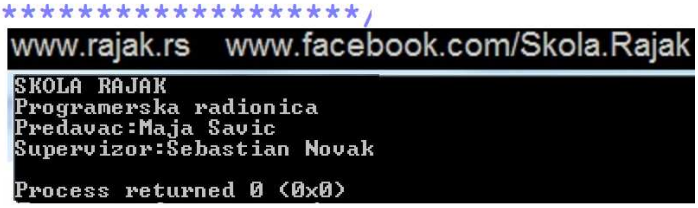
```c
/*********************************************
Skola matematike i racunara Rajak
www.rajak.rs
*********************************************/
#include <stdio.h>
#define p "Maja Savic"
#define s "Sebastian Novak"
int main(){
    printf("SKOLA RAJAK \n"
           "Programerska radionica \n"
           "Predavac:%s \n"
           "Supervizor:%s \n", p, s );
    return 0;
}
```



Садржај

Увод у програмирање кроз Ц језик

Кратка историја Ц-а:

Под именицом програмски језик се подразумева било који вештачки језик којим је могуће комуницирати и давати детаљније инструкције некој машини (рачунару).

Програмски језик Ц је развијен у времену од 1969. до 1972. године у истраживачком центру познатом као "Белова лабораторија" у САД-у од стране Дениса Ричија (енгл. *Dennis Ritchie*). Првобитно је представљао софтверско оружије за развијање оперативног система ЈУНИКС(енгл. UNIX). Овај оперативни систем је данас стандардни ОС за 16-тно битне рачунаре. Једна од најпознатијих варијанти ЈУНИКСА је ЛИНУКС (енгл. LINUX) за PC .

Продором ОС ЈУНИКС, растао је и значај програмског језика Ц, што је и природно ако се зна да је 90% ЈУНИКСА написано у Ц-у и да је већина корисничких програма за ЈУНИКС такође написано у Ц-у.

Елементи језика Ц:

Основне елементе језика Ц чине :

- Скуп знакова
  Скуп знакова у Ц-у образују :
  - Мала и велика слова енглеске абецеде
  - Децималне цифре
  - Специјални знаци
  - Невидљиви (бели)знаци

- Кључне (резервисане) речи
  Имају унапред дефинисано значење и не могу се користити у другом контексту.
  Примери кључних речи:
  Double , else , break , case , char , else , long , while , void , sizeof , goto , if , do , int..
  (У даљем тексту ћемо објаснити улогу сваке од ових речи.)

- Идентификатори
  Су низови знакова који означавају један објекат , функцију или друге различите елементе Ц језика.

  Структура Ц програма изгледа овако:
  Предпроцесорска директива

```
   Main()
{
      naredbe
   }
```

Развој програма:

Развој програма не почиње одмах са писањем програмског кода , већ у себи обухвата следеће кораке:

1. **Дефинисање циљева програма**

- У овом кораку се дефинише шта корисник тражи од програма за излаз и уноси податке које требају да буду обрађени.

2. **Пројектовање програма**

- Формирање алгоритма

3. **Писање програмског кода**

- Представља процес кодирања који се реализује у неком едитору или интегрисаном развојном окружењу (енгл. Integrated Development Environment, IDE).

4. **Извршавање програма**

- Одабира се програмски језик и извршава се превођење на машински језик. Преводиоци врше превођење тек након завршеног уноса програмског кода.

5. **Превођење и повезивање програма**
6. **Тестирање и отклањање грешака**
7. **Одржавање и модификовање програма**

Вероватно вам до сад ништа није било јасно и питате се какве ово има везе са програмирањем. Да бих вам приближила све ово , мораћемо сву ову сувопарну теорију да применимо практично на примерима

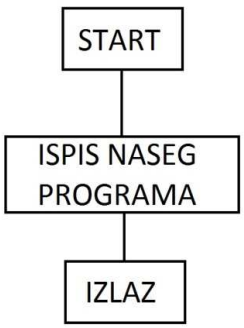
Да се упознамо, ја сам Ц језик...

Пример 1 : Упознавање са програмом.

У овом примеру, наш задатак је да напишемо програм у Ц-у који ће нам исписати на екрану текст : **Здраво ,ја сам Ц (**латиницом, наравно**).**

Овај проблем ћемо решити поступно, пратећи кораке из поглавља “РАЗВОЈ ПРОГРАМА”.

1. На листи је: ***Дефинисање циљева***:
   Први корак ка решавању задатог проблема је тачно дефинисање циљева , од којих података желимо да добијемо резултат. Наш циљ у овом примеру је испис поруке на екрану и у овом случају неће бити улазних параметара.
2. Корак је : ***Пројектовање програма***:
   Кад смо јасно дефинисали циљ програма , сад иде редослед радњи. Приказаћемо то једноставним алгоритмом где имамо **почетак,испис и крај.** Изгледа једноставно зар не?



3. Корак је : ***Писање програма***:

У овом кораку почиње да буде занимљивије и овај корак може да буде најтежи за оне који се први пут сусрећу са неким програмским језиком.

