Objektno orjentisano programiranje

Uvod u predmet

Objektno orjentisano programiranje

- Nastavnik: doc. dr Dejan Nikolić
  dejansnikolic@gmail.com

AKREDITOVANI UNIVERZITET MB

Cilj predmeta

- Upoznavanje sa elementima objektno orijentisanog programiranja
- Sticanje osnovnih teorijskih znanja i praktičnih znanja o programiranju u jeziku C++
- Osposobljavanje za samostalno rešavanje programskih problema kroz praktičan rad na računaru
- Osposoboljavanje za pisanje programa na jeziku C++ koristeći napredne tehnike programiranja

Ishod predmeta

- Sposobnost snalaženja i rada u različitim razvojnim okruženjima i grupni rad pri rešavanju programskih problema i projekata iz različitih oblasti računarstva.
- Posedovanje teorijskih i praktičnih znanja o objektno orijentisanom programiranju i samostalan i grupni rad pri rešavanju programskih problema i projekata iz različitih oblasti računarstva.

Raspored časova

Predavanja

- Termin: subota, 14:00-16:00
- Predavač: doc. dr Dejan Nikolić
- E-mail: dejansnikolic@gmail.com
- Konsultacije:

Vežbe

- Termin: subota, 16:00 –18:00
- Asistent:

Praktična nastava

- Kroz praktičnu nastavu studenti će se bliže upoznati sa karakteristikama i korišćenjem programskog jezika C++.
- Kroz samostalne zadatke dublje se prodire u određene konstrukcije i primene programskog jezika C++ razvijajući ujedno sposobnosti za rešavanje programskih problema i projekata iz različitih oblasti računarstva.

Ocenjivanje

- Aktivnost u toku predavanja: 10 poena
- Praktična nastava: 20 poena
- Kolokvijum #1: 10 poena
- Kolokvijum #2: 10 poena
- Seminarski radovi: 20 poena
- Završni ispit: 30 poena
- Ukupno: 100 poena

AKREDITOVANI UNIVERZITET MB

Aktivnosti u toku predavanja – 10 poena

Uključuje:

- prisustvo na predavanjima – do 5 poena
- aktivno učešće na predavanjima
- prezentacija tema na predavanjima koje nisu uključene u teme iz domena predmeta

AKREDITOVANI UNIVERZITET MB

Praktična nastava – 20 poena

Uključuje:

- prisustvo na vežbama – do 5 poena
- izrada zadaća koje se daju studentima

AKREDITOVANI UNIVERZITET MB

Kolokvijum 1 – 10 poena

- Radi se u 6. nedelji nastave
- Sastoji se od 30 teorijskih pitanja sa opcionim odgovorima iz gradiva od 1. do 5. nedelje
- Svako pitanje nosi po 0,33 poena

AKREDITOVANI UNIVERZITET MB

Kolokvijum 2 – 10 poena

- Radi se u 12. nedelji nastave
- Sastoji se od 30 teorijskih pitanja sa opcionim odgovorima iz gradiva od 7. do 11. nedelje
- Svako pitanje nosi po 0,33 poena

AKREDITOVANI UNIVERZITET MB

Teme seminarskih radova – 2 seminarska rada po 10 poena

Oblasti

- C++ i Linux OS
- C++ i UNIX OS
- OpenGL programiranje
- Poređenje programskih jezika
- Razvoj aplikacija u jeziku C++
- Integracija C++ i Python
- Integracija C++ i C#
- Integracija C++ i Java
- Integracija C++ i objective-C
- Integracija C++ i Matlab
- .....

Završni ispit – 30 poena

- Programiranje na računaru
- Sastoji se od 1 do 3 zadatka
- Svaki zadatak je konkretan problem za čije rešavanje treba napisati program
- Da bi se dobio maksimalan broj bodova potrebno je da program „proradi“ na računaru

AKREDITOVANO
Univerzitet MB
UNIVERZITET MB

Formiranje konačne ocene

broj poena ≤ 50	5
51 ≥ broj poena ≤ 60	6
61 ≥ broj poena ≤ 70	7
71 ≥ broj poena ≤ 80	8
81≥ broj poena ≤ 90	9
broj poena ≥ 91	10

Preduslov za uspeh na predmetu OOP

- Stalno raditi, tokom celog semestra.
- Redovno pratiti nastavu, samostalno praktično vežbati na računaru.
- Postavljati pitanja o svemu što nije jasno i to odmah, ne čekati da se nejasnoće gomilaju
- Ići na konsultacije.
- Ako na početku sve zvuči poznato, ne znači da će tako ostati do kraja.

Plan predmeta

1. Uvod u predmet. Uvod u jezik C++.
2. Tipovi podataka i operatori.
3. Naredbe za kontrolu toka programa.
4. Nizovi, stringovi, pokazivači.
5. Pregled pređenog gradiva i priprema za Kolokvijum #1.
6. **Kolokvijum #1.**
7. Funkcije.
8. Klase i objekti.
9. Nasleđivanje, virtuelne funkcije i polimorfizam.
10. Ulazno-izlazni (U/I) sistem.
11. Pregled pređenog gradiva i pripreme za Kolokvijum #2
12. **Kolokvijum #2.**
13. Izuzetci.
14. OOP projektovanje. Osnovni principi projektovanja. Šabloni.
15. **Popravni kolokvijum.**

Objektno orjentisano programiranje

Uvod u jezik C++

Programski jezici

- Postoji mnogo programskih jezika i mnogo različitih mišljenja o tome šta je „dobar programski jezik“.
  - fokus nekih jezika je brzina izvršavanja
  - fokus nekih jezika na jednostavnosti pisanja programskog kôda.
  - fokus nekih jezika je samo savršeno izvršavanje jednog jedinog zadatka

