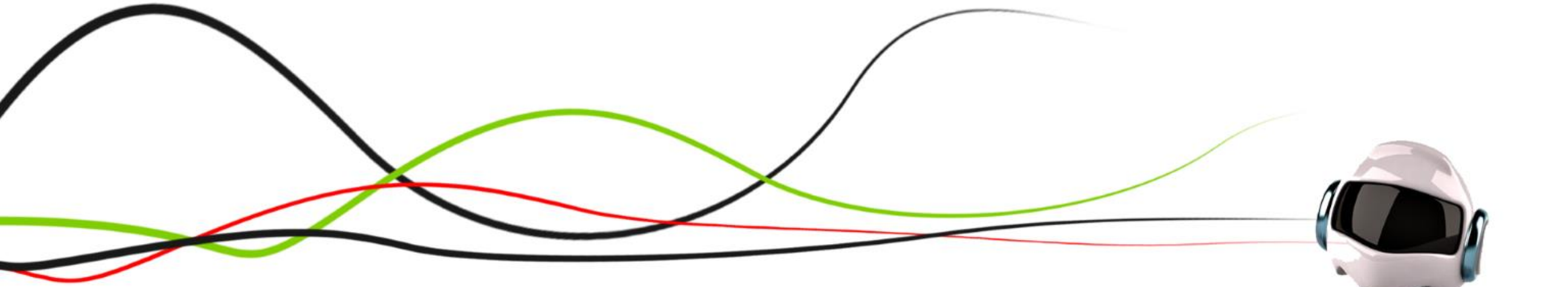
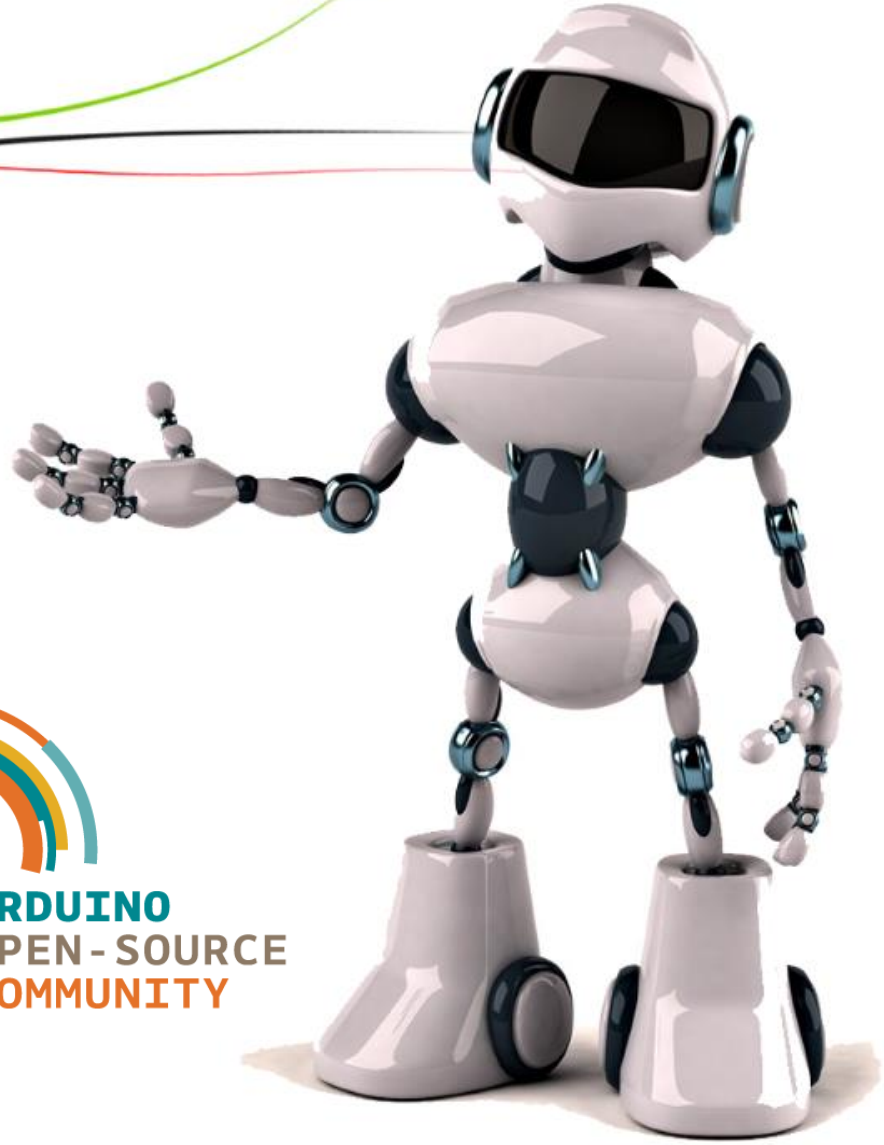


