II – Karakteristike C jezika

- Programski jezik C je viši programski jezik opšte namene.
- Tesno je povezan sa UNIX OS uz koji je razvijan.
- Razvio ga je Dennis Ritchie 1970 u Bell Telephone Laboratories, Inc.
- Opis jezika dat je u knjizi Brian W. Kernighan, Dennis M. Ritchie: ***The C Programming Language, Prentice-Hall, 1978.***
- Tokom 70 i 80 godina jezik se brzo širio pa je ***American National Standard Institute*** (ANSI) izvršio njegovu standardizaciju 1989 god.
- Definisan kao strukturni jezik opšte namene i otklonjene su mnoge neodređenosti i izmenjeni neki koncepti u njemu
- Ima neke specifičnosti koje ga približavaju asemblerskim jezicima
- C je portabilan a to znači da se ne vezuje za OS ili hardware.
- Programi napisani u C-u su efikasni, uglavnom manji i brži od programa pisanih u drugim programskim jezicima.
- Smatra se za jedan od najvažnijih programskih jezika u istoriji komercijalne računarske industrije.

C jezik se preporučuje svima onima koji se nikada ranije nisu susreli sa bilo kakvim programiranjem.

II – Karakteristike C jezika

- Jezik vrlo niskog nivoa jer su programi u C bliski načinu rada hardvera
- C jezik koristi kratke sintaksne konstrukcije – **teško čitljiv program**
- Koristi se kada je ključna brzina izvođenja i/ili prenosivost programa
- Fleksibilan jezik opšte namene, što znači da se u njemu mogu programirati gotovo **sve vrste aplikacija:**
  - ✓ Razni namenski programi,
  - ✓ CAD, igre, komercijalni sistemi,
  - ✓ Veštačka inteligencija (ekspertni sistemi, robotika)
  - ✓ Upravljanje procesima, real-time sistemi
  - ✓ Sistemski softver (operat.sistemi, kompajleri, linkeri, drajveri itd.)
  - ✓ Aplikacije na mobilnim telefonima, ...
- Dozvoljava operacije niskog nivoa:
  - ✓ nedostatak (slabi pouzdanost sistema)
  - ✓ prednost (sprega niskog nivoa sa resursima sistema)
- Pod uticajem C-a razvijeni su brojni drugi programski jezici
- Mnogi od njih su nasledili njegovu sintaksu: C++, Java, JavaScript, C#, PHP, Objective-C, ...

II – Prednosti upotrebe C jezika

- ✓ Kompaktnost asemblerskog jezika
- ✓ Mali skup komandi tj. mali broj ključnih reći
- ✓ Efikasnost
- ✓ Nije jezik sa jakom tipizacijom
- ✓ Strukturni jezik visokog nivoa
- ✓ Modularan dizajn
- ✓ Integracija sa asemblerskim jezikom
- ✓ Rad sa bitovima - ima bogat i složen operatorski jezik
- ✓ Pokazivačke promenljive
- ✓ Razvijen sistem funkcija
- ✓ Prilagođene strukture i mogućnost prenosivosti
- ✓ Podrška nezavisnih proizvođača

