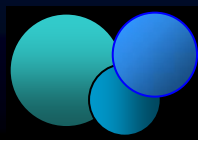
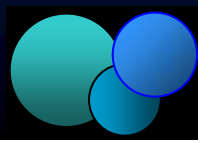
Основи програмирања 1

Висока школа електротехнике и рачунарства
струковних студија
Београд



Садржај

- Шта је програмски језик
- Машински језик и асемблери
- Виши програмски језици
- Развојни циклус програма
- Увод у програмски језик „*C*”
- Основе развојног окружења



Програмски језици

Machine Language

COBOL

Visual Basic

Fortran

BASIC

C / C++

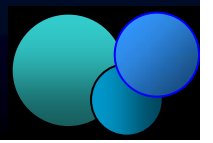
Assembly Language

Pascal

Ada

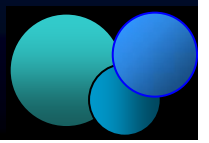
Java

- **Програмски језик** је скуп наредби за извршавање рачунарских задатака
- Представља скуп речи и скуп правила на основу којих се пише програм
- **Развојни цилкус програма** скуп правила на којима се заснива развој софтвера



Развој програмских језика

- Програмски језици су подељени по генерацијама
- Језици најнижег нивоа су најстарији
- Постоји пет генерација програмских језика:
  - Машински језици
  - Асемблерски језици
  - Процедурални језици
  - Проблем-оријентисани језици
  - Природни језици

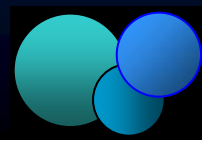


Језици прве генерације

Машински језици:

- Састоје се од бинарних бројева (0 и 1)
- Није потребно превођење
- Везан је за конкретан рачунар (машину)
  - Свака фамилија процесора има свој (посебан) машински језик

```
Program Fragment:        Y = Y + X

Machine Language Code
(Binary Code)

Opcode          Address
1100 0000       0010 0000 0000 0000
1011 0000       0001 0000 0000 0000
1001 0000       0010 0000 0000 0000


Memory Cell Definitions:

     Addr.      Name       Cell Contents

     1000       X          32
     2000       Y          16
```

Језици друге генерације

Асемблерски језици:

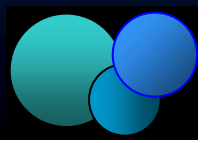
- Надоградња машинских језика
- Нижи програмски језик
- Користи кратке словне замене за програмске наредбе.
  Ове словне замене се називају **мнемоници**
- Програм се прво пише као *source code* (текстуални фајл) а потом се преводи у машински језик
- Програм за превођење: Асемблер

Assembly Language Code

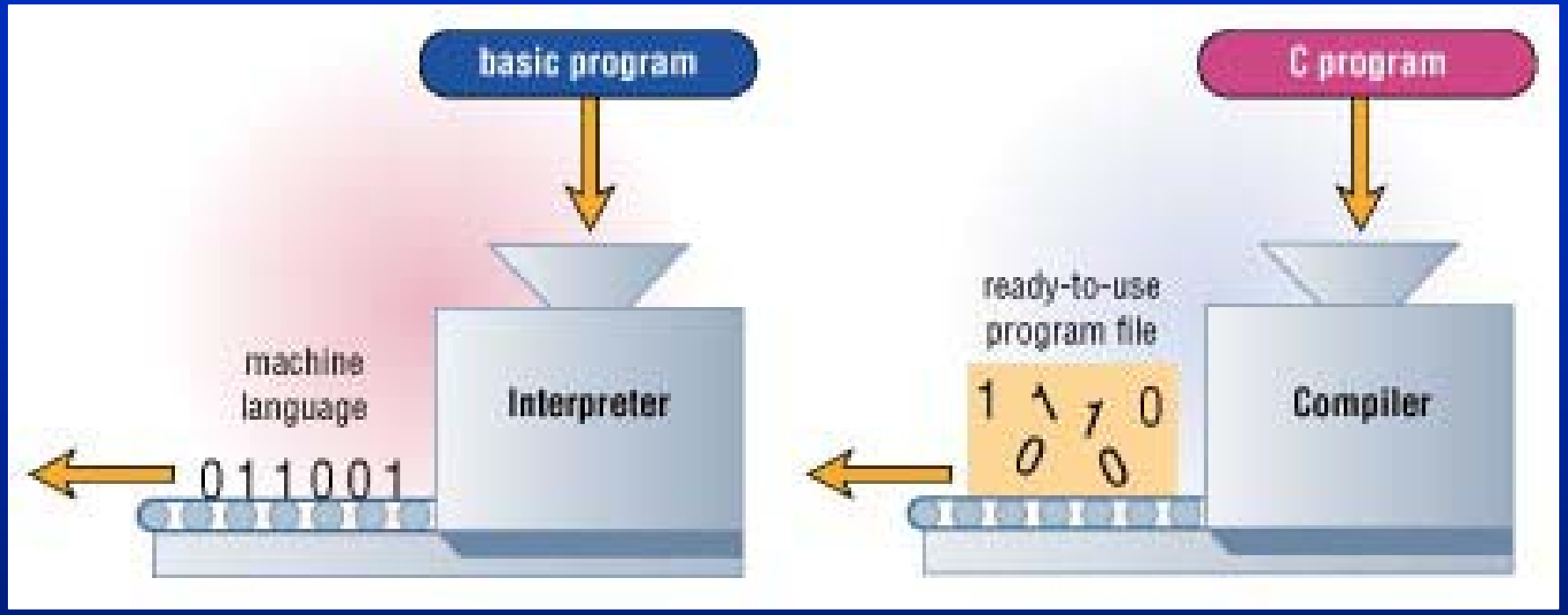
```
LOAD  Y
ADD   X
STORE Y
```

Језици треће генерације

- Процедурални језици:
  - Језици високог нивоа апстракције
  - Лакши за читање, писање и преправке од машинских и асемблерских језика
  - Користе *compiler* или *interpreter* за превод кода
- *Fortran* и *COBOL* су језици треће генерације



Compiler & Interpreter

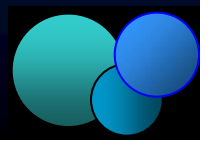


- **Компајлер** је програм који преводи код (*source code*) у објектни код
- **Интерпретер** преводи по једну линију кода и одмах је извршава

Језици треће генерације (наставак)

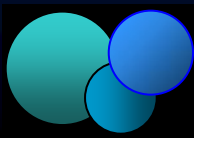
***Spaghetti Code & the Great Software Crisis*:**

- **GOTO** наредба довела је до тога да се програм тешко прати
- Тај проблем је довео до тзв. софтверске кризе 1960-тих
  - Рокови за програмирање су се пробијали
  - Програми су "пробијали" предвиђени буџет
  - Програми су садржали превише грешака
  - Корисници нису били задовољни



Језици треће генерације (наставак)

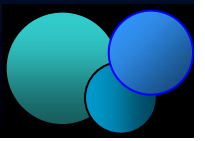
- **Структуирани програмски језици:**
  - Развијени су како би унапредили развој софтвера
  - Представници *Algol* и *Pascal*
  - Забрана употребе **`GOTO`** наредби
  - Употреба контролних структура **`IF-THEN-ELSE`**



Језици треће генерације (наставак)