AKREDITOVANI UNIVERZITET MB

Niži programski jezici

Mašinski jezik

- Mašinski jezik je najniži nivo programskog jezika
  - jedini jezik koji računar poznaje, program napisan u tom jeziku može se bez dodatnog prevođenja izvršiti na računaru
  - program u mašinskom jeziku za jednu vrstu računara nije primenljiv na drugoj vrsti računara
  - programer, koji piše program u mašinskom jeziku, mora dobro poznavati arhitekturu jedinice za obradu računara

AKREDITOVANI UNIVERZITET MB

Niži programski jezici (2)

Asembler

- Programski jezik niskog nivoa koji mašinski jezik specifične procesorske arhitekture predstavlja u ljudima čitljivom obliku.
- Simbolički kôdovi (mnemonici) zamenjuju binarne kôdove naredbe, kao što su npr. MOV (move), STO (store).
  - ako je asemblerski program napisan za jedan računar nije primenljiv na drugom računaru

AKREDITOVANI UNIVERZITET MB

Viši programski jezici

- Jezici čije se naredbe ne mogu direktno prevesti u binarne naredbe mašinskog jezika (npr. C, C++, Java).
- Više su okrenuti korisniku i problemu, a manje računaru
  - programer najčešće ne mora znati ništa o arhitekturi računara, čime je pisanje programa omogućeno
  - isti programi mogu se izvršavati na različitim vrstama računara

AKREDITOVANI UNIVERZITET MB

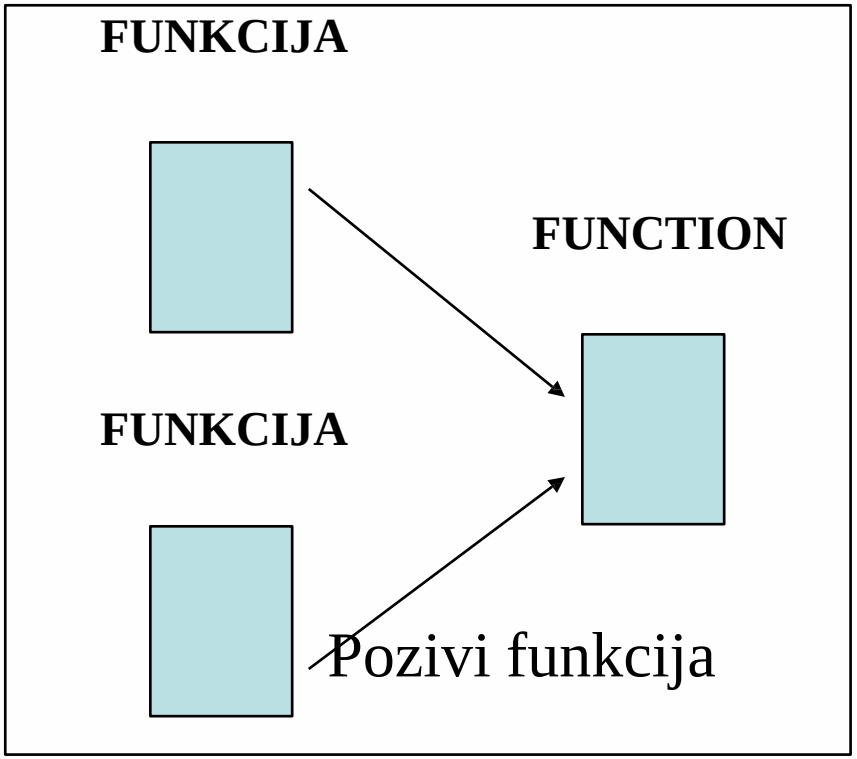
Tipovi programiranja

- Struktuirano programiranje (engl. structured programming)
- Objektno-orijentisano programiranje (Object-Oriented Programming - OOP)

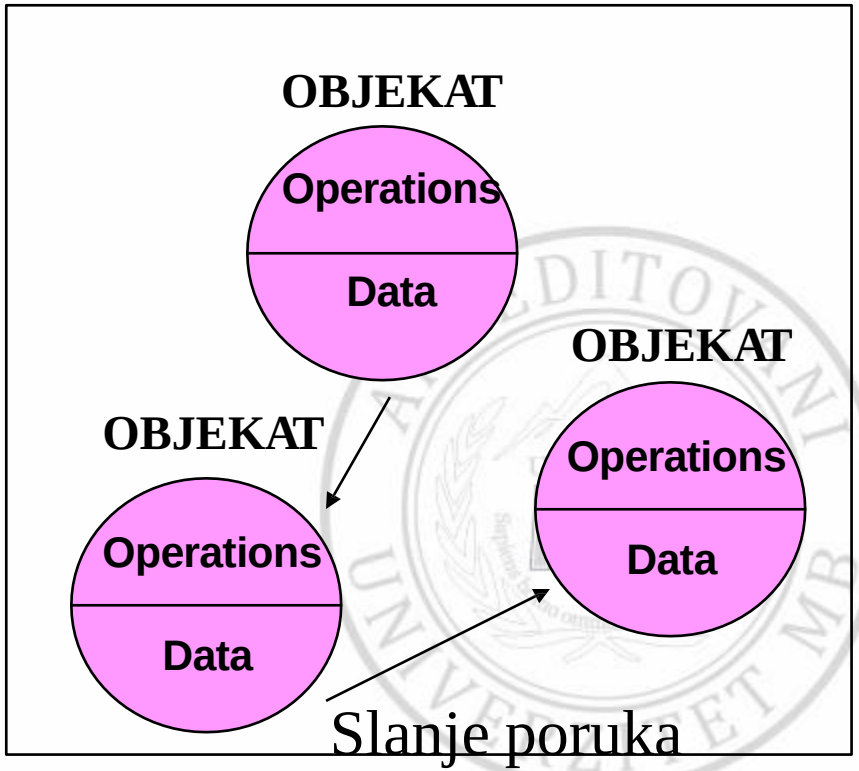
AKREDITOVANI UNIVERZITET MB

Dva modela programiranja

Strukturni (Proceduralni) PROGRAM



Objektno-orijentisani PROGRAM



Struktuirano programiranje

- U početnoj fazi računarstva, izvršavanje operacija se odvijalo pomoću jednostavnih konstrukcija:
  - ako je uslov ispunjen, ići na zadatu lokaciju (koristiti GOTO komandu)
- Dovoljno je koristiti desetak takvih komandi da bi praćenje programa postalo zamorno.
- Umesto nepotrebnih GOTO naredbi, potrebno je izvršiti struktuiranje i programa i podataka koji se obrađuju.