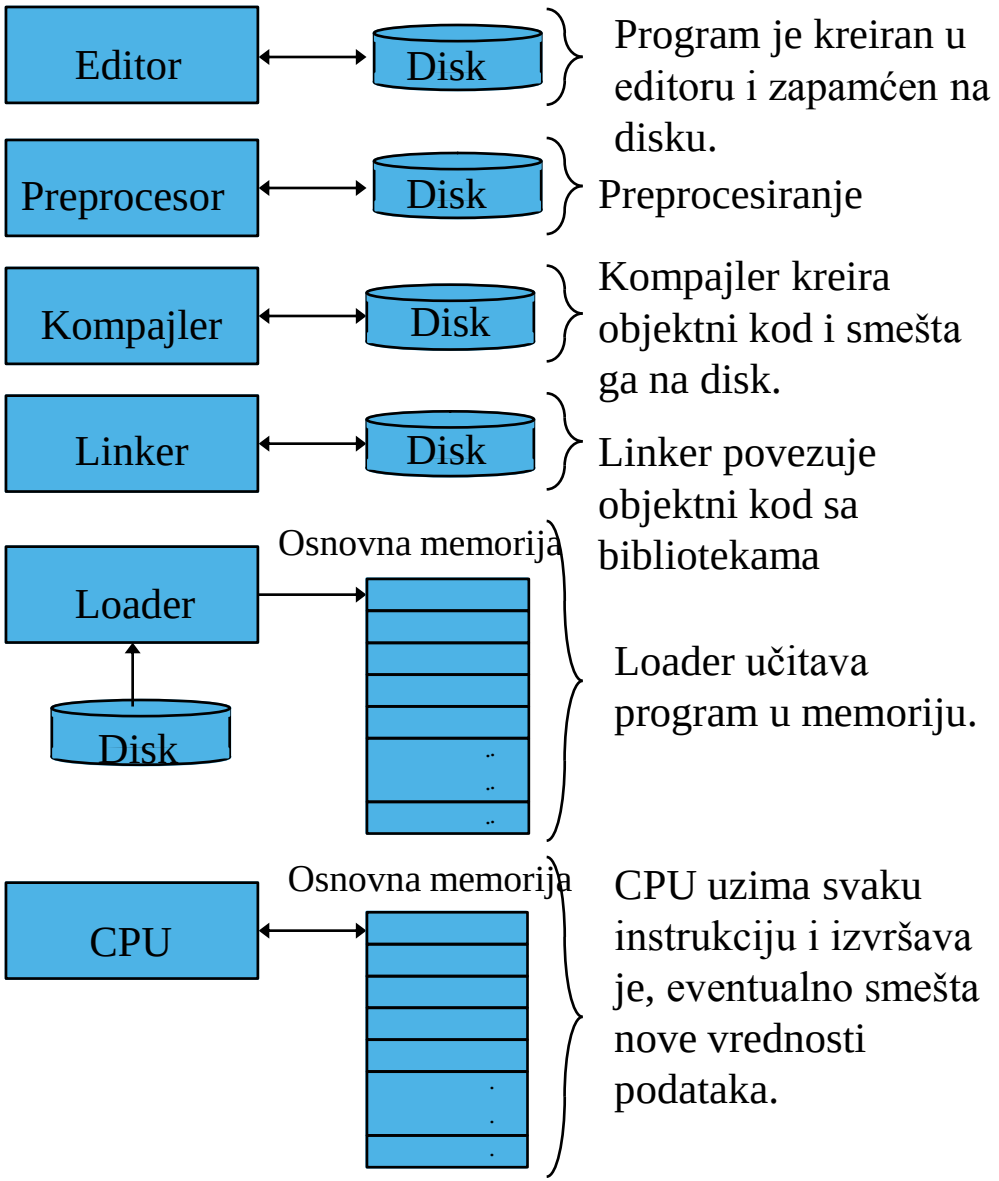
II – Nedostaci upotrebe C jezika

- Ne postoje operacije za direktnu manipulaciju sa složenim objektima (stringovi, liste, polja) - ne postoji operacija za manipulaciju sa celim poljem ili stringom
- Ne postoji direktna podrška za ulaz i izlaz – ne postoje READ i WRITE naredbe
- Nije ugrađen pristup fajlovima tj. ne postoje metode pristupa
- C nije strogo tipiziran jezik
- Ne postoji automatska konverzija neusaglašenih tipova
- C je relativno mali jezik, opisan kratko i uči se brzo.
- Ima reputaciju da je težak za učenje.
- Većina koncepata u C-u se nalaze i u Pascalu: funkcije, povezane liste, polja, parametri

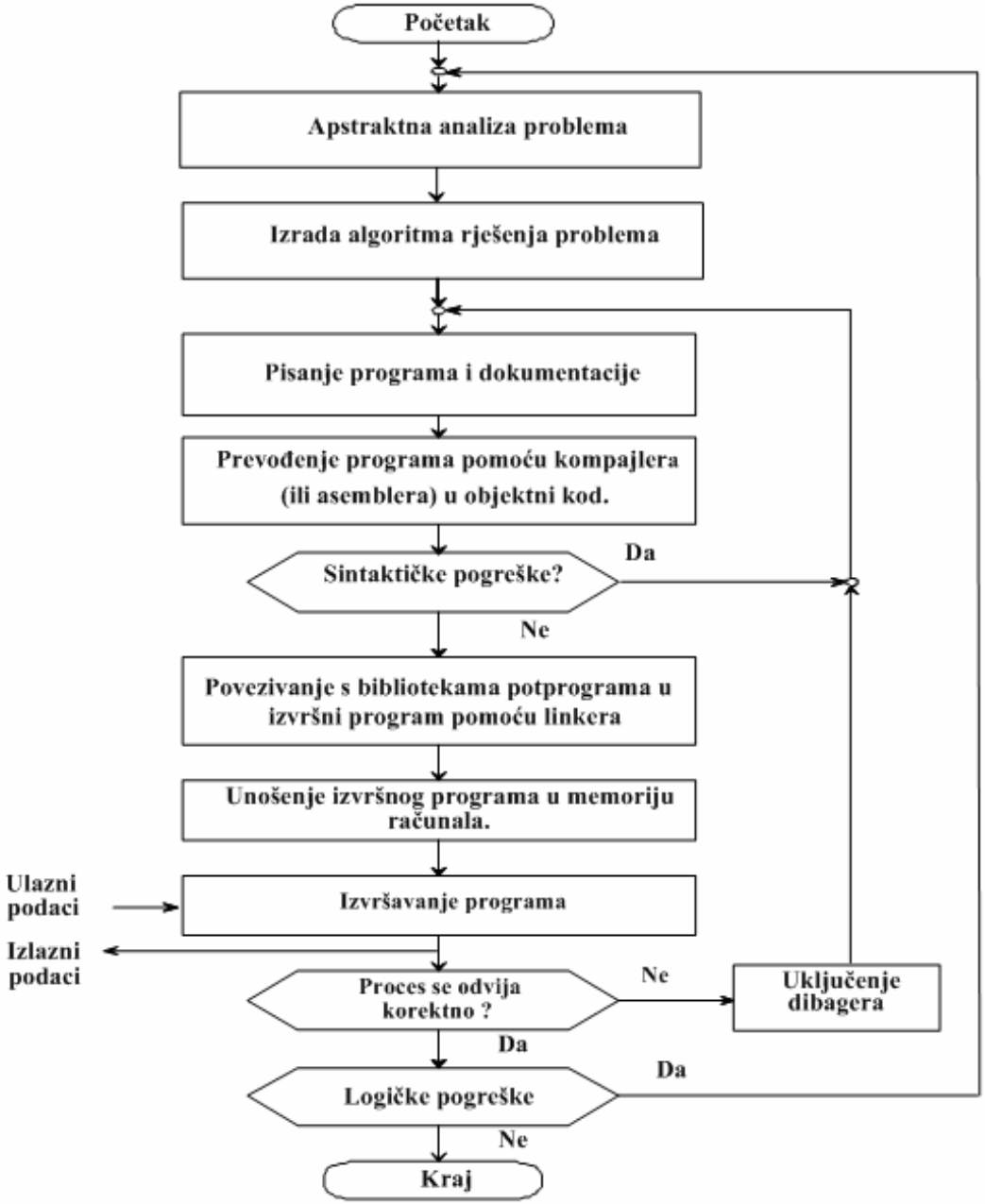
II – Razvoj C programa

Faze:

1. *Editovanje*
2. *Preprocesiranje*
3. *Kompilacija*
4. *Linkovanje*
5. *Učitavanje(Load)*
6. *Izvršenje*



II - Razvoj C programa



II – Razvoj C programa

- Svakoj izradi programa prethodi analiza problema i izrada algoritama za rešenje problema
- U toku pisanja programa često je potrebno ispravljati sintaktičke greške koje mogu da se dogode.
- Ukoliko se program ne izvršava u potpunosti, moguće je postojanje greške u korišćenju računarskih resursa (na primer korišćenje memorije).
- Postojanje takvih grešaka se ispituje posebnim programima – dibagerima (*debugger*).
- Postupak programiranja ne može biti završen ako program pri izvršavanju iskazuje nelogične rezultate.
- Tada ne preostaje ništa drugo nego da se krene od početka i da se ponovo kritički sagleda zadatak programiranja.

II – Struktura C programa

- Izvorni program (*source code*) u C-u je običan tekstualni fajl, kreiran u bilo kom editoru teksta (npr. *Notepad*).
- Svaki C program sastoji se od **funkcija i promenljivih**.
- **Funkcija sadrži naredbe** koje određuju računarske operacije koje treba da se obave prilikom izvršavanja programa
- **Promenljive** čuvaju **vrednosti** koje se koriste tokom izračunavanja.
- C funkcije su slične potprogramima i funkcijama u *Fortranu* ili procedurama i funkcijama u *Pascalu*.
- Jedan od metoda za razmenu podataka između funkcija je da pozivajuća funkcija obezbedi listu vrednosti, tzv. **argumenata**, za funkciju koju poziva.
- Zagrade iza imena funkcije okružuju listu **argumenata**.
- Ako funkcija ne očekuje ni jedan argument, to se označava praznom listom ( ).
- Naredbe u C-u su razdvojene znakom '**;**'-ima ulogu separatora naredbi

II – Struktura C programa

- Naredbe od kojih se sastoji funkcija čine ***telo funkcije***.
- Telo funkcije se navodi iza ***zaglavlja funkcije***, unutar velikih (vitičastih) zagrada **{ }**.
- Funkcija koja mora da postoji u svakom programu je **main()**.
- Izvršavanje programa se svodi na izvršavanje tela ove funkcije.
- Telo funkcije se navodi iza zaglavlja funkcije main(), između velikih zagrada **{ }**
- U telu funkcije se na početku navode deklaracije pomenljivih, nakon čega sledi proizvoljan niz naredbi.

```
int main()/*zaglavlje funkcije main*/
{       /*ovde počinje telo funkcije*/
        /*deklaracije promenljivih*/
        /*naredbe*/
}       /* kraj tela funkcije */
```

II – Primer C koda

Primer: ***Napisati program koji na standardni izlaz ispisuje poruku ”Zdravo, svete!”***

```
#include <stdio.h>
int main()
{
printf("Zdravo, svete!\n");
return 0;
}
```

- ***Objašnjenje:***
- **#include <stdio.h>** - ukazuje prevodiocu da uključi informacije o standardnoj ulazno/izlaznoj biblioteci; ovaj red se nalazi na početku mnogih izvornih datoteka u C-u.
- **int main()** – obavezna funkcija u svakom C programu
- **{ }** – velike zagrade ograničavaju svako telo funkcije main()
- **printf("Zdravo, svete!\n“)** - funkcija kojom se na standardnom izlazu ispisuje poruka ***”Zdravo, svete!”.***
- **return 0** – izlaz iz funkcije gde 0 označava da nema nekog rezultata

II – Primer C koda

- Svaka funkcija se poziva navođenjem njenog imena i iza toga liste argumenata u zagradama koji su potrebni za izvršenje ove funkcije.
- Sekvenca znakova između dvostrukih navodnika, kao što je „Zdravo, svete!\n“, naziva se znakovni string ili string konstanta.
- Sekvenca **\n** u stringu je C notacija koja označava novi red, koji kada se navede proizvodi prelazak u levu marginu novog reda.
- Ako se izostavi onda se ne dolazi do prelaska u novi red

Primer: *Šta je izlaz iz sledećeg programa?*

```
#include <stdio.h>
int main()
{
printf("Zdravo, ");
printf("svete!");
printf("\n");
return 0;
}
```

Rešenje

Zdravo svete!

II – Primer C koda

➢ *Uvođenje promenljivih u program.*

```
#include <stdio.h>
int main()
{
/* Deklaracija vise promenljivih istog tipa */
int rez, pom1, pom2;
pom1 = 20;
pom2 = 15;
rez = pom1 - pom2;
/* Ispisivanje rezultata */
printf("Rezultat je %d - %d = %d\n", pom1, pom2, rez);
return 0;
}
```

Izlaz iz programa je: **Rezultat je 20 – 15 = 5**

II – Okruženje za pisanje programa

- Microsoft Visual Studio 2013 predstavlja integrisano okruženje za razvoj softvera. Sastoji se od sledećih celina:

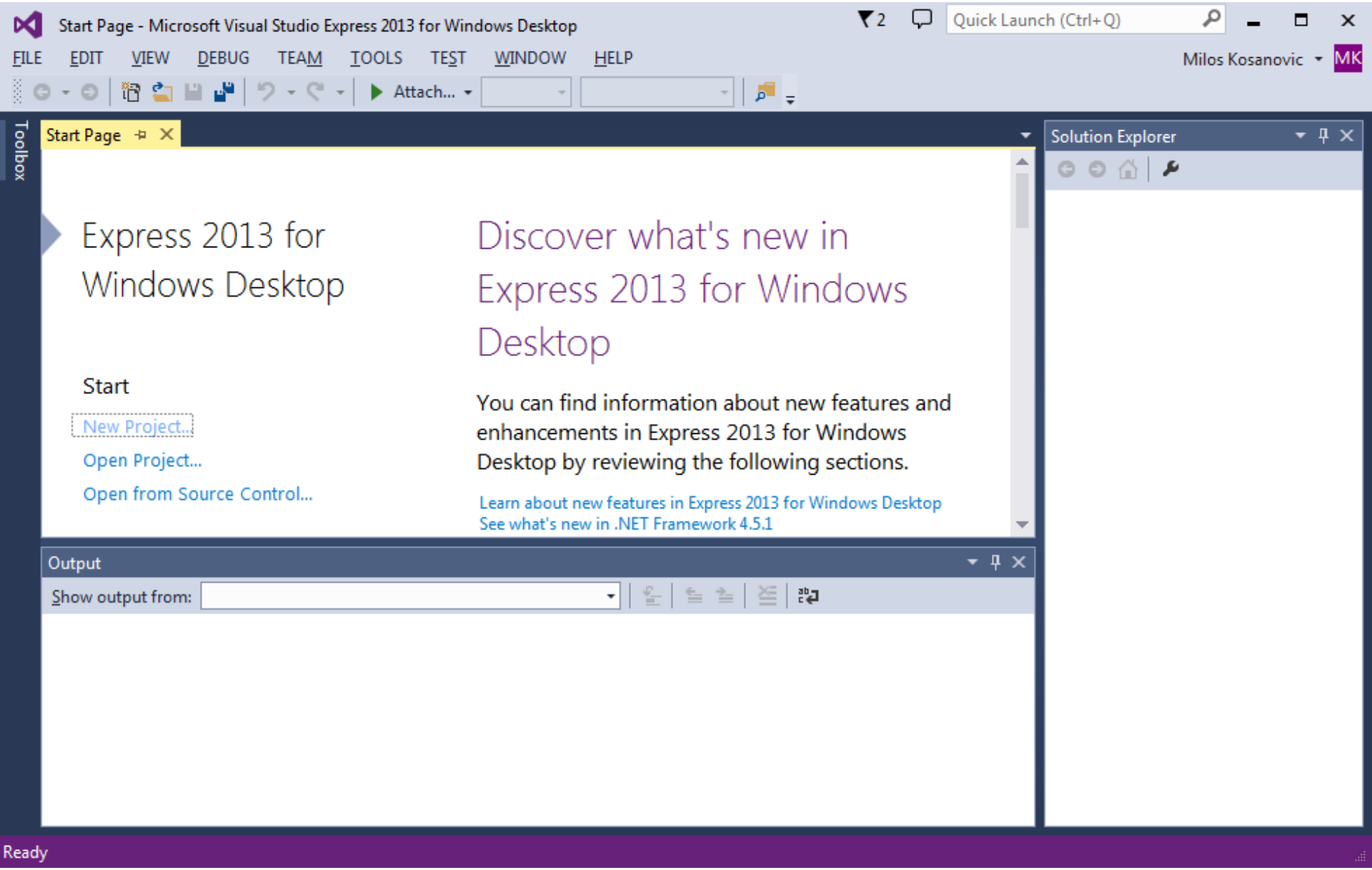
  Visual C++,
  Visual Basic,
  Visual C#
  MSDN Library.
  Druge…

- Visual studio C++ je okruženje koje omogućava razvoj i pisanje programa u program. jeziku C++, a samim tim i u prog.jeziku C.

Pokretanje Visual Studio 2013 okruženja

- Okruženje Visual Studio 2013 se može otvoriti klikom na taster **Start** i izborom stavke **Microsoft Visual Studio 2013** iz liste programa.
- Nakon startovanja otvoriće se osnovi prozor okruženja Visual Studio 2013 koji je prikazan na sledećoj slici:

II - Okruženje za pisanje programa

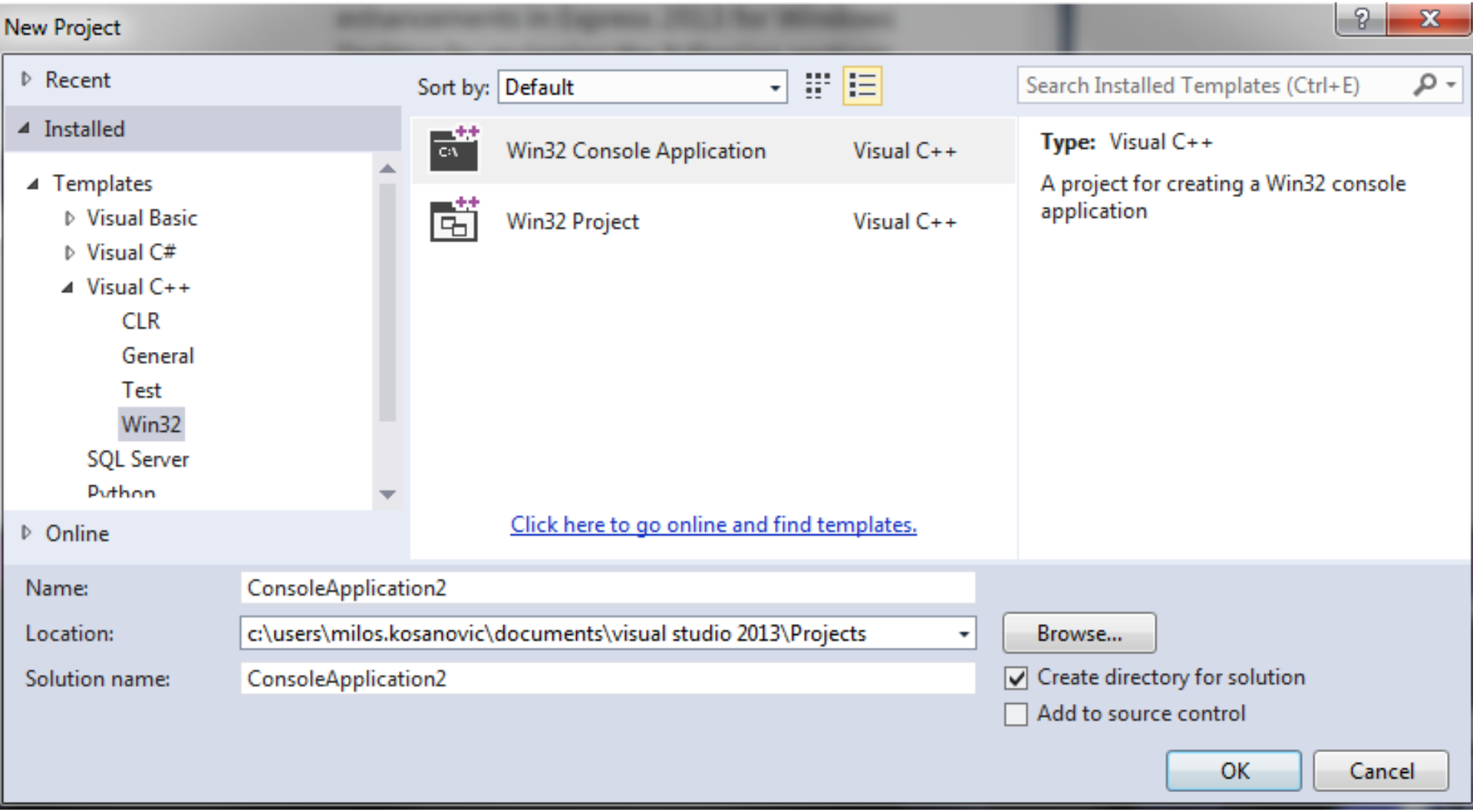


II – Okruženje za pisanje programa

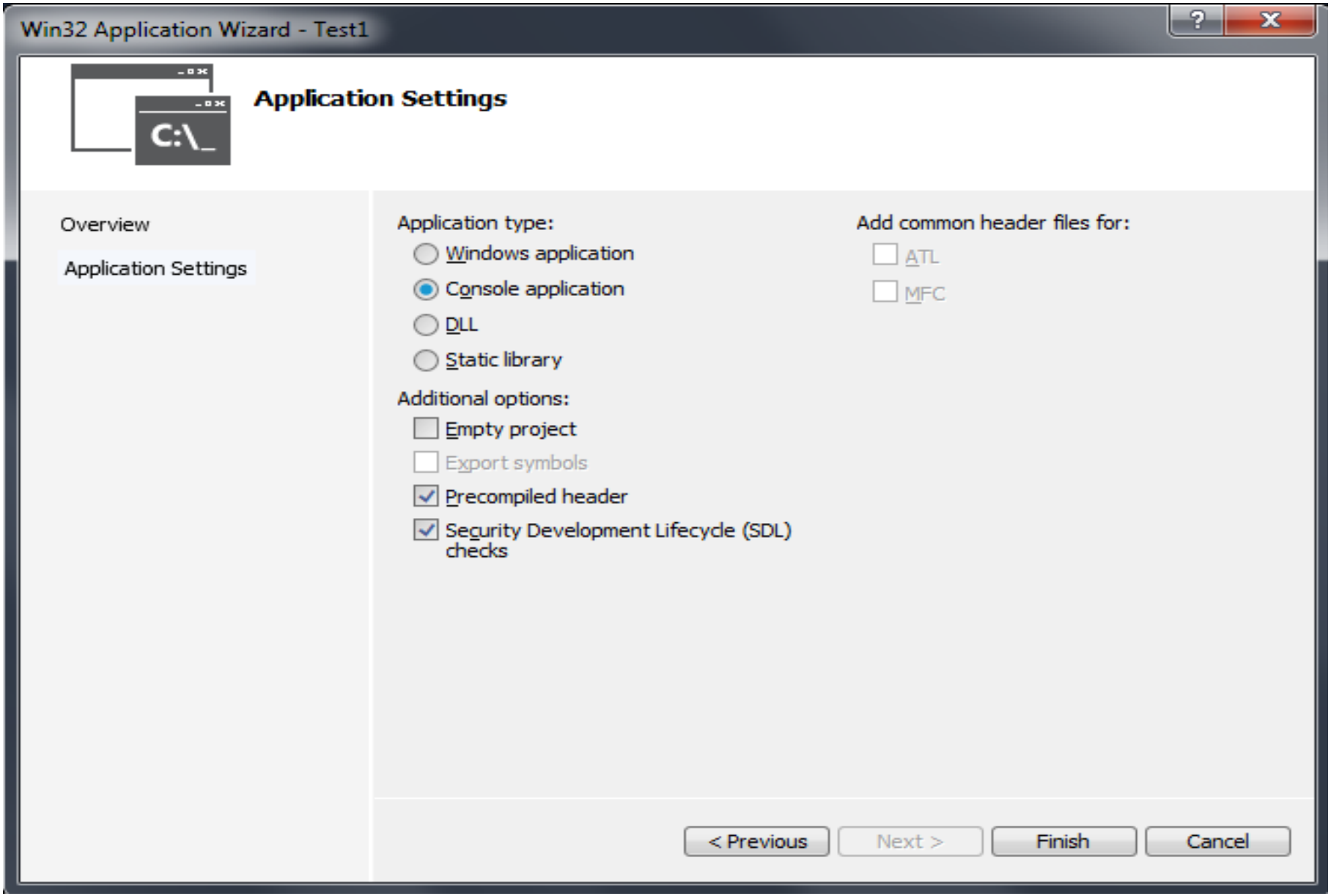
Kreiranje projekta za konzolnu aplikaciju