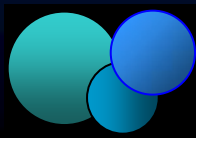
- **Модуларни програмски језици:**
  - Развијени због проблема који су настали у структуираним програмским језицима
  - Користе се за креирање програма који су издељени на засебне модуле

    Сваки модул обавља специфичну функцију
  - за различите улазне вредности даје различите излазне вредности



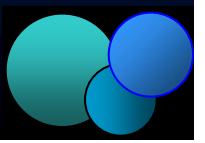
Развојни циклус програма

- Планско решавање проблема програмирања и подела на значајне целине
- Шест фаза:
  1. Дефинисање проблема
  2. Дизајнирање програма
  3. Писање кода (*Coding*)
  4. Тестирање и дебаговање
  5. Формализовање решења
  6. Имплементација и праћење рада програма



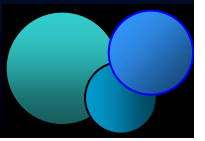
Увод у програмски језик „C”

- Једноставан
- Флексибилан
- Преносив
- Модуларан



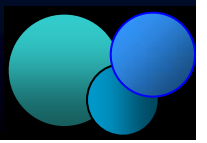
Историја

- Програмски језик ***C*** је први пут развијен почетком 1970-тих (*Bell Labs, USA*)
- Намена: основа за развој оперативног система *UNIX*.
- Други виши програмски језици тог времена (*COBOL, FORTRAN, PL/I, ALGOL...*) су били превише спори.
- Као основа за програмски језик ***C*** су послужила два виша програмска језика: *ALGOL* и *BCPL*.

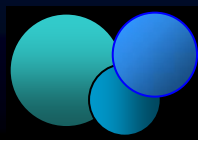


Писање програма

- Скуп наредби (*instructions, statements*) се назива **кôд** (*Code*) или **изворни кôд (***Source code*).
- Фајл који садржи кôд се назива ***Source file***.
- У ***C*** програмском језику се низ наредби групише у целине које се називају **функције** (*function*)



- Сваки програм мора да садржи бар једну функцију и то функцију
  `main( )`
- Функција **`main( )`** се не позива, то ради оперативни систем.
- Свака функција се састоји од
  - повратне вредности (**`int`**)
  - имена (**`main`**)
  - листе улазних аргумената (листа унутар заграда **`()`**)
  - тела (унутар заграда **`{}`**)



C програм

```
main( )
{

}
```

```
main( )
{
   ;
}
```

```
int main( )
{
 return 0;
}
```

```
/* komentar */
int main( )
{
 return 0;
}
```

```
#include <stdio.h>
/* ispisujem: Zdravo! */
int main( )
{
   printf("Zdravo!\n");
   return 0;
}
```

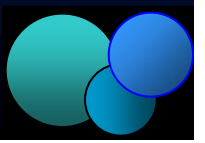
C програм

```
main( )
{


}
```

- Име функције **main**
- Тело функције **{ }**
- Најкраћи програм

```
int main( )
{
 return 0;
}
```

- Тип функције **int**
- наредба **return 0**
- завршетак наредбе **;**
- повратна вредност **0**

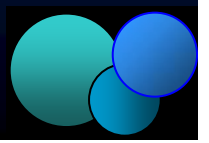


return 0

- Функција **`main()`** враћа целобројну вредност
- Коме?
- Наредба којом се та вредност враћа и којом се одређује која вредност се враћа

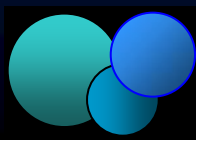
```
return 0;
```

- Оперативном систему се враћа вредност 0
- Касније детаљније о повратној вредности



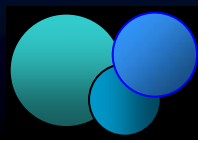
Пут до извршног програма

1. Предобрада (*Preprocessing*)
2. Превођење (*Compilation*)
3. Повезивање (*Linking*)



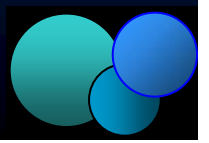
1. Предобрада # (*Preprocessing*)

- Обавља се програмом који се назива ***preprocessor***.
- Мења изворни кôд (у *RAM*-у) у складу са наредбама за предобраду (*preprocessor directives*). Ове наредбе ближе дефинишу начин превођења.
- Занемарују се коментари (*comments*) и празан простор (*white space*) из кôда.
- Кôд, који је програмер написао и претходно снимио на диск, се НЕ МЕЊА у процесу предобраде!



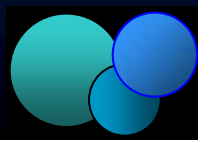
2. Превођење (*Compilation*)

- Програм се назива преводилац (***compiler***)
- Изворни кôд, претходно подвргнут предобради, се преводи у **објектни кôд** (***object code*****).**
- Проверава се синтакса
  грешке (***errors***) и
  упозорења (***warnings***)
- Снима објектни кôд на диск (нови фајл).
- Ако се приликом превођења констатује грешка, НЕ ГЕНЕРИШЕ се објектни кôд.