Struktuirano programiranje (2)

Cilj

- Poboljšanje jasnoće, kvaliteta i vremena razvoja računarskih programa
  - intenzivno korišćenje potprograma, blok struktura, for i while petlje i slično.
  - izbegavanje korišćenja jednostavnih testova i programskih skokova, npr. izjava goto koja može da dovede do špageti koda (engl. spaghetti code) kojeg je teško i pratiti i održavati.

Struktuirano programiranje (3)

- Glavno pravilo struktuiranog programiranja je „podeli i vladaj“.
  - računarski program se može zamisliti kao skup zadataka (engl. task).
  - svaki složeni zadatak se može podeliti na skup manjih zadataka sve do trenutka kada delovi postanu dovoljno mali i sami sebi dovoljni da bi se lako mogli razumeti.

Struktuirano programiranje (4)

- Program je podeljen u manje celine koje se naknadno ugrađuju u glavni program.
  - ove celine je lakše napisati i proveriti nego kada su nedeljivi deo glavnog programa
  - skup programskih metoda koje vode ka logičkoj organizaciji i čitljivosti programa
- Logička organizacija programa olakšava pisanje, održavanje i ispravke programa.

Struktuirano programiranje (5)

- Strukturno orijentisani jezici (npr. Pascal, dBASE) prisiljavaju programera na strukturno programiranje, za razliku od nestrukturiranih (npr. BASIC, FORTRAN, COBOL) koji prepuštaju kreiranje programa u potpunosti programeru.

AKREDITOVANI UNIVERZITET MB

Špageti kôd (engl. spaghetti code)

- Špageti kôd je pogrdna reč za izvorni kôd koji ima složenu i zamršenu strukturu kontrole, naročito koristeći mnoge GOTO izjave, izuzetke ili druge " nestruktuirane" konstrukcije grananja.
- Ime dolazi od činjenice da program izgleda kao posuda sa špagetima.

Strukturno programiranje: Primer

Izračunavanje zarada zaposlenih u nekoj kompaniji

- Računanje zarade može biti jako složen zadatak koji se može razložiti na manje podzadatke:
  1. Koliko ima zaposlenih u kompaniji?
  2. Izračunati šta svaki zaposleni treba da dobije.
  3. Izračunati ukupan iznos za sve zarade.
  4. Podeliti ukupan iznos za sve zarade sa ukupnim brojem zaposlenih.

AKREDITOVANI UNIVERZITET

Biranje okruženja

- Računanje ukupnog iznosa zarada se može podeliti:
  1. Dobiti zapis (engl. record) za svakog zaposlenog.
  2. Pristupiti pojedinačnim zaradama.
  3. Dodati pojedinačnu zaradu trenutnoj ukupnoj sumi.
  4. Dobiti zapis za sledećeg zaposlenog

AKREDITOVANI UNIVERZITET MB

Objektno orijentisano programiranje-OOP

- OOP je model programiranja zasnovan na konceptu:
  - **objekata**, koji su strukture podataka koje sadrže podatke, u obliku polja (engl. field) ili atributa
  - **kôda**,u obliku procedura ili metoda.

OOP (2)

- Program se kreira upotrebom skupa objekata (zasebnih programa koji mogu da izvršavaju određene zadatke) koji razmenjuju poruke.
- Glavni program nema direktan pristup podacima objekta i ne može direktno obavljati operacije nad njima kao kod tradicionalnih jezika (npr. BASIC, Pascal).
- Programski jezici - C++, Java, C# (C Sharp), Visual Basic

OOP karakteristike

- **Apstraktni tipovi** podataka koje korisnik može proizvoljno definisati
- **Enkapsulacija -** sakrivanje detalja realizacije nekog tipa
- **Nasleđivanje** - kreiranje novih tipova koji će naslediti sve osobine starih tipova, a ponešto novo i dodati
- **Polimorfizam** - pojavljivanje u više oblika

AKREDITOVANI UNIVERZITET MB

OOP prednosti

- U programiranje se uvodi nov način razmišljanja
  - fokus na projektovanje, manje na samu implementaciju (kodiranje)
  - u OOP se prvo razmišlja o problemu koji se rešava, a onda o programskom rešenju
- Više se razmišlja o delovima sistema (objektima), a ne o algoritmu
  - fokus se prenosi sa realizacije delova programa na međusobne veze između delova programa.