- Da bi se napisao, kompilirao, izvršio i eventualno debagirao C program, u Visual Studiju je potrebno kreirati projekat koji sadrži odgovarajuće datoteke sa izvornim kodom programa.
- Za potrebe pisanja programa u C-u biće korišćen najjednostavniji tip projekta koji obezbeđuje pravljenje konzolnih aplikacija.
- Konzolne aplikacije su programi koji se izvršavaju u komandnom (DOS) prozoru i koriste standardni ulaz i izlaz.
- Kreiranje projekta se vrši startovanjem stavke iz menija **File/New**, izborom kartice **Project** i stavke **Visual C++** i zatim **Win32 Console Application** (vidi sliku na sledećem slajdu).
- Pri kreiranju projekta potrebno je uneti naziv projekta u polje **Project name**, i eventualno promeniti direktorijum na disku gde će projekat biti kreiran (polje **Location**).
- Po unosu naziva projekta aktiviraće se taster **OK**.
- Klikom na ovaj taster pojaviće se prozor za izbor **tipa konzolnog projekata** koji se želi kreirati (sledeći slajdovi).

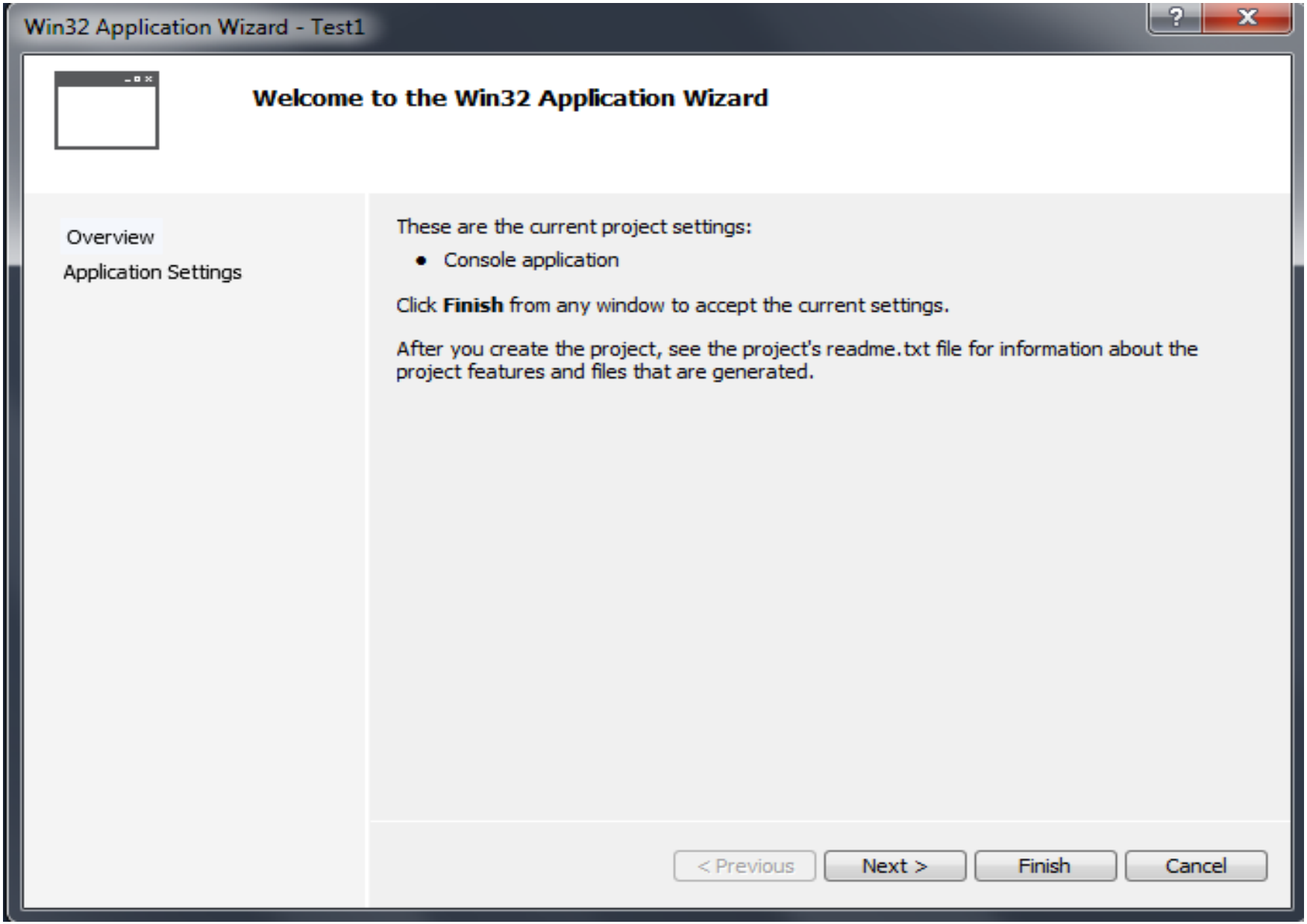
II - Okruženje za pisanje programa



II - Okruženje za pisanje programa



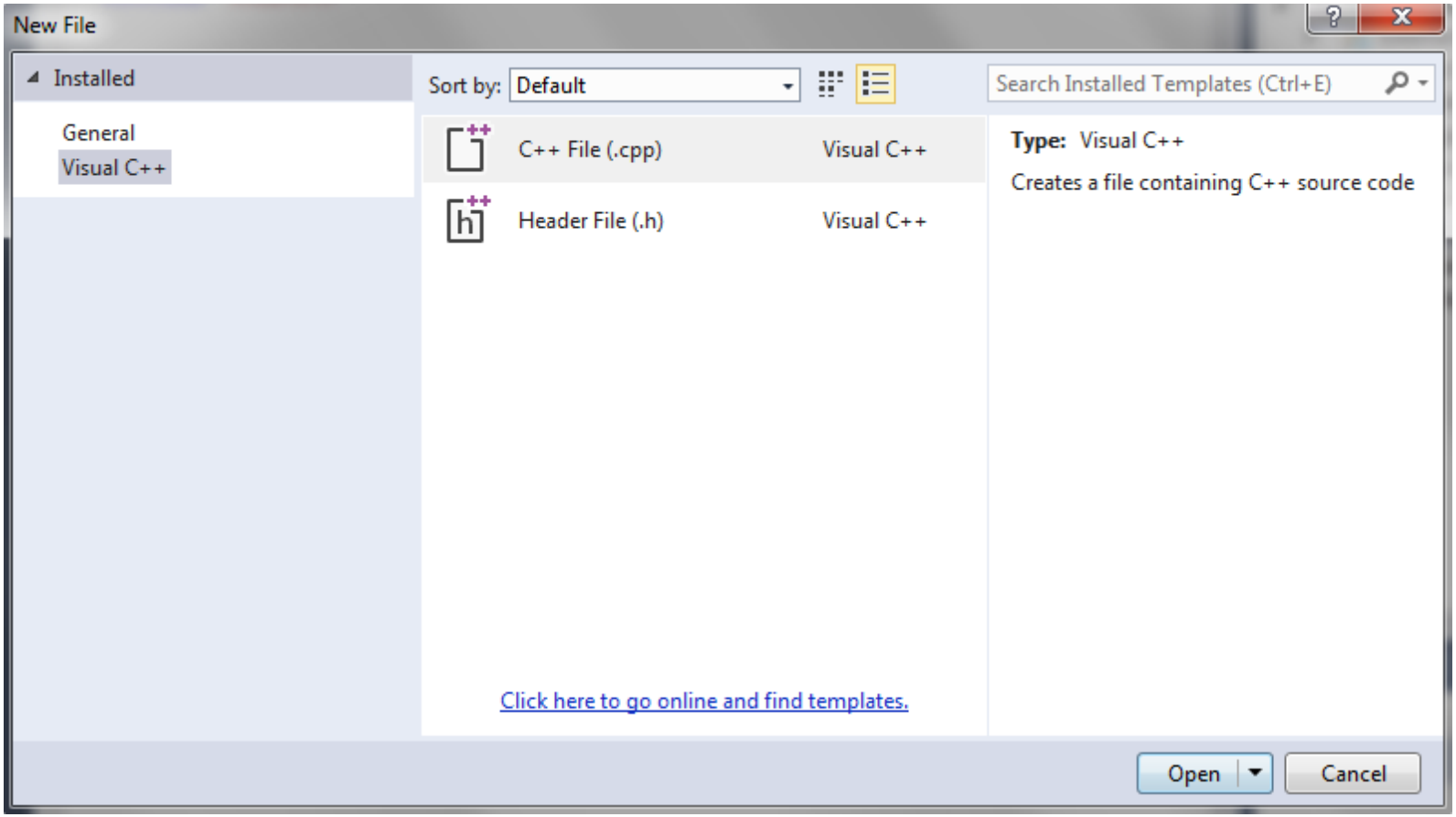
II - Okruženje za pisanje programa



II – Okruženje za pisanje programa

