

Izpis na ekran (1)

18

- na ekran izpisujemo s pomočjo objekta **cout**
- definiran je v knjižnici *iostream*
- je v imenskem prostoru *std*

- spomnimo se, na začetku programa imamo:

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

- to je zato, ker smo uporabljali cout

Izpis na ekran (2)

19

- informacijo pošljemo objektu cout z operatorjem <<
- je en simbol zapisan z dvema znakoma <
- med njima ne sme biti presledka

PRIMERI:

- cout << "Moje ime je Andrej.";
- cout << endl;
- cout << "PI = " << 3.141592;

```
Moje ime je Andrej.
PI = 3.14159
Process returned 0 (0x0)   execution time : 0.015 s
Press any key to continue.
```

Izpis na ekran (3)

20

- v splošnem
  - cout << podatek; // izpišemo en podatek
  - cout << podatek1 << podatek2 << podatek3 ;

    ali
  - cout << podatek1;
  - cout << podatek2;
  - cout << podatek3;

ni omejeno na tri podatke!

Izpis na ekran (4)

21

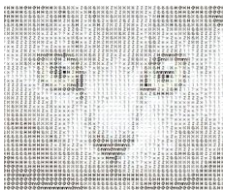
- vrste podatkov, ki jih lahko na ta način izpišemo
  - besedilo (tekst oz. nizi)
    - prepoznamo po dvojnih narekovajih
  - cela števila
    - lahko zapišemo v desetiškem številskem sestavu (kot smo vajeni)
  - realna števila
    - več možnosti zapisa
    - vajeni smo npr. 3,141592
    - NE SMEMO uporabljati decimalne vejice, ampak **piko**
    - npr. 3.141592
  - znaki
    - kadar gre za podatkovni tip 1 znak, ga pišemo v enojnih narekovajih

→ **osnovni podatkovni tipi**

Izpis na ekran (5)

22

- Bi znali na ekran izpisati tole:
  - imenovani tudi ASCII ART
    - torej umetnost z znaki ASCII tabele



- Kaj pa kaj lažjega, kot na primer svoje ime v takšni obliki:

```
 AAA  N   N DDDD  RRRR  EEEEE     J
A   A NN  N D   D R   R E         J
AAAAA N N N D   D RRRR  EEEE      J
A   A N  NN D   D R  R  E     J   J
A   A N   N DDDD  R   R EEEEE  JJJ
```

Osnovni podatkovni tipi

23

- podatkovni tip pove vrsto podatka
- C++ pozna kar nekaj osnovnih podatkovnih tipov
- tri skupine osnovnih podatkovnih tipov
  - cela števila
  - realna števila
  - znaki

3.141592 # -34512

A @ 1

Podatkovni tipi za cela števila

24

- osnovni podatkovni tip za celo število je *int*
  - *32-bitno predznačeno celo število*
  - -2 147 483 648 ... 2 147 483 647
- če želimo samo nenegativna števila
  - *unsigned int*
  - 0 .. 4 294 967 295
- za ta podatkovni tip so definirane
  - osnovne aritmetične operacije (+ - * /)
  - osnovne relacijske operacije (< > <= >= != ==)

Podatkovni tipi za realna števila

25

- za realno število smatramo tisto, ki lahko ima tudi decimalni del
- vsako realno število ima torej
  - celi del (pred decimalno piko)
  - decimalni del (za decimalno piko)
- na voljo imamo podatkovna tipa
  - float
  - double – mi bomo uporabljali tega
  - razlika je natančnost in velikost števil, ki jih lahko z njima predstavimo

Podatkovni tip za znak

26

- podatkovni tip za znak je *char*
- je tesno povezan s celimi števili (spomni se poglavja predstavitev znakov v računalniku)
  - celo število predstavlja znak iz ASCII tabele (in obratno)
  - npr:
    - 'A' predstavlja 65
    - 'B' predstavlja 66
    - 'a' predstavlja 97
    - 'b' predstavlja 98
    - '@' predstavlja 64