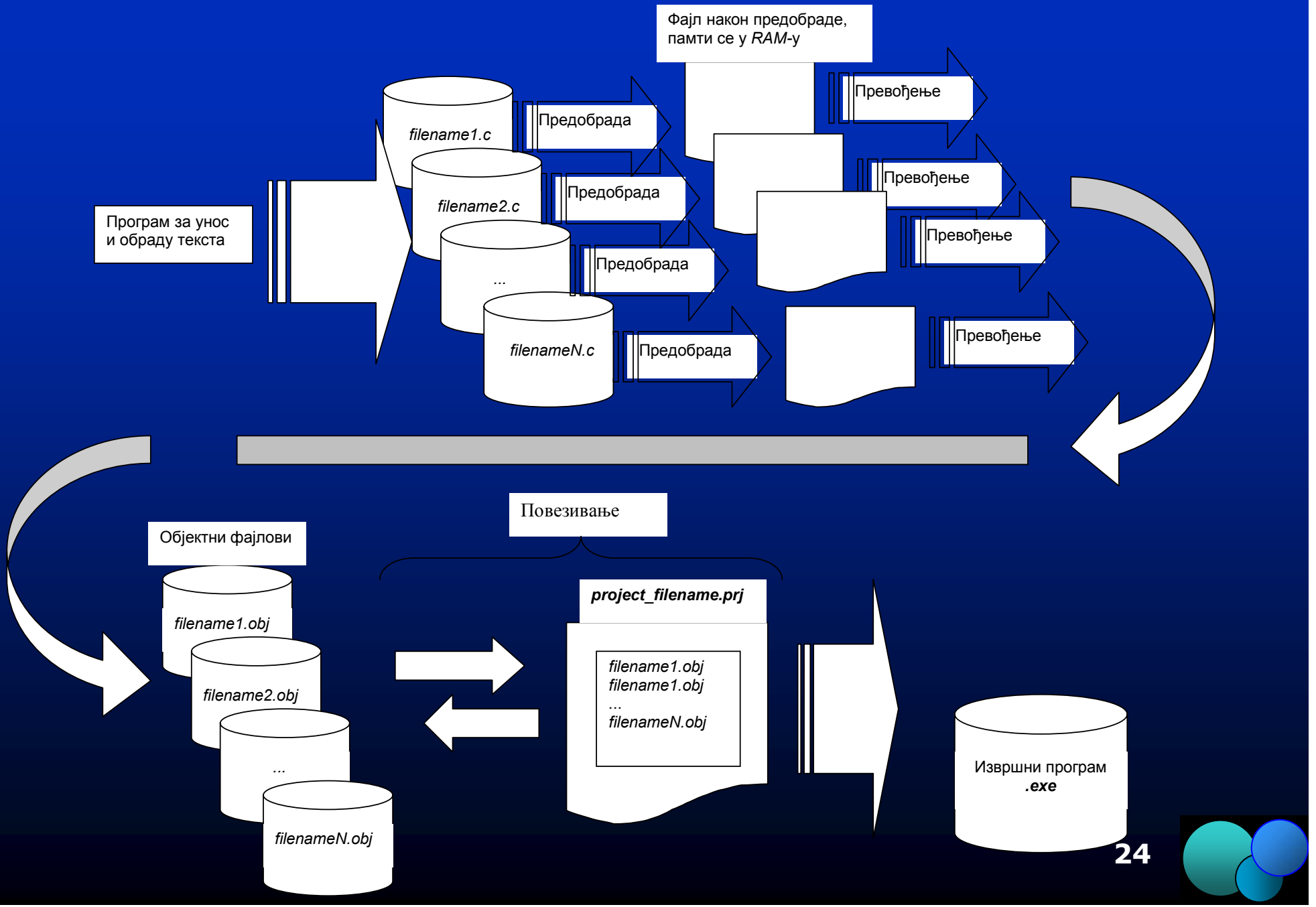


3. Повезивање (*Linking*)

- Уколико је кôд писан у више фајлова постоји више фајлова са објектним кодом.
- Објектни фајлови се повезују у једну целину како би се формирао **извршни фајл** (***executable file***, са екстензијом *.exe*).
- Други објектни фајлови могу да настану позивањем готових функција из тзв. библиотека (*Libraries, Run-Time Library*) или објектних фајлова које је креирао програмер.
- Уколико се приликом линковања констатује грешка неће се формирати извршни фајл!



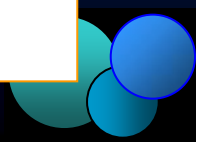
Како до извршног програма?



Формат *C* програма

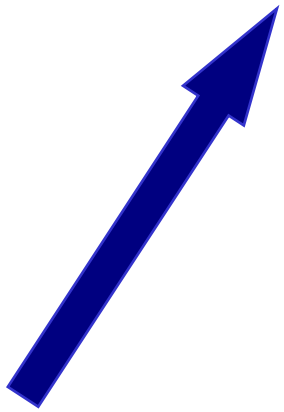
```
#include <stdio.h>
/* C program koji ispisuje: "Zdravo!" */
int main( )
{
    printf("Zdravo!\n");
    return 0;
}
```

```
#include <stdio.h>/* C program koji
  ispisuje: "Zdravo!" */int main
  (){printf("Zdravo!\n");return 0;}
```

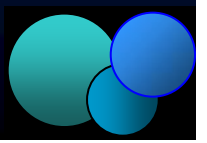


Формат *C* програма

```
#include<stdio.h>
/* C program koji
  ispisuje:
"Zdravo!" */
int
main
(
)
```



```
{
printf(
"Zdravo!\n"
)
;
return
0
;
}
```

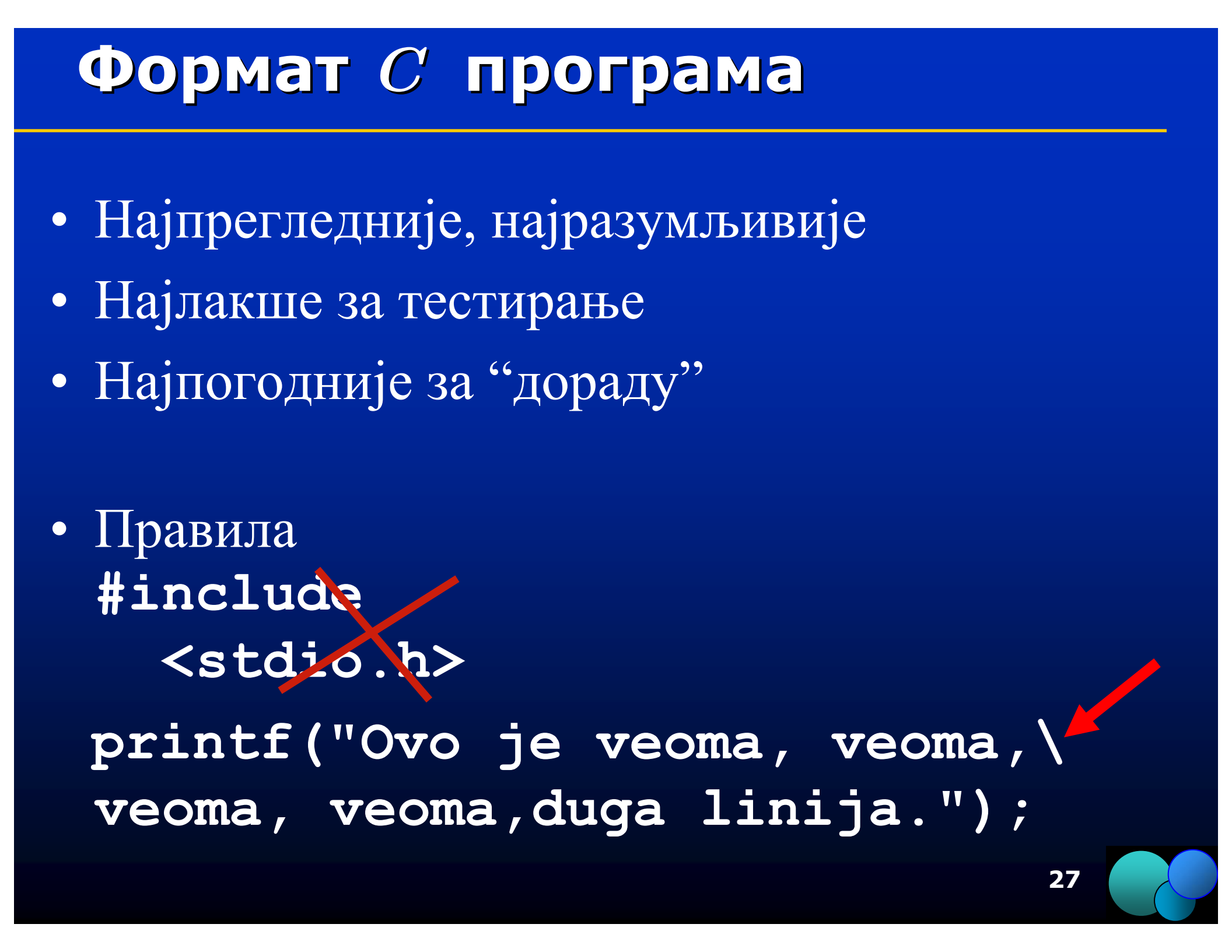



Формат *C* програма

- Најпрегледније, најразумљивије
- Најлакше за тестирање
- Најпогодније за “дораду”

- Правила

```
#include
   <stdio.h>
printf("Ovo je veoma, veoma,\
veoma, veoma,duga linija.");
```



Елементи језика *C*

- **Скуп знакова**
- Велика и мала слова енглеског алфабета

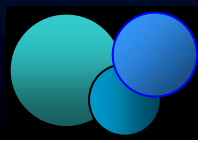
```
Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm
Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz
```