Karakteristike C++ jezika

- **C++** je programski jezik opšte namene
- **C++** je jezik koji kompajlira (prevodi) izvorni kôd (engl. compiled language).
- **C++** je statički tipiziran jezik (engl. statically-typed language).
  - ako se želi manipulisati podacima u C++ jeziku, mora se unapred definisati tip podataka (npr. integer ili real)
  - to se ne može nigde promeniti u programu
- **C++** je jezik visokog nivoa

C++: Programski jezik visokog nivoa

Source Code

```
#include <iostream>
using namespace std;
int main() {
      cout << "Hello World!";
      cout << endl;
}
```

Compiler

Assembly

```
section     .text
global      _start

_start:
    mov     edx,len
    mov     ecx,msg
    mov     ebx,1
    mov     eax,4
    int     0x80
    mov     eax,1
    int     0x80

section     .data
msg     db  'Hello World!',0xa
len     equ $ - msg
```

OO osobine jezika C++

- Enkapsuliranje ili skrivanje podataka (engl. encapsulation)
- Nasleđivanje (engl. inheritance)
- Polimorfizam (engl. polimorphism)

AKREDITOVANI UNIVERZITET MB

Enkapsuliranje

- Osobina da nešto bude samostalna jedinica (engl. self-contained unit) se zove enkapsulacija.
- Sa enkapsulacijom se postiže skrivanje podataka.
- C++ postiže enkapsulaciju kreiranjem klasa, tipovima koje kreira korisnik.

AKREDITOVANI UNIVERZITET MB

Nasleđivanje i ponovno korišćenje (engl. reuse)

- C++ podržava ideju o ponovnom korišćenju pomoću karakteristike nasleđivanja.
- Može se definisati novi tip, koji je proširenje starog tipa.

AKREDITOVANI UNIVERZITET MB

Istorijat jezika C

- Godine 1972. Ken Thompson i Dennis Ritchie su kreirali jezik C.
- C postao popularan jezik, jer je bio jednostavan:
  - izvršavao se prilično brzo
  - izvršavao se na svakom računaru koji je imao C prevodilac
- Kritika na jezik C, jer je bio jednostavan.
  - ne podržava objekte ni klase
  - ponekad je čak i jednostavne zadatke izuzetno teško programirati

AKREDITOVANI UNIVERZITET MB

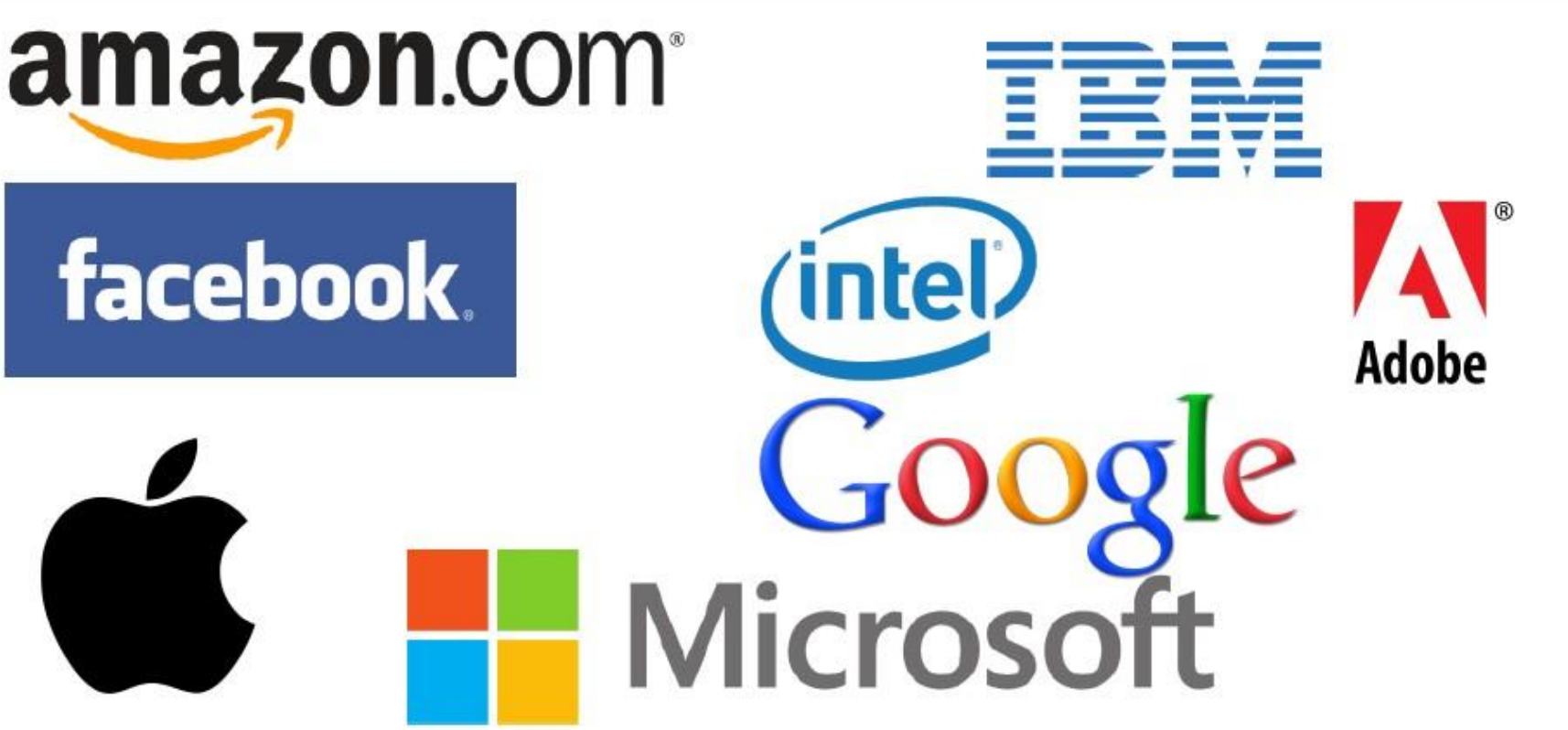
Istorija jezika C++

- 1983. Bjarne Stroustrup je kreirao jezik C++ .
- C++ je programski jezik kojim se pojednostavljuju složeni zadaci bez žrtvovanja performansi.

Zašto C++? - Performanse

- Dobro napisani C++ program nudi neke od najboljih performansi bilo kojeg jezika koji se trenutno koristi.
- Facebook je počeo korisititi C++ i C++ bazirane alate za kontrolu saobraćaja na mreži.
- C++ je jedan od najviše korišćenih jezika u svetu.