- U ovom slučaju ne treba ništa menjati jer je već odabrana opcija za kreiranje **praznog projekta** (*An empty project*).
- Da bi se kreirao projekat potrebno je kliknuti na taster **Finish**.
- Ukoliko želite možete kliknuti i na dugme **Next**.
- U tom slučaju će Vam biti prikazan prozor sa dodatnim opcijama.
- Nakon kreiranja praznog projekta za konzolnu aplikaciju potrebno je kreirati i dodati u projekat datoteke koje će se koristiti za unos izvornog koda programa.
- U VS2013 obično je za Vas već kreiran fajl koji se zove isto kao vaš projekat.
- Ukoliko nije, potrebno je kreirati i dodati datoteku na sledeći način.
- Pozivate stavku iz menija **File/New File** i stavke Visual C++.
- Zatim možete izabrati između **C++ File** ili **Header File**.
- Potrebno je uneti željeni naziv datoteke u polje **File name** i kliknuti na taster **OK**.

II – Okruženje za pisanje programa



- Po kreiranju i dodavanju nove datoteke u projekat potrebno je uneti odgovarajući sadržaj (ukucati program).
- Za to se koristi centralni deo glavnog prozora
- Visual Studio poseduje neke napredne opcije za prikaz izvornog koda programa kao što su bojenje ključnih reči i komentara.