- Децималне цифре

```
0 1 2 3 4 5 6 7 8 9
```

- Знаци:

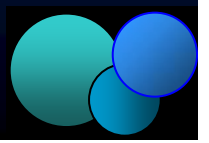
```
:  _  !  ?  .  '  :  ;  `  =  +  -
/  *  #  %  &  \  |  (  )  [  ]  {
}  <  >
```

О коментрима

- Део кода који се не преводи
- Намена му је да програмеру олакша праћење кода
- може да се налази у једној или у више линија

```
/* komentar u celoj liniji */

printf("Ispis"); /* komentar dela linije*/

/* komentar u
   više
   linija */
```

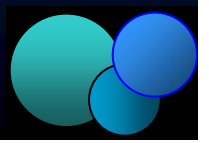



C програм са коментарима

```
/*************************************
 ** Program: zdravo.c                **
 ** Autor: Henry Neeman              **
 ** Kurs: Programiranje1  2006/07    **
 ** Objasnjenje: Ispis "Zdravo svima!" **
 **              standardni izlaz.   **
 *************************************/
#include <stdio.h>

int main() /* pocetak funkcie main */
{
 /******  Telo funkcije  *********/

 /*Prikazi recenicu */
 printf("Zdravo svima!\n");

} /* kraj funkcije main */
```

C програм без коментара

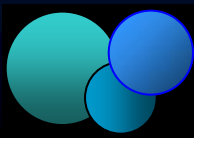
```
#include <stdio.h>

int main()
{
  printf("Zdravo svima!\n");

}
```

О испису

```
#include <stdio.h>
/* C program koji ispisuje: "Zdravo!" */
int main( )
{
    printf("Zdravo svima!");

    return 0;
}
```

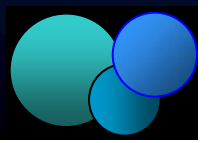
- Излаз
  Zdravo svima!
- **printf** постојећа функција за испис на екран
- исписује све између знакова навода



О испису

```
#include <stdio.h>
int main( )
{
    printf("Zdravo ");
    printf("svima!");

    return 0;
}
```

- Излаз
  Zdravo svima!

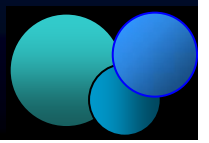


О испису

```
#include <stdio.h>

int main( )
{
    printf("Zdravo \nsvima!");

    return 0;
}
```

- Излаз

  Zdravo
  svima!

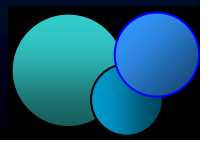


О испису

```
#include <stdio.h>

int main( )
{
    printf("Zdravo \n\nsvima\n!");

    return 0;
}
```

- Излаз

```
Zdravo


svima
!
```

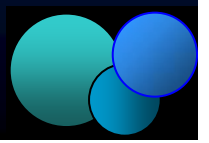



О испису

```
#include <stdio.h>

int main( )
{
    printf("\tZdravo \n\nsvima!");

    return 0;
}
```

- Излаз

```
    Zdravo


svima!
```

О прихвату података

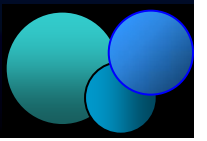
```
#include <stdio.h>
#define MAX 30
int main( )
{
    char ime[MAX];

    printf("\nVase ime?\n\n");
    gets(ime);

    printf("Vi ste %s\n", ime);

     return 0;
}
```

- Излаз

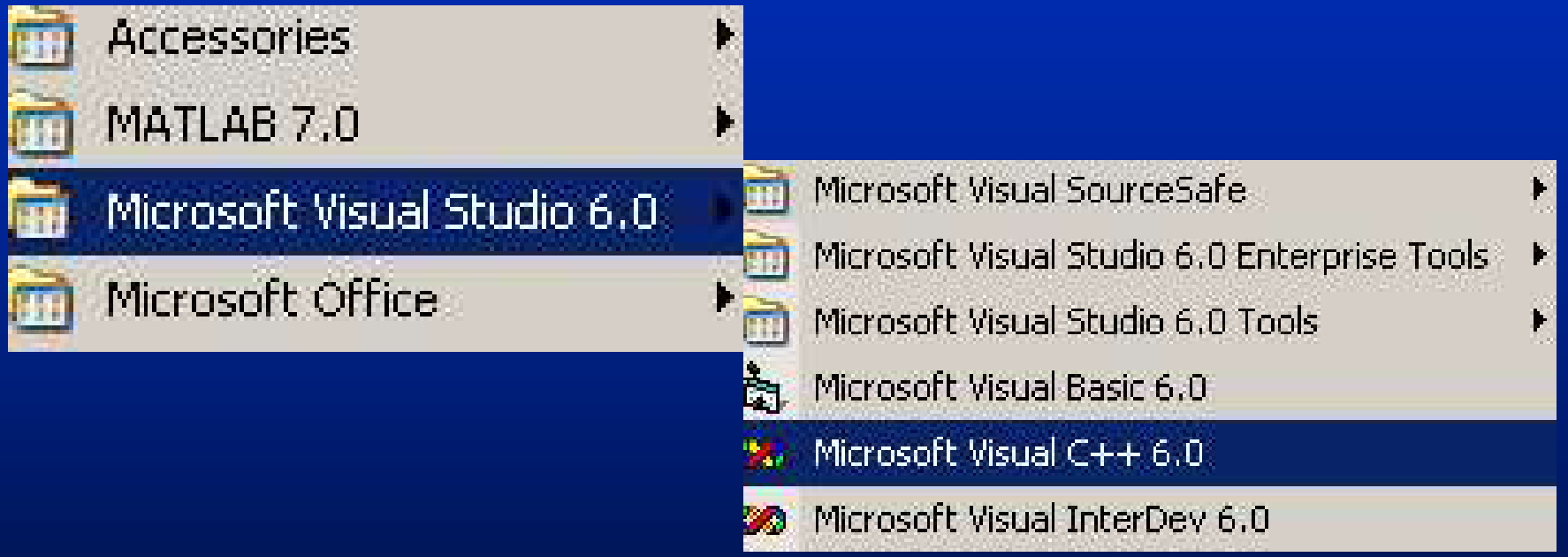
```
Vase ime?
```

```
Vase ime?
Marko
```