Zašto C++? - Korisnici (kompanije)



Zašto C++? - Korisnici (web pretraživači)

Zašto C++? - Korisnici

Lovac F-35 Lightning II (Joint Strike Fighter) se jako oslanja na programski jezik C++.

NASA. Autonomni sistem pokretanja za vozilo koje ispituje površinu Marsa je pisano u jeziku C++. C++ je na Marsu.

The ANSI Standard

- Komitet Accredited Standards Committee, koji radi po procedurama American National Standards Institute-ANSI, je kreirao međunarodni standard za C++ jezik.
  - poznat pod nazivom ISO (International Organization for Standardization) Standard ili NCITS (National Committee for Information Technology Standards) Standard i X3 (staro ime za NCITS) Standard i kao ANSI/ISO standard

Standardizacija jezika C++

- ISO/ANSI Standard: započet 1989, završen tek 1998. (http://www.iso.org)
- Podržava ga većina OS i kompajlera (prevodilaca)
- Relativno prenosiv (pod uslovom da aplikacija nema grafički interfejs)
- Microsoft C++/CLI: 2003
- Izvršava se u .NET okruženju

AKREDITOVANI UNIVERZITET MB

Pitanje

- C je podskup C++ jezika, treba li onda prvo učiti C?
- Stroustrup i većina drugih C++ programera se slažu da ne samo da nije neophodno prvo učiti jezik C, nego da to ne predstavlja nikakvu prednost.
- C programiranje je primer koncepta strukturnog programiranja dok je C++ baziran na OOP.

C++, Java i C# jezici

- Javu su u kompaniji Sun Microsystems razvili bivši C++ programeri
- C# je programski jezik koji je razvila kompanija Microsoft za .NET platformu.
- **C#** jezik koristi uglavnom sintaksu **C++ jezika**.
  - mada su jezici različiti na više načina, učenje C++ jezika obezbeđuje većinu onoga što treba znati o C# jeziku.

C++, Java i C# jezici (2)

- Sintaksa i objektni model sva tri jezika gotovo su identični.
- Najvažnija razlika između ovih jezika je tip okruženja u kome se izvršavaju.
- Ovi jezici su projektovani za različite namene, pa treba izabrati onaj najpogodniji za određenu primenu:

  - C++ - za pisanje efikasnih programa

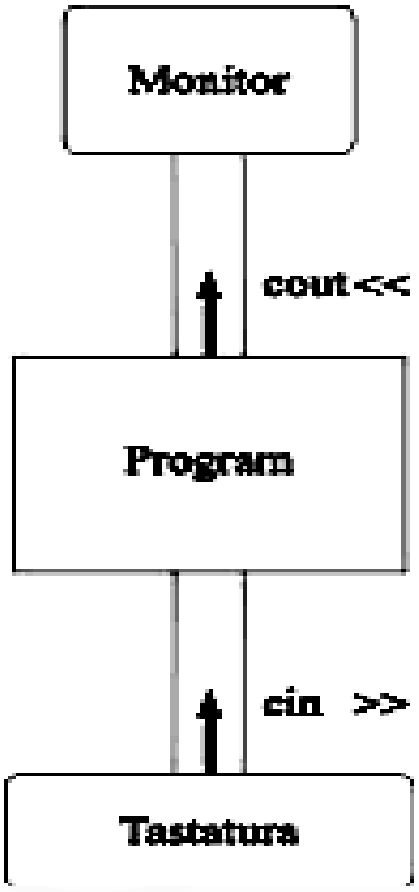
  - Java - za Internet programiranje

  - C# - za specifične Windows programe kod kojih efikasnost nije najvažnija

Izlaz na monitor

- Izlaz na monitor je **tok podataka** (engl. data stream), odnosno niz karaktera.
- Konzolni izlaz ili tok **cout** predstavlja standardni izlazni tok.
- Podaci se mogu poslati na izlazni tok koristeći operator umetanja “**<<“** (engl. insertion operator).

cout << “Hello world”;



Ekstenzije izvornog C++ koda

C++ implementacija	Ekstenzije izvornog C++ koda
Unix	C, cc, cxx, c
GNU C++	C, cc, cxx, cpp, c++
Digital Mars	cpp, cxx
Borland C++	cpp
Watcom	cpp
Microsoft Visual C++	cpp, cxx, cc
Freestyle CodeWarrior	cpp, cp, cc, cxx, c++

Kreiranje objektne datoteke pomoću prevodioca

- C++ izvorni kôd (engl. source code) ima ekstenziju **.cpp**.
- C++ objektni kôd ima ekstenziju .obj ili .o.
- Kôd .o još uvek nije izvršni, da se to desi treba pokrenuti linker koji prevodi .o datoteku u izvršnu datoteku.

AKREDITOVANI UNIVERZITET MB

Prvi C++ program

```
// Prvi program prvi.cpp
#include <iostream > //preprocesorska naredba
int main()    //glavna funkcija
{
    std::cout << „Ovo je prvi C++ program!\n";
    return 0;
}
```

AKREDITOVANI UNIVERZITET MB

Ulazni tok (engl. input stream)

Zaglavlje (engl. header) daje klase tokova (engl. stream) standardnog ulaza i standardnog izlaza

header
<istream>

#include <iostream>

Zaglavlje omogućuje standardni ulaz i kombinaciju ulaz/izlaz stream klasa.