مکتب خونه
آکادمی آنلاین تخصص ها



دوره آموزشی آردوینو

فصل دوم

بخش پنجم محیط برنامه نویسی ، برنامه نویسی اولین برنامه

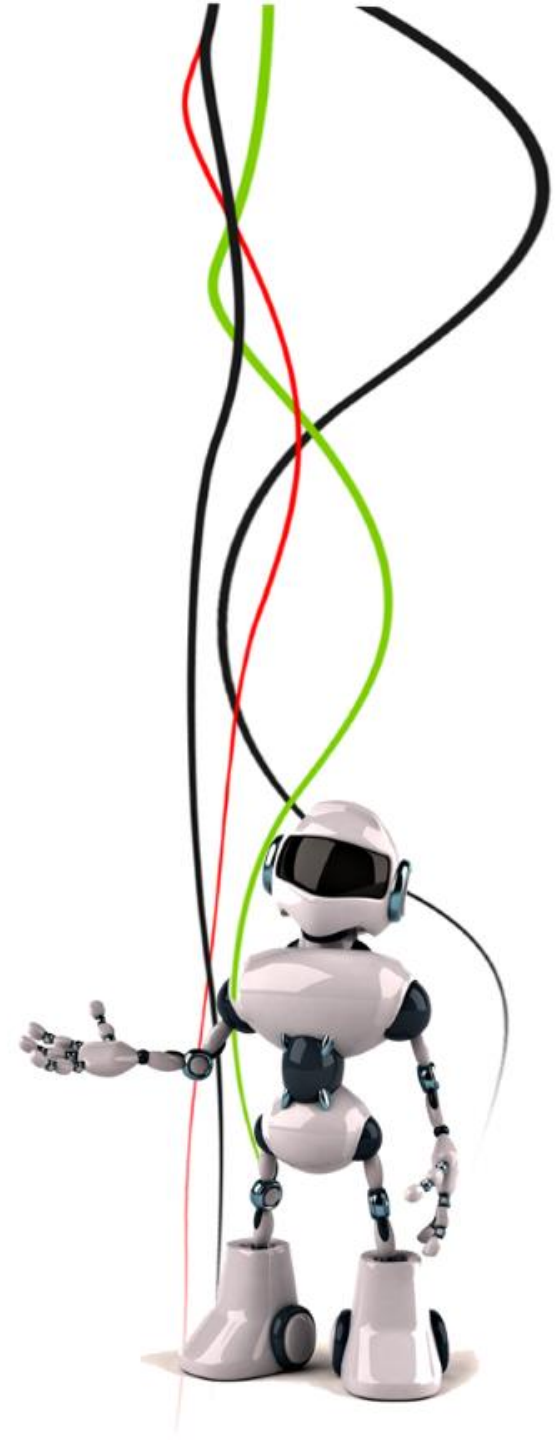
مدرس : محمد پارسا کریمی

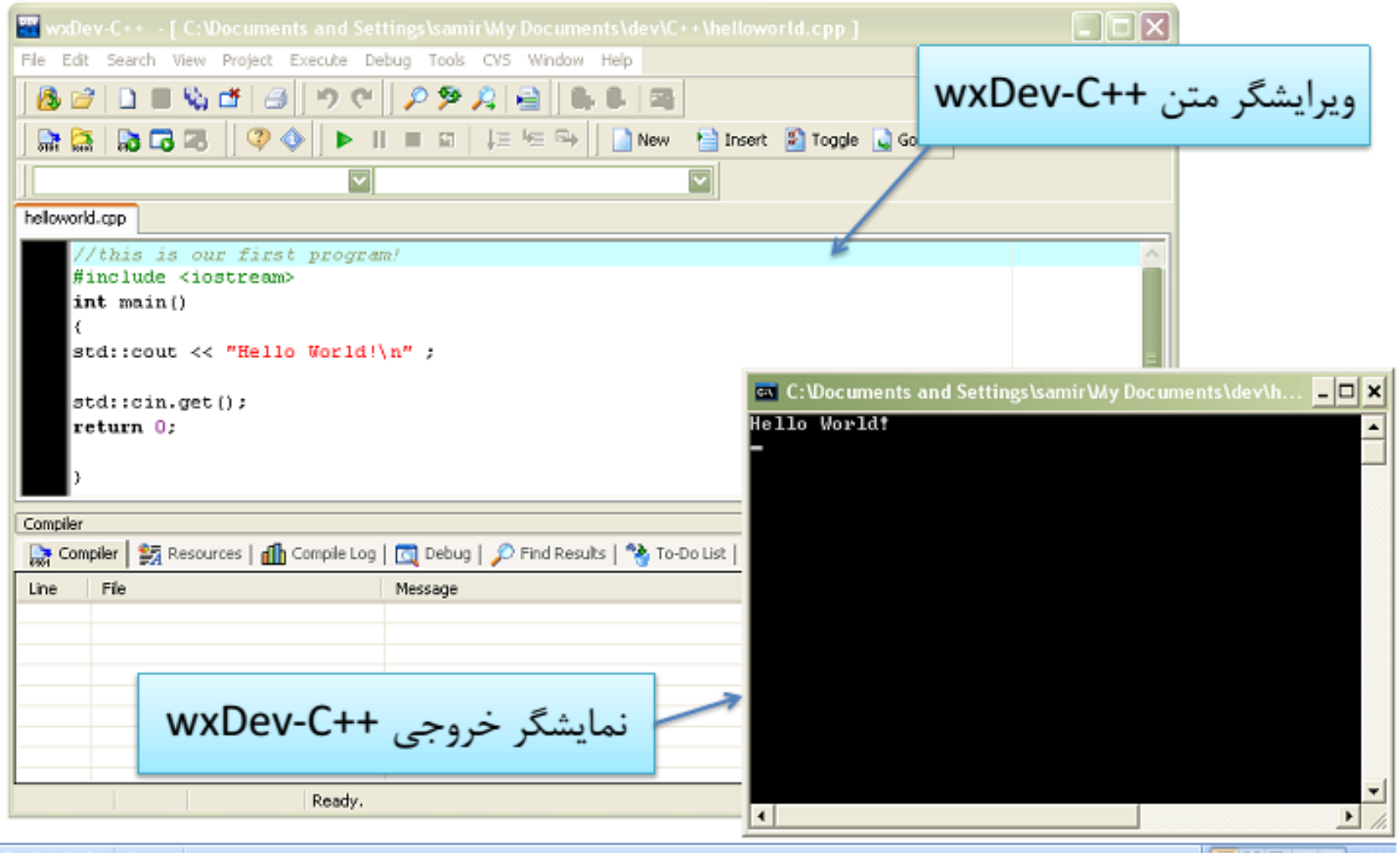
ARDUINO
OPEN-SOURCE
COMMUNITY