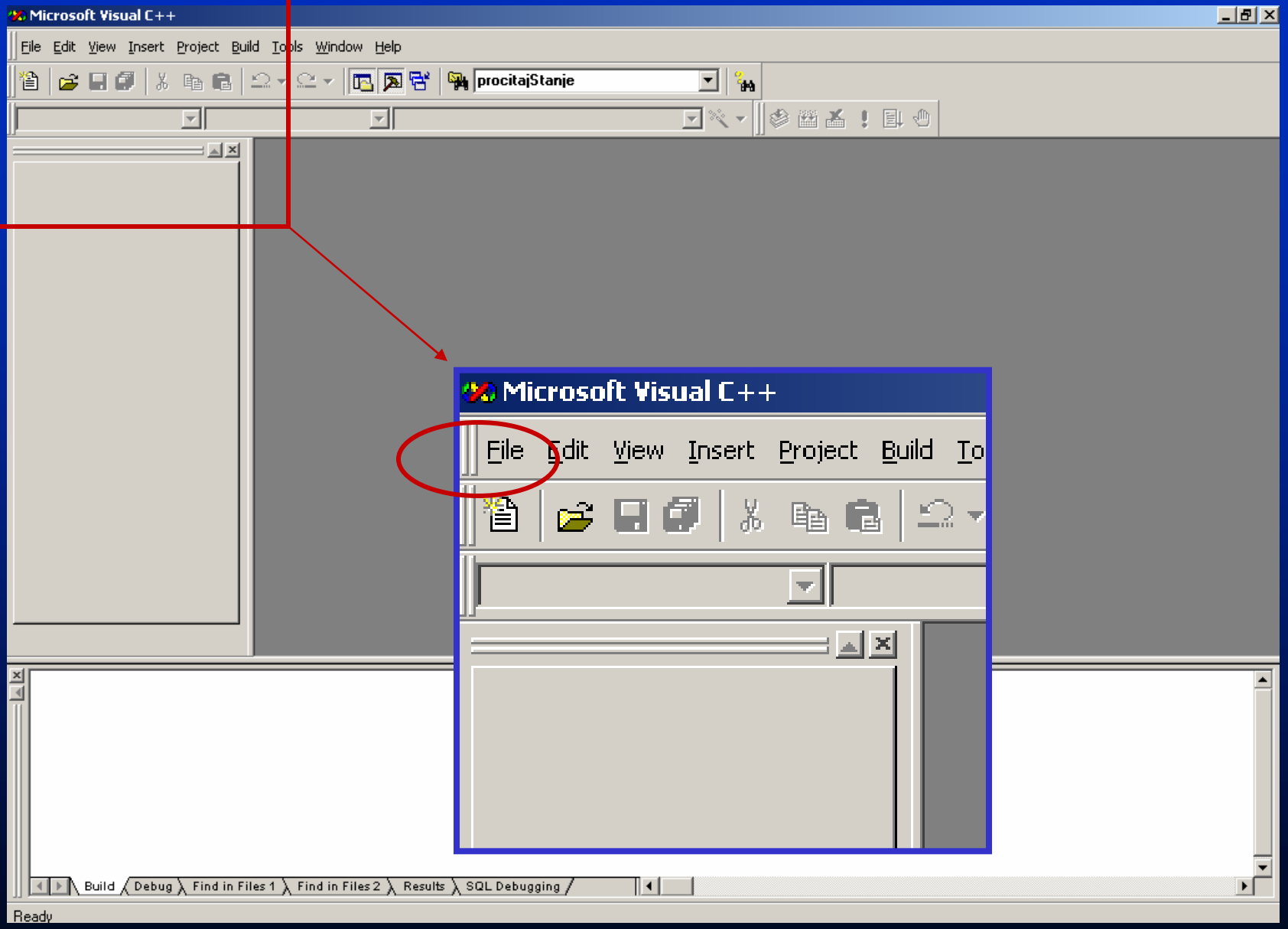
```
Vase ime?
Marko
Vi ste Marko
```

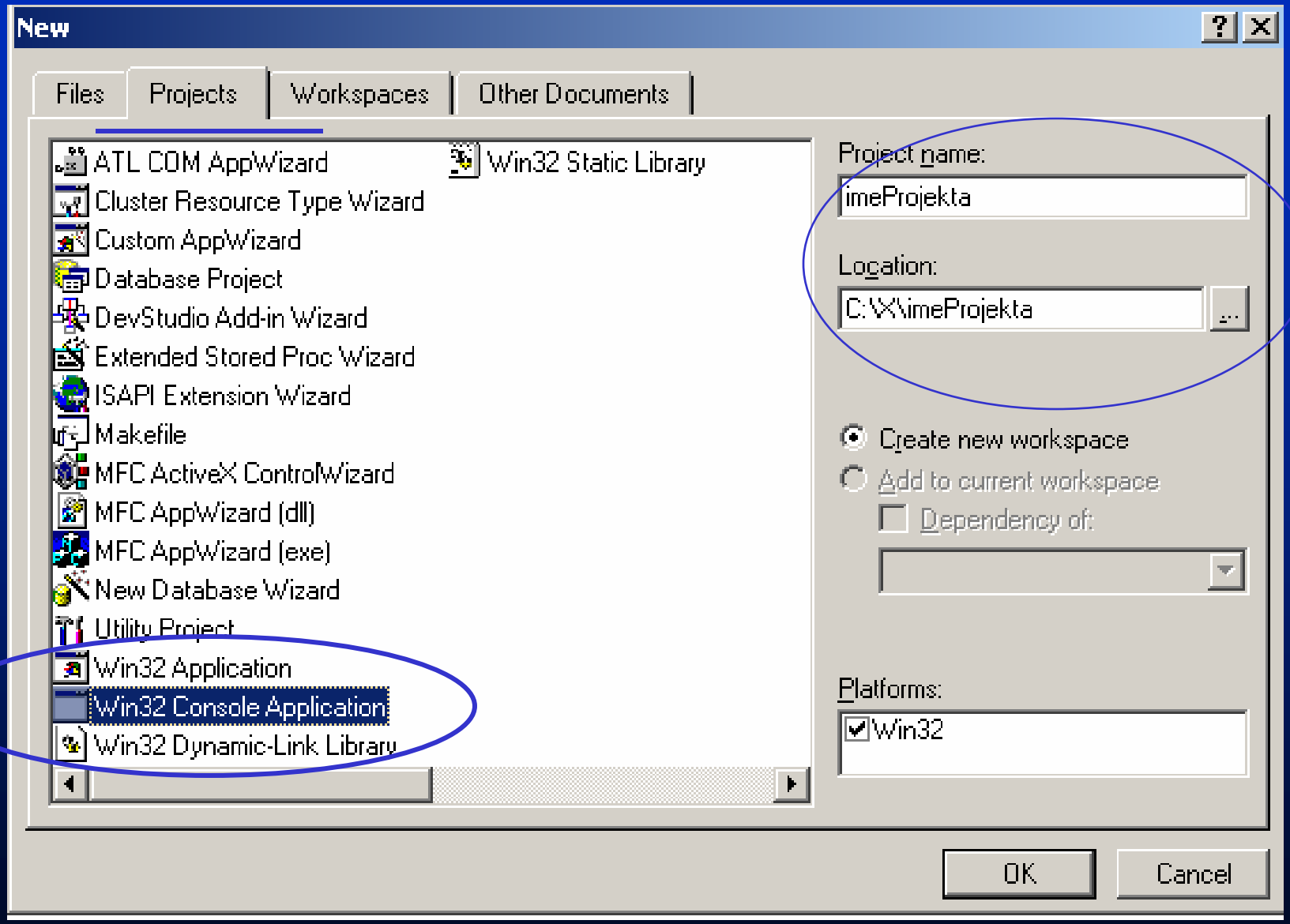
Развојно окружење