Podatkovni tip za znak

27

- poznamo tudi "posebne" znake
  - znaki, ki jih ne moremo izpisati direktno
  - npr. kako bi izpisali narekovaj?
- ti znaki se pričnejo s poševnico nazaj \
  - pomembni primeri
    - \n ... skok v novo vrstico (kaj pa *endl*?)
    - \\ ... poševnica nazaj
    - \" ... dvojni narekovaj
    - \' ... enojni narekovaj
    - \t ... tabulator

Imena v C++

28

- imena predstavljajo vrednosti oz. komponente
- nekatera imena so že rezervirana kot del jezika
  - ključne oz. rezervirane besede
  - ne smemo oz. ne moremo jim spreminjati pomena
  - so sestavljena samo iz malih črk (angleške abecede)
  - prevajalnik jih prepozna in pozna njihov pomen
- identifikatorji – imena, ki jih definiramo mi oz. so definirana dodatno (niso del samega jezika C++)
  - imena spremenljivk, funkcij, objektov

Kako tvorimo ime?

29

- veljavno ime je sestavljeno iz naslednjih znakov
  - črke angleške abecede
    - nobenih šumnikov!
    - črke so lahko male in/ali velike
  - števk
    - prvi znak imena ne sme biti števka!
  - podčrtaj
    - ga lahko uporabimo namesto presledka
    - seveda nobenih presledkov v imenih

Spremenljivke oz. objekti

30

- vsi objekti, ki jih uporabljamo, morajo biti definirani
- z definicijo objekta povemo ime objekta in njegov tip (vrsta podatka, ki ga vsebuje)
- sintaksa definicije objekta:

tip ime*1*, ime*2*, ..., ime*n*;

vrsta podatka
do sedaj:
• int – celo število
• double – realno število
• char – en sam znak

seznam identifikatorjev ločenih z vejicami

na koncu seznama je podpičje

Primeri definicije spremenljivk

31

```
int stevilo;
int vsota;
double povprecje, x;
char odgovor;
```

spremenljivke v tem primeru nimajo določene vrednosti

```
double pi = 3.141592;
int steviloOseb(18);
```

spremenljivki lahko ob definiciji tudi določimo vrednost na dva možna načina

Prireditveni stavek

32

- osnovni stavek za delo s spremenljivkami je **prireditveni stavek**
- ta stavek spremenljivki priredi novo vrednost
  - v spremenljivko shrani novo vrednost
  - spremenljivki določi novo vrednost
- sintaksa prireditvenega stavka:

```
imeSpremenljivke = vrednost ;
```

podpičje na koncu stavka

v katero spremenljivko shranjujemo vrednost

operator prirejanja

vrednost, ki jo želimo shraniti v spremenljivko

Prireditveni stavek

33

- sintaksa prireditvenega stavka:
  imeSpremenljivke = vrednost ;
- spremenljivka mora biti že definirana!
- delovanje prireditvenega stavka:
  - prireditveni stavek deluje od desne proti levi
  - to pomeni:
    - najprej izračuna oz. ovrednoti vrednost na desni strani enačaja
    - nato vrednost shrani v spremenljivko na levi strani enačaja
- vrednost na levi in vrednost na desni morata biti kompatibilnega tipa

Vrednosti

34

- vrednosti so lahko
  - konstante oz. literali
    - 12, 3.141592, 'a', "besedilo"
  - izrazi sestavljeni iz aritmetičnih operacij
    - seštevanje, odštevanje, množenje, deljenje, modulo
  - klici funkcij, ki vračajo ustrezno vrednost
  - kombinacija prej naštetih

Aritmetične operacije

35

- v C++ poznamo naslednje aritmetične operacije
  - seštevanje (+)
  - odštevanje (-)
  - množenje (*)
  - deljenje (/)
  - modulo (%) – ostanek pri deljenju
- operatorji delujejo nad različnimi podatkovnimi tipi
- rezultat operacije je **vedno** enakega tipa kot sta operanda (če sta enakega tipa)
- če sta operanda različnega tipa, je rezultat "močnejši" tip – bolj splošni od dveh

Primeri aritmetičnih operacij

36

- v nadaljevanju naj bodo:

```
int x, y;
double z;
```

- primeri:

```
x = 10/3;      // x dobi vrednost 3
y = 3/2;       // y dobi vrednost 1
z = 10/3;      // z dobi vrednost 3
z = 10.0/3;    // z dobi vrednost 3.33333

z = 10.0/3 + y*2 – x;
```

Ostanek pri celoštevilskem deljenju - modulo

37

- samo nad celimi števili
- izračuna ostanek pri celoštevilskem deljenju
  - primer:
    - 123.0 / 7 = 17,571428
    - 123 / 7 = 17
    - 123 % 7 = 4 ... zakaj?
      ker je 123 – 7 * 17 = 4
    - 3 % 2 = 1
    - 4 % 10 = 4
    - 12 % 6 = 0

Mešani izrazi

38

- v izrazih, kjer nastopajo različne računske operacije velja prioriteta operatorjev
- prioriteta določa, kaj se izračuna najprej
- podobno kot v matematiki:
  - množenje in deljenje imata prednost pred seštevanjem in odštevanjem
- vrstni red računanja za vse operacije:
  - ( ) najprej izračunaj znotraj oklepajev (najbolj notranji najprej)
  - * / % nato izračunaj te operacije imajo enako prioriteto, od leve proti desni
  - + - nazadnje izračunaj operaciji + in – imata enako prioriteto, od leve proti desni

Oklepaji in primeri

39

- kadar želimo spremeniti prioriteto aritmetičnih operatorjev, uporabimo oklepaje
- oklepaje uporabimo tudi za lepšo čitljivost izrazov

- PRIMERI:
  - x = 10 – 4 * (5 – 2); // najprej odšteje, kar je v oklepajih
  - y = 10 – 4 * 5 – 2; // množenje ima prednost

Še en pomemben primer

40

```
int x, y;
double z;
x = 7;
y = 2;
z = x / y;      // z dobi vrednost 3, saj sta x in y celi števili
                // sedaj ne moremo pripisati decimalne pike
                // želeli bi pa natančen rezultat

z = double(x) / y;  // v tem računu x smatraj kot realno število
                    // sam x se s tem ne spremeni!!!
                    // z dobi vrednost 3.5
```

Branje vrednosti preko tipkovnice (1)

41

- pogosto želimo, da uporabnik vnese vrednost spremenljivke
- pravimo, da preberemo vrednost preko tipkovnice
- s tipkovnice dobimo podatke s pomočjo objekta **cin**
- definiran je v knjižnici *iostream*
- je v imenskem prostoru *std*

- spomnimo se, na začetku programa že imamo:

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

- torej že lahko uporabljamo objekt cin

Branje vrednosti preko tipkovnice (2)

42

- informacijo pošljemo iz objekta cin z operatorjem >>
- je en simbol zapisan z dvema znakoma >
- med njima ne sme biti presledka

PRIMERI:

```
cin >> z;          // s tipkovnice pridobi podatek in ga
                   // shrani v spremenljivko z
cin >> y >> x;     // s tipkovnice pridobi dva podatka
                   // prvega shrani v spremenljivko y
                   // drugega pa v spremenljivko x
```

Branje vrednosti preko tipkovnice (3)

43

- v splošnem:
  - cin >> spremenljivka; // preberemo vrednost v spremenljivko
  - cin >> spr1 >> spr2 >> spr3; // branje več podatkov

    ali
  - cin >> spr1;
  - cin >> spr2;
  - cin >> spr3;

Branje vrednosti preko tipkovnice (4)

44

- vrste podatkov, ki jih lahko na ta način preberemo
  - besedilo (tekst oz. nizi)
    - več o tem, ko bomo delali podatkovni tip za besedilo
  - cela števila
    - lahko zapišemo v desetiškem številskem sestavu (kot smo vajeni)
  - realna števila
    - NE SMEMO uporabljati decimalne vejice, ampak **piko**
    - npr. 3.141592
  - znaki

→ **osnovni podatkovni tipi**

Delo s spremenljivkami - primeri

1. Napiši program, ki prebere ceno artikla brez davka. Nato izračuna in izpiše ceno z vključenim davkom. Predpostavite, da je davčna stopnja 20%.
2. Napiši program, ki prebere polmer kroga. Nato izračuna in izpiše obseg ter ploščino kroga.
3. Napiši program, ki prebere dve celi števili. Nato zamenja in izpiše njuni vrednosti.