Програм се пише по унапред одређеним правилима.

Програм мора да садржи:

a) **Предпроцесорске наредбе**
b) **Главну функцију main();**
c) **Декларацију променљивих** које ће нам требати у решавању проблема(улазни параметри)
d) **Испис тражених вредности**
(излазни параметри)

… што би у нашем примеру изгледало овако:

```
#include<stdio.h>
int main()
{
    printf("Zdravo, ja sam C.");
}
```

У овом примеру прескачемо прву тезу(јер немамо улазних параметара). За излаз желимо текст на екрану, зато позивамо функцију ***printf(" * * * * * ");*** где ****** представљају жељени текст који очекујемо на екрану.

Да би могли да радимо са њом(касније и осталим функцијама) морамо прво да напишемо где се она налази.

#include< * * * * > *користимо за такве ствари*. ***** су имена "библиотека" где се оне налазе.

Стандардна библиотека ***stdio.h*** (где је .h ознака за библиотеку и настала је од речи **header** , а **stdio** је скраћеница од ***STANDARD INPUT OUTPUT***) садржи све изласно–улазне функције у коју се убраја и наша printf.

Поред printf()-а, имамо и другу функцију која се зове **main(). *Main()*** је индикација за почетак извршавања програма. Опширније о њој и зашто смо ставили **int main ()** ћемо причати касније, за сад ћемо то узети здраво за готово.

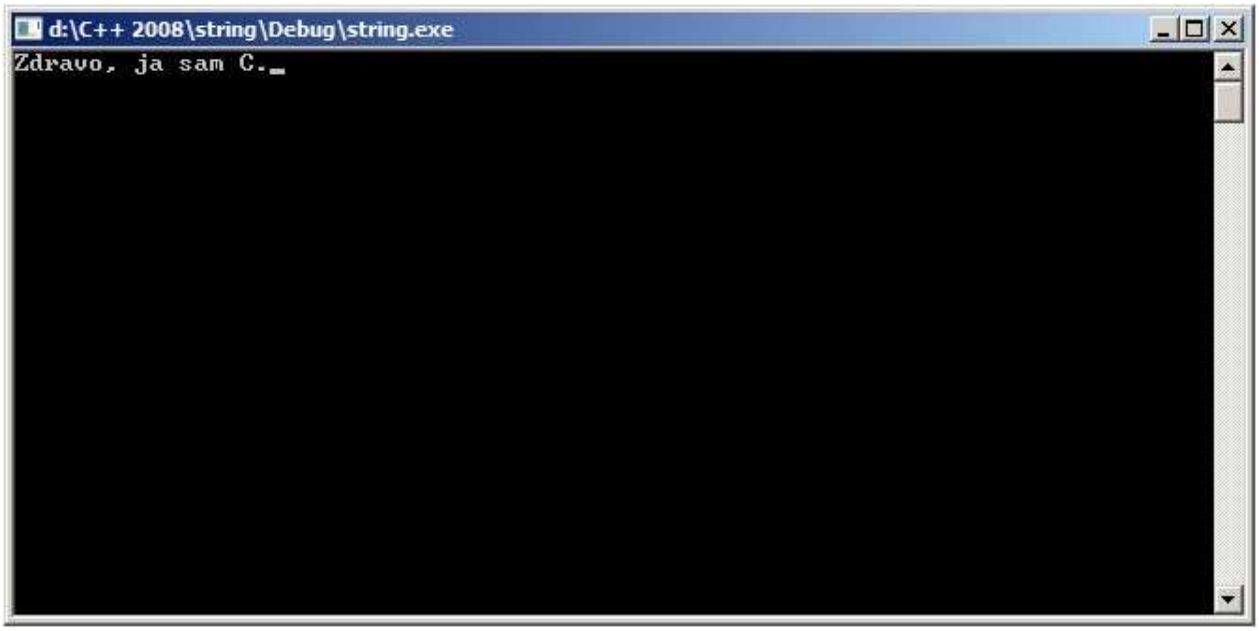
Када смо искодовали наш програм време је да га тестирамо.

Кад смо покренули програм, видели би смо да се на екрану само на моменат приказала порука и програм се угасио.То се догодило јер смо искуцали код само за приказ поруке, и чим се та наредба изврши, програм се угаси. Да се то не би дешавало, морамо да засуставимо слику на екрану, зато код допуњавамо са 2 нове линије.

```
#include<stdio.h>
#include<conio.h>
int main()
{
    printf("Zdravo, ja sam C.");
    getch();
}
```

Прва промена је извршена код предпроцесорских директива додавајући још једну библиотеку, која се зове **conio.h** (**.h** долази од hedera,а **conio** је скраћеница од **CONSOLE INPUT OUTPUT** која је за разлику од stdio нестандардна библиотека) у којој се налази функција која се зове **getch().** Ова функција замрзава слику на екрану и држи је док корисник не притисне неко дугме на тастатури.

Затим поново покрећемо програм....



Овако би требало да изгледа решење нашег проблема.