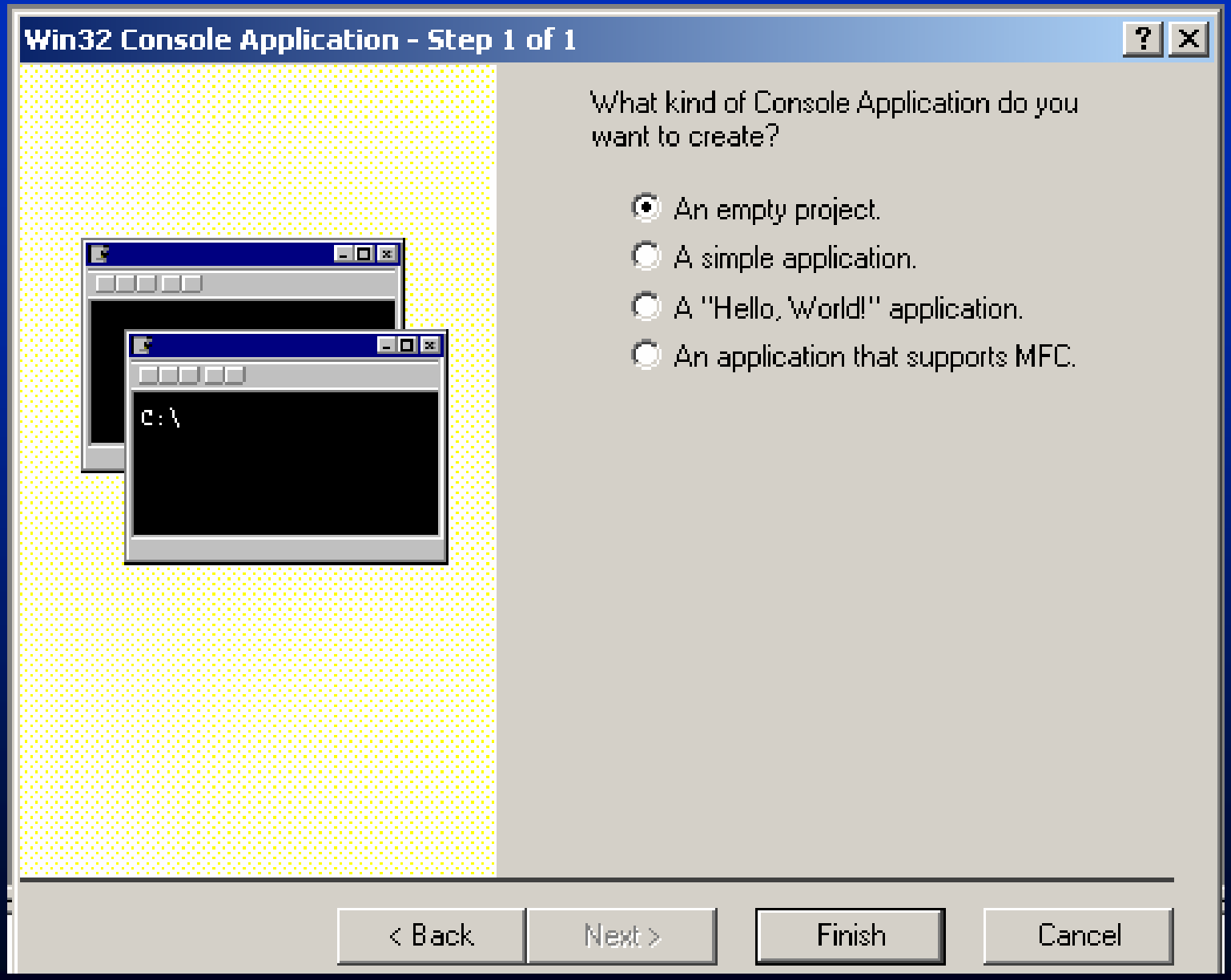
Развојно окружење



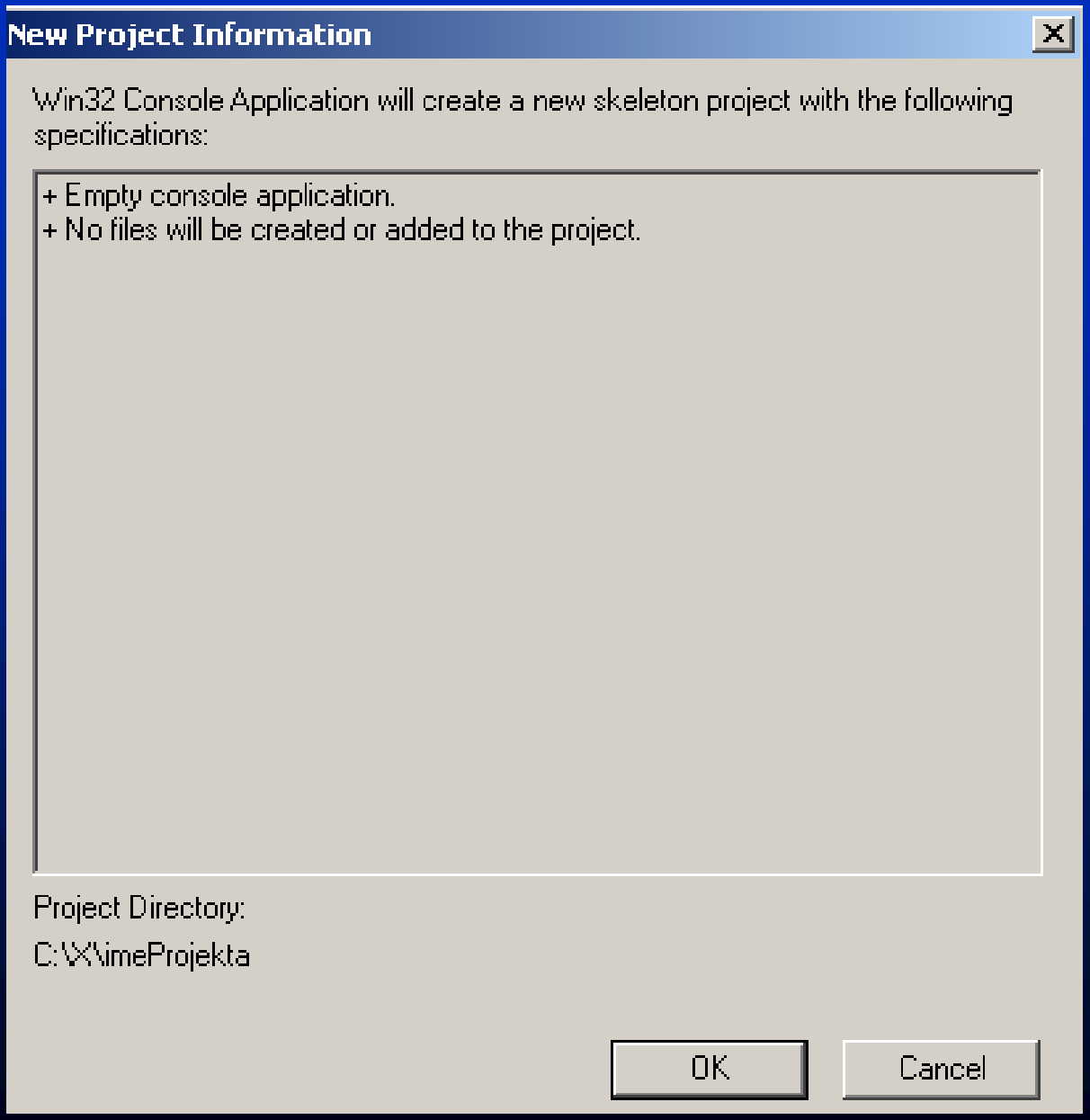
Развојно окружење



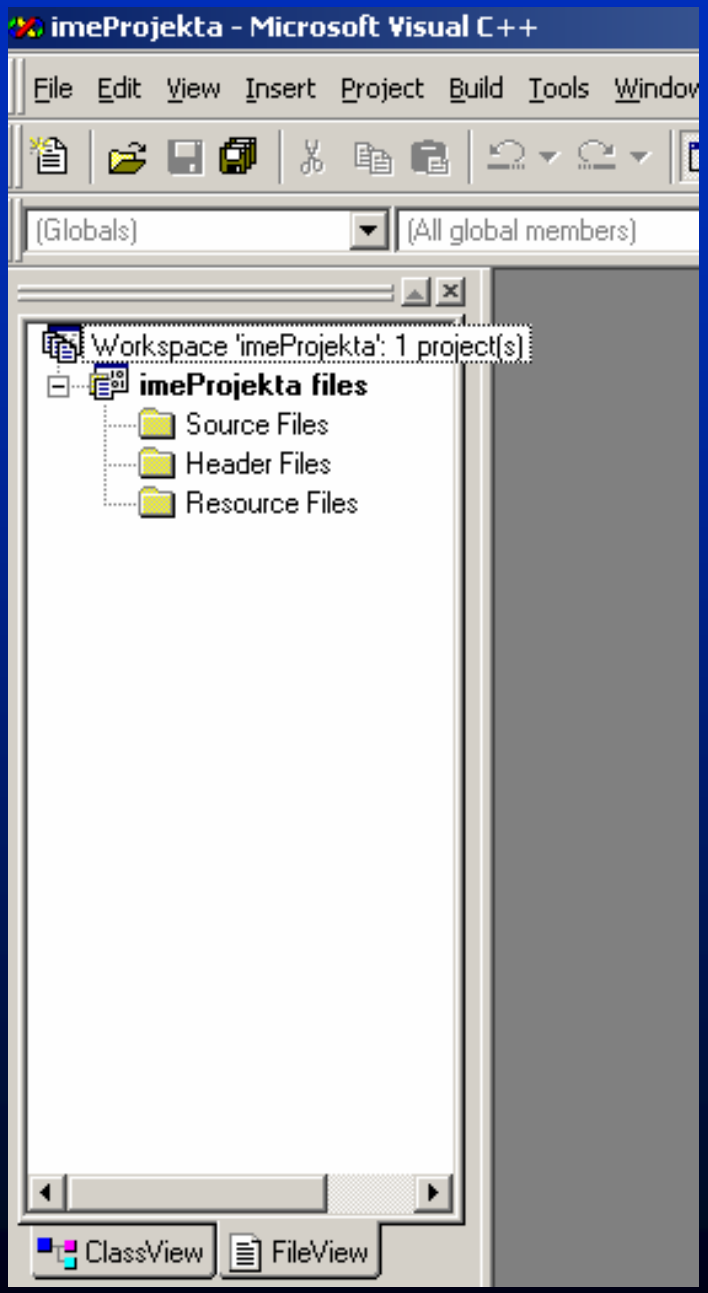