II – Osnovni pojmovi u C jeziku

- **Izraz** (*Expression*) – kombinacija literalnih vrednost, promenljivih, operatora ili funkcija
- **Opseg važenja** (*Scope*) – deo programa u kome je ime nekog entiteta (promenljive ili funkcije) validno i može se iskoristiti za pristup tom entitetu
- **Deklaracija (definicija)** promenljive – određivanje tipa promenljive prilikom koje se rezerviše memorijski prostor za taj tip promenljive. ***Primer***: int broj (rezerviše 4B), char p (rezerviše 1B)
- **Inicijalizacija** promenljive – dodeljivanje inicijalne vrednosti nekoj promenljivi
- **Komentar** – komentar jedne linije ili komentar dela programa
- **Promenljiva** (*Variable*) – ime ili referenca na zapamćenu vrednost (obično u memoriji)
- **Tip podatka** (*Data type*) – određuje veličinu promenljive u memoriji, definiše koje vrednosti ona može da uzme kao i koje operacije su dozvoljene sa njom

II – Leksička pravila

- Program se piše povezivanjem osnovnih simbola u logičke izraze
- Koriste se velika i mala slova, cifre i specijalni simboli iz ASCII skupa
  - ✓slova engleske abecede **A, B, C, ... , X, Y, Z, a, b, c, ... , x, y, z**
  - ✓cifre **0, 1, 2, 3, 4, 5, 6, 7, 8, 9**
  - ✓specijalni znaci: **+ = _ - ( ) * & % # ! | . , ; : ' / ? { } ~ \ [ ] ^**
  - ✓neštampajući znaci blanko (*space*), znak za novu liniju, tab, ...
- Programski jezik C razlikuje velika i mala slova! - ***case sensitive***.

int x, X; /*To su dve različite promenljive!!!*/

- **Komentari** služe za objašnjenja u toku pisanja programa.
- Pomoću komentara program postaje čitljiviji i jasniji za čoveka
- Prevodiocu (kompajleru) je potpuno sve jedno da li postoji sto hiljada linija komentara ili ni jedna jedina jer ih on jednostavno ignoriše.
- Komentar počinje znakom **/*** , a završava se znakom ***/**
- Ako se koriste znakovi **/*** i ***/** komentari se mogu prostirati u više linija ali ne mogu biti ugnježdeni.
- Ako je za komentar dovoljan jedan red teksta možemo koristiti i znakove **//** na primer: // ovo je komentar

II – Elementi prog.jezika – Tokeni

- Osnovni element svakog programskog jezika je reč ili **token**
- **Reč** (*token*) je niz znakova koji predstavljaju elementarnu logičku celinu koja ima neko značenje u programskom jeziku.
- Tokeni se razdvajaju belinama(*blanko*), tabulatorima i novim redovima
- Prilikom prevođenja kompajler prepoznaje tokene i proverava u sintasniom analizatoru da li je njihov redosled odgovarajući
- Postoji šest vrsta tokena:

1. Ključne reči
2. Identifikatori
3. Separatori
4. Konstante
5. Stringovi (Literali)
6. Operatori

II – Ključne reči

- To su rezervisane reči koje imaju unapred definisano značenje
- Postoje 32 ključne reći i one se ne mogu koristiti kao identifikatori.
- Ključne reči se koriste za: definisanje jezičkih konstrukcija (*if*, *while*, *for)*, imena tipova (*int*, *float*, *char)*, *itd.*
- Sve ključne reči u programskom jeziku C pišu se **malim slovima**.

auto	**double**	**int**	**struct**
break	**else**	**long**	**switch**
case	**enum**	**register**	**typedef**
char	**extern**	**return**	**union**
const	**float**	**short**	**unsigned**
continue	**for**	**signed**	**void**
default	**goto**	**sizeof**	**volatile**
do	**if**	**static**	**while**

II – Identifikatori

1. Identifikatori (simbolička imena)

- Identifikatore čine reči koje se sastoje od: slova, cifara i znaka '_'
- Svaki identifikator mora da počne sa slovom ili znakom '_'
- Koriste se za označavanje osnovnih objekata jezika: konstanti, labela, promenjivih, funkcija i korisničkih tipova podataka.
- Identifikator može biti proizvoljne dužine, ali stari kompajleri tretiraju:
  - **31** znak za interna imena koja se definišu i koriste samo u jednom fajlu
  - **6** znakova za imena koja se koriste u više fajlova.
- Primeri ispravnih imena: **skola, x, a, zbir, Programski_jezik_C, INT**
- Nedozvoljeno je: **3dan, int, x+y, programski jezik, -povrsina, $a, …**

Primer:

```
Ulaz: N;
F=1;
for i=2, N do
    F=F+i
end
Prikaz: F;
```

II – Separatori

- Separatori su posebni specijalni znaci koji imaju određeno (unapred definisano) sintaksno i semantičko značenje u programu
- Separatori ne označavaju nikakvu operaciju već razdvajaju druge tokene u grupe.

U ovu grupu reči spadaju:

1. **{** - početak neke programske celine (bloka)
2. **}** - kraj programske celine (bloka)
3. **(** - početak liste parametara funkcije
4. **)** – kraj liste parametara u funkciji
5. **;** - kraj naredbe (ne predstavlja prelazak u novi red)
6. **//** - oznaka za linijski deo koda – komentar, koji kompajler ne prevodi
7. **/* ... */** - oznaka za blokovski deo koda – komentar, koji kompajler ne prevodi

Hvala na pažnji !!!

Pitanja

? ? ?