Class templates

basic_istream	Input stream (class template)
basic_iostream	Input/output stream (class template)

Classes

istream	Input stream (class)
iostream	Input/output stream (class)
wistream	Input stream (wide) (class)
wiostream	Input/output stream (wide) (class)

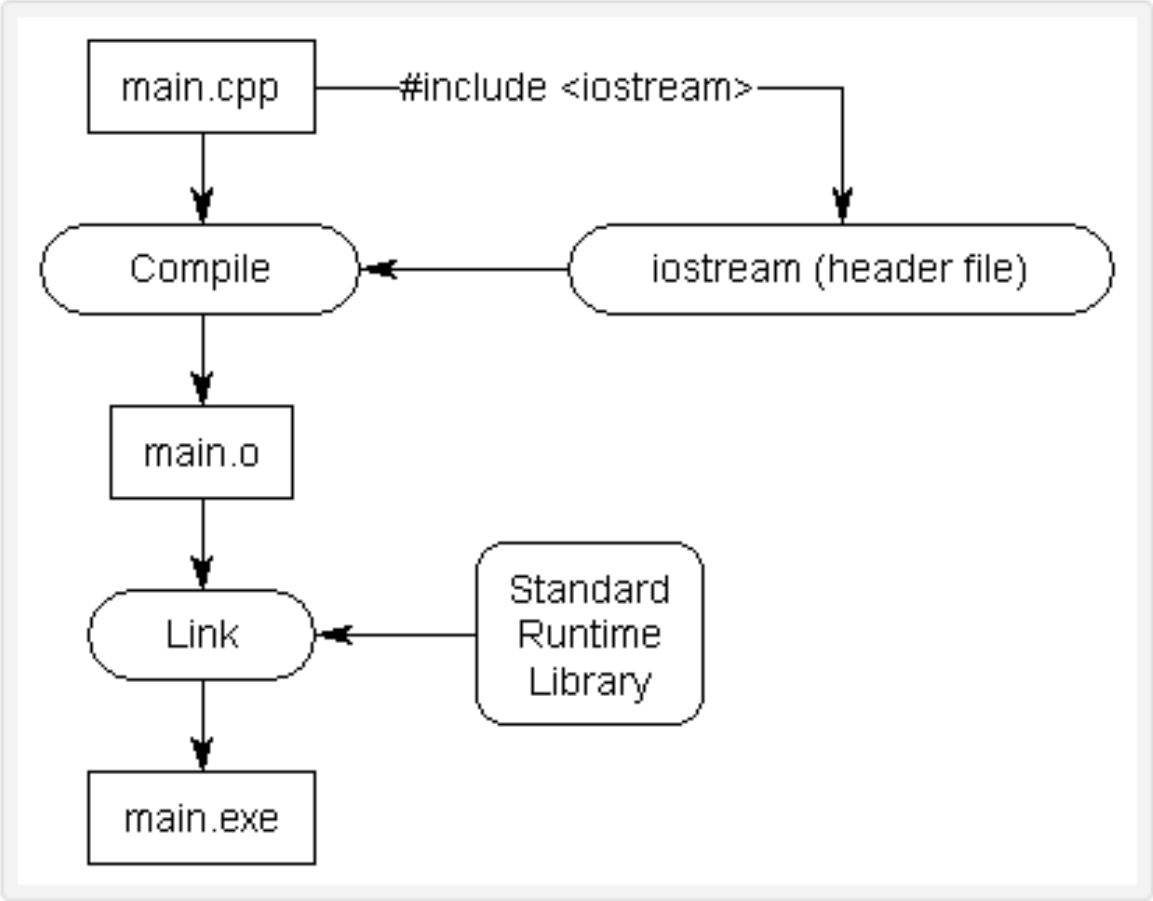
Prepocesorske naredbe

- Jezik C++ je preuzeo od jezika C metod da se:
  - izvorni tekstualni program, koji je napisao programer, najpre procesira posebnim programom za obradu programskog teksta
  - zatim se stvarno prevede.
- Ovaj program se zove **preprocesor** (engl. preprocessor) ili **makroprocesor**.

AKREDITOVANI UNIVERZITET MB

Univerzitet MB
Tvoj

Koristeći datoteke zaglavlja (engl. header files)



#include <iostream > //preprocesorska naredba

mbuniverzitet.edu.rs

Preprocesorske naredbe (2)

- Naredbe koje se izvršavaju pre početka prevođenja izvornog kôda.
  - zavisno od preprocesorskih naredbi, preprocesor menja i dopunjuje izvorni kôd.
- Opšti oblik preprocesorske naredbe

#naredba parametri

  - preprocesorske naredbe započinju znakom #. Za razliku od ostalih naredbi, ne završavaju znakom tačka-zarez (;).
  - postoji više preprocesorskih naredbi, a jedna od najčešćih je naredba *include.*

C++ preprocesor

- C++ preprocesor modifikuje izvorni kôd (engl. source code) pre predaje tog koda prevodiocu.
- Najčešće, preprocesor uključuje druge datoteke direktno u program ili definiše konstante.
- Takođe će C++ preprocesor „ugušiti“ ulaz koji ne poštuje C++ leksička pravila.

AKREDITOVANI UNIVERZITET MB

Direktive

- Počinju znakom # i završavaju na kraju reda.
- Ako je potreban nastavak direktive, kao veza se koristi znak (\).
- Direktiva važi od mesta navođenja do kraja datoteke.
- Direktive nemaju nikakve veze sa ostakom C++ koda.

```
#include <iostream>
```

AKREDITOVANI UNIVERZITET MB

Direktive (2)

- Linije kôda koje počinju sa znakom „#“ su prepocesorske komande (engl. preprocessor command), koje obično menjaju kôd koji se treba prevesti.
- #include kaže procesoru da ubaci u kôd sadržaj druge datoteke
  - ovde je to datoteka iostream, koja sadrži procedure za I/O operacije

```
#include <iostream>
```

Primer

#include <iostream.h>

- Na mesto ove direktive u datoteci preprocesor unosi kompletnu datoteku iostream.h
- Uključiti datoteku iostream.h u program

#include “iostream.h“

- Ovde se datoteka traži na nekom posebno definisanom mestu (obično tekući direktorij) i ako se tu ne nađe, pretraživanje se nastavlja na istom mestu kao i u prvom slučaju.

Preprocesor makro definicije

- Svaka pojava identifikatora TABLE_SIZE (makroa) u tekstu će se zameniti sa tekstom (telom) makroa.

```
#define TABLE_SIZE 100
int table1[TABLE_SIZE];
int table2[TABLE_SIZE];

#define sum (x,y) ((x) + (y))
a=sum (i+j, 2);  // sum ((i + j) +(2))
```

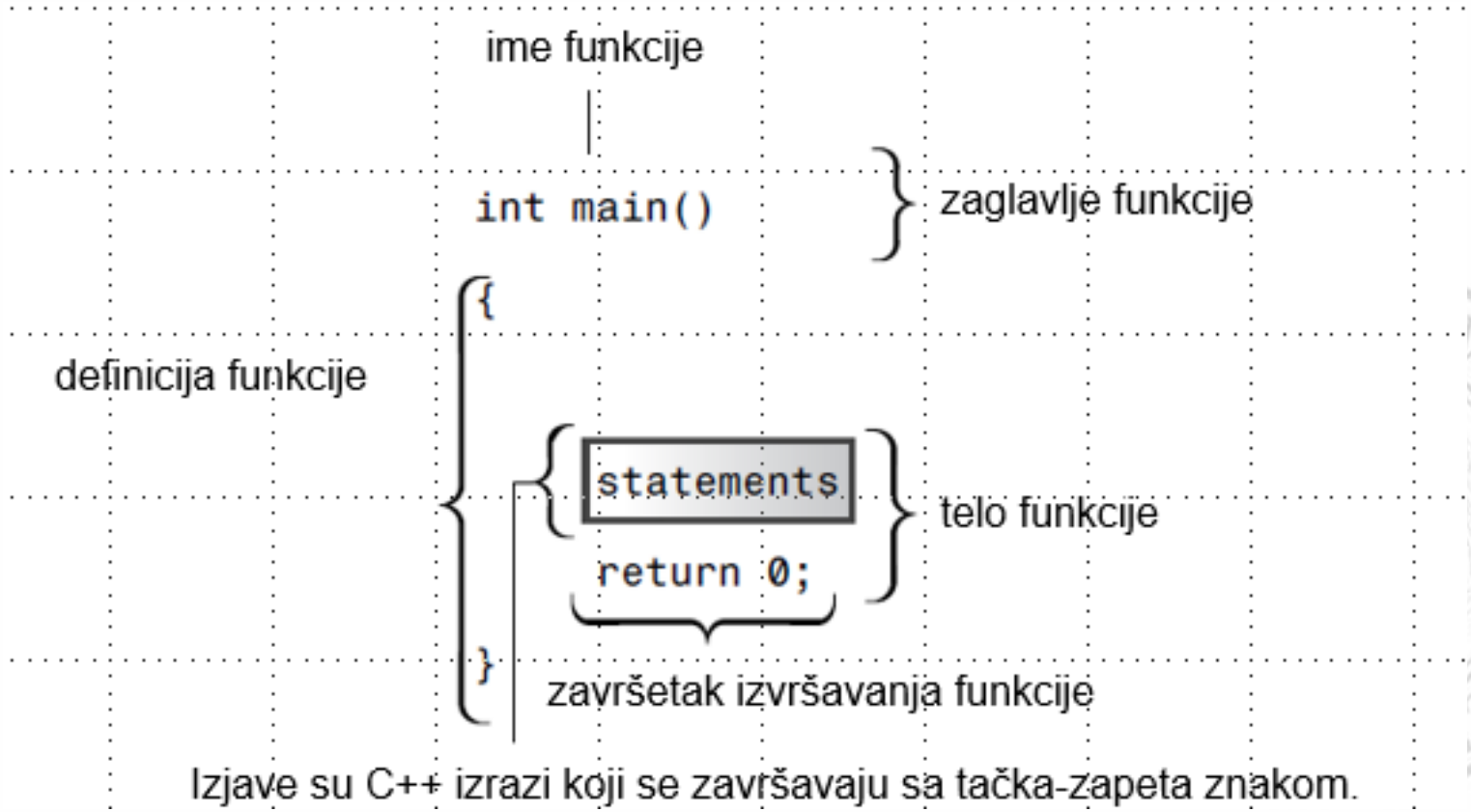
Funkcija main ()

int main() {...}

- Definiše kôd koji treba da se izvrši kada se startuje program.
- Vitičaste zagrade {...} grupišu skup komandi u jedan blok.

AKREDITOVANI UNIVERZITET MB

Funkcija main () (2)



Izjave su C++ izrazi koji se završavaju sa tačka-zapeta znakom.

Prototip za funciju main ()

```
#include <iostream>
int main(); // za većinu prevodilaca ova linija
            //kôda nije potrebna
int main()
{
   std::cout <<”Hello World!\n”;
   return 0;
}
```

Prototipovi funkcija

```
#include <iostream>

using namespace std;

//prototipovi preklopljenih funkcija
int min(int a, int b);
char min(char a, char b);

int main() {
    cout << min(2, 3) << endl;
    cout << min('a', 'B') << endl;
    return 0;
}
```

```
int min(int a, int b) {
    int min;
    (a>b) ? min=a : min=b;
    return min;
}

//min za char ignorise velika slova
char min(char a, char b) {
    char min;
    (tolower(a)>tolower(b)) ? min=a : min=b;
    return min;
}
```

Kontrolne sekvence (engl. escape sequences)

std::**cout** << „Ovo je prvi C++ program! **\n**";

- Primer: „\n“ označava karakter nove linije (engl. newline character.

Escape Sequence	Represented Character
\a	System bell (beep sound)
\b	Backspace
\f	Formfeed (page break)
\n	Newline (line break)
\r	“Carriage return” (returns cursor to start of line)
\t	Tab
\\	Backslash
\’	Single quote character
\"	Double quote character
some integer x	The character represented by x

mbuniverzitet.edu.rs

return 0

- Ova komanda kaže OS da se program uspešno završio – to je zadnja linija u bloku main () funkcije.

Univerzitet MB
Tvoj Univerzitet

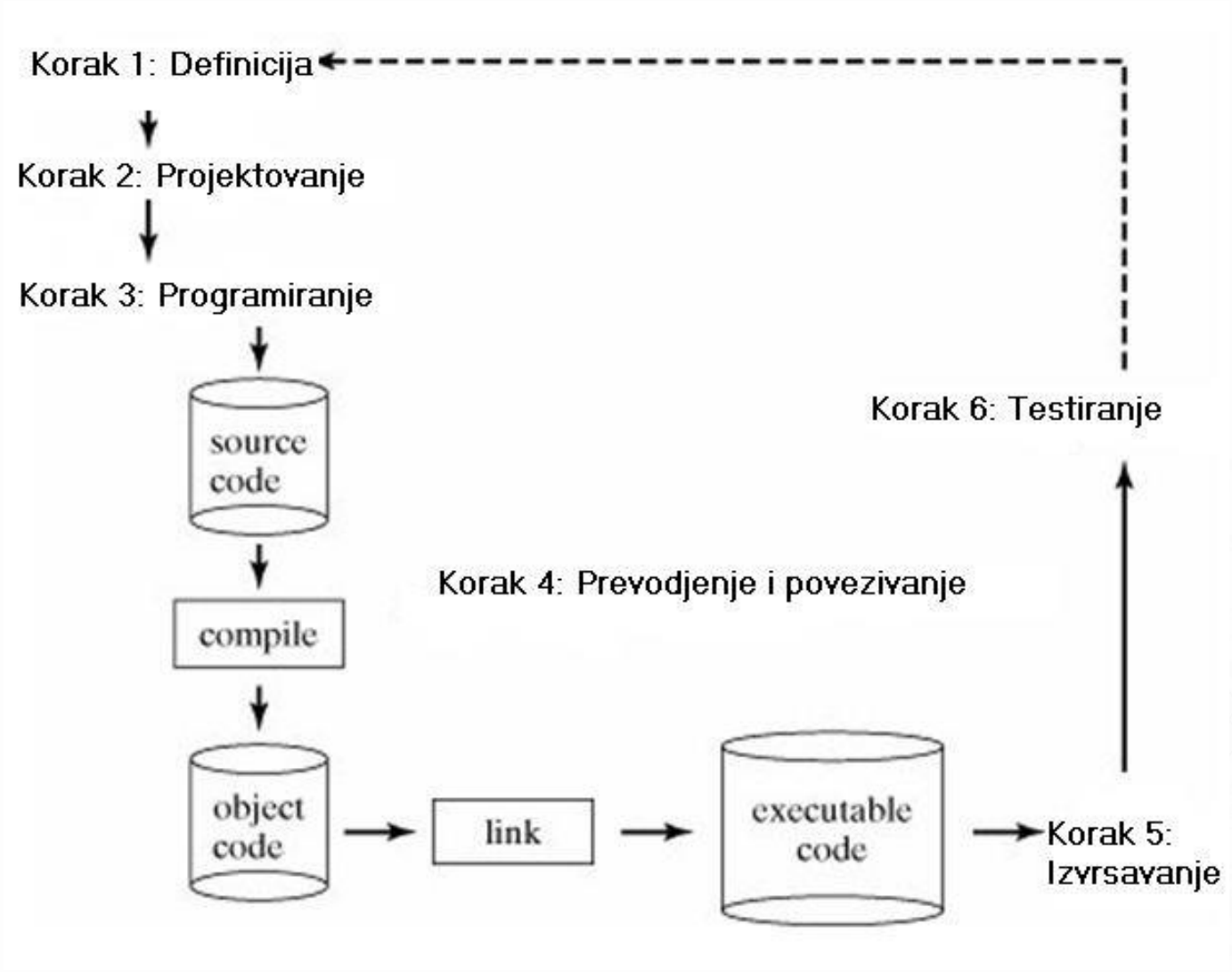
Faze C++ programiranja u C++ programskom okruženju

- **Uređivanje** (engl. edit) - u tekst editoru, snimanje datoteka sa .cpp ekstenzijom
- **Preprocesiranje** - zamena teksta, ubacivanje drugih datoteka iz biblioteka u program i da se izvrši prevođenje
- **Prevođenje** (engl. compile) - u objektni kôd (engl. object code)
- **Povezivanje** (engl. link) - povezivanje objektnog kôda sa kôdom funkcija iz standardnih biblioteka koje se pozivaju u programu
- **Build** procesa - generiše se izvršni format (engl. executable image)
- **Load** – program se postavlja u memoriju
- **Execute** – izvršava se jedna po jedna instrukcija

mbuniverzitet.edu.rs

Univerzitet MB
Tvoj Univerzitet

Ciklus razvoja aplikacije



mbuniverzitet.edu.rs

Zaključak

- Niži programski jezici – mašinski jezik, asembler
- Viši programski jezici – C, C++, Java, ...
- Tipovi programiranja – struktuirano i objektno orjentisano programiranje
- OOP karakteristike – enkapsulacija, nasleđivanje, polimorfizam
- C++ je standardizovan jezik visokog nivoa
- Preprocesorske naredbe
- Faze C++ programiranja – uređivanje, preprocesiranje, prevođenje, povezivanje, izvršavanje

Kraj prezentacije

HVALA NA PAŽNJI!