Развојно окружење



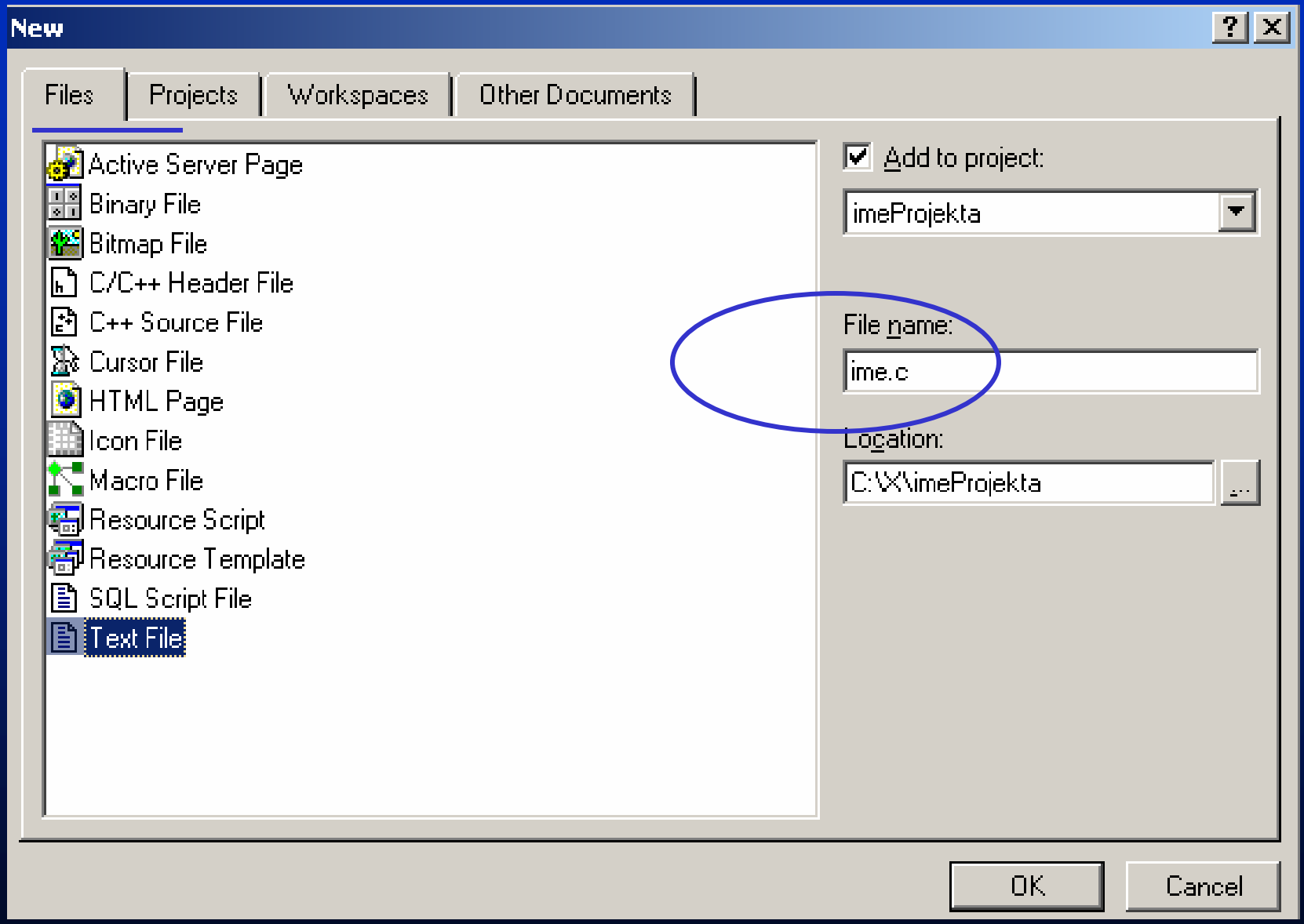
Развојно окружење



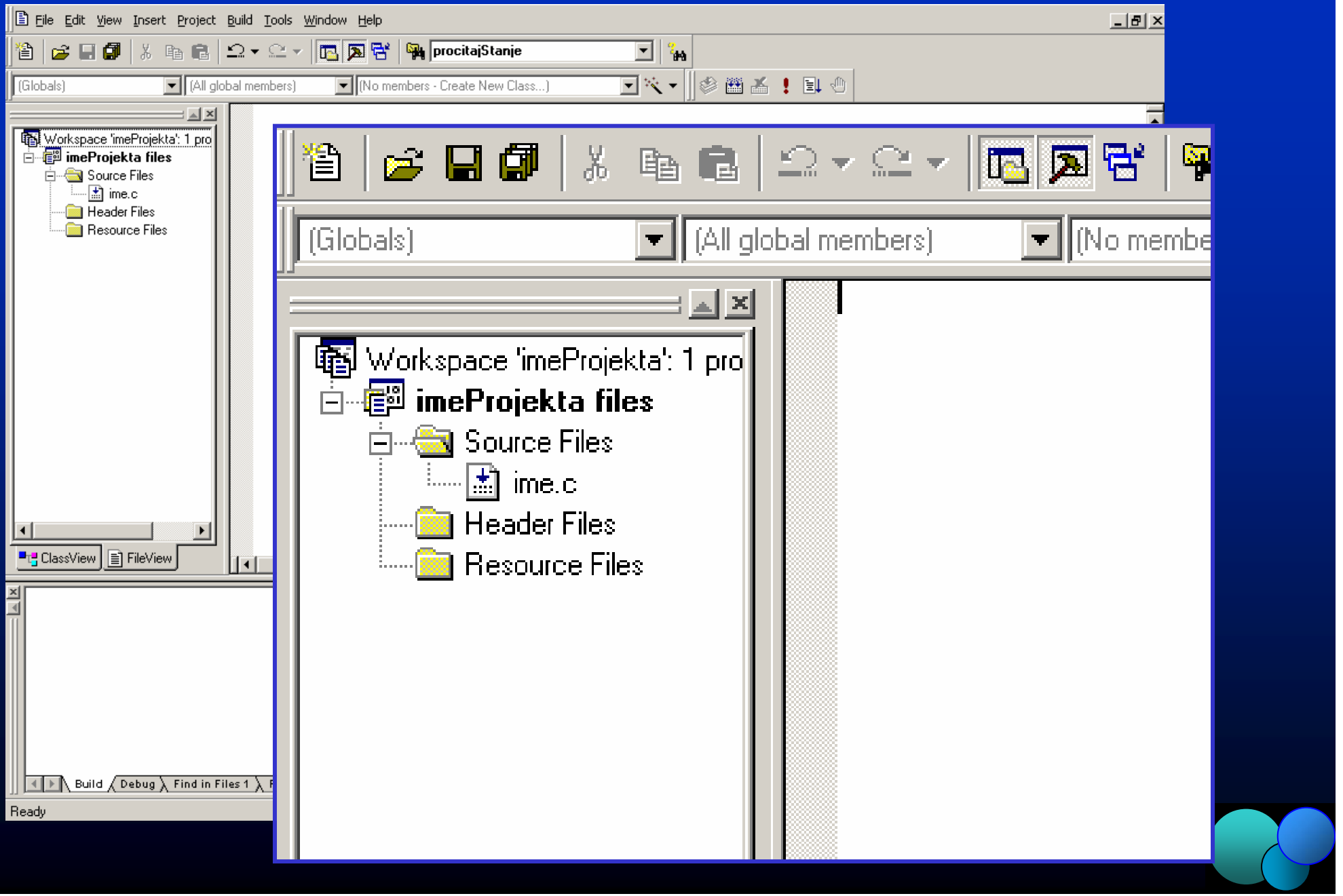
Развојно окружење



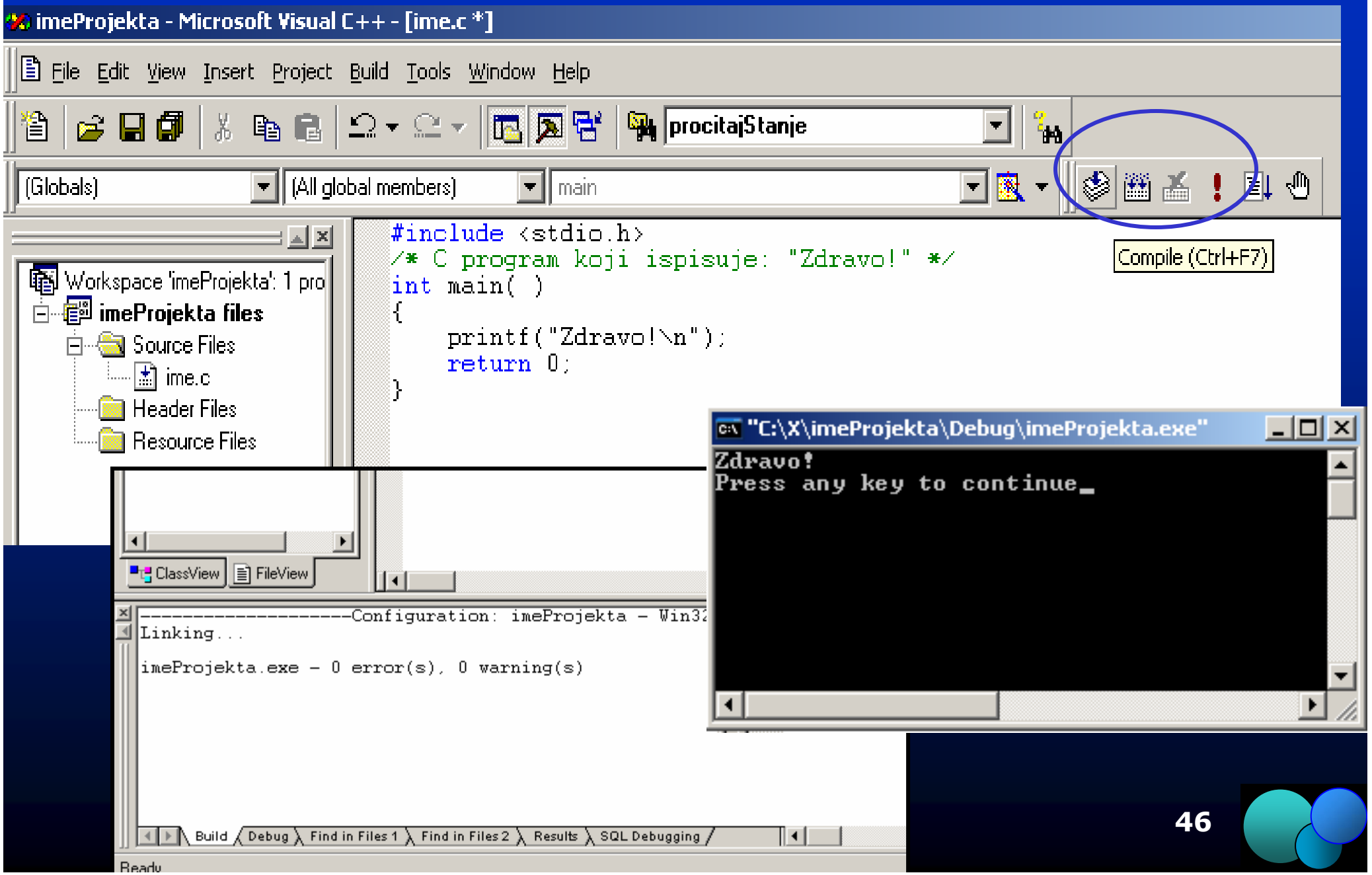
Развојно окружење



Развојно окружење



Развојно окружење



Хвала на пажњи

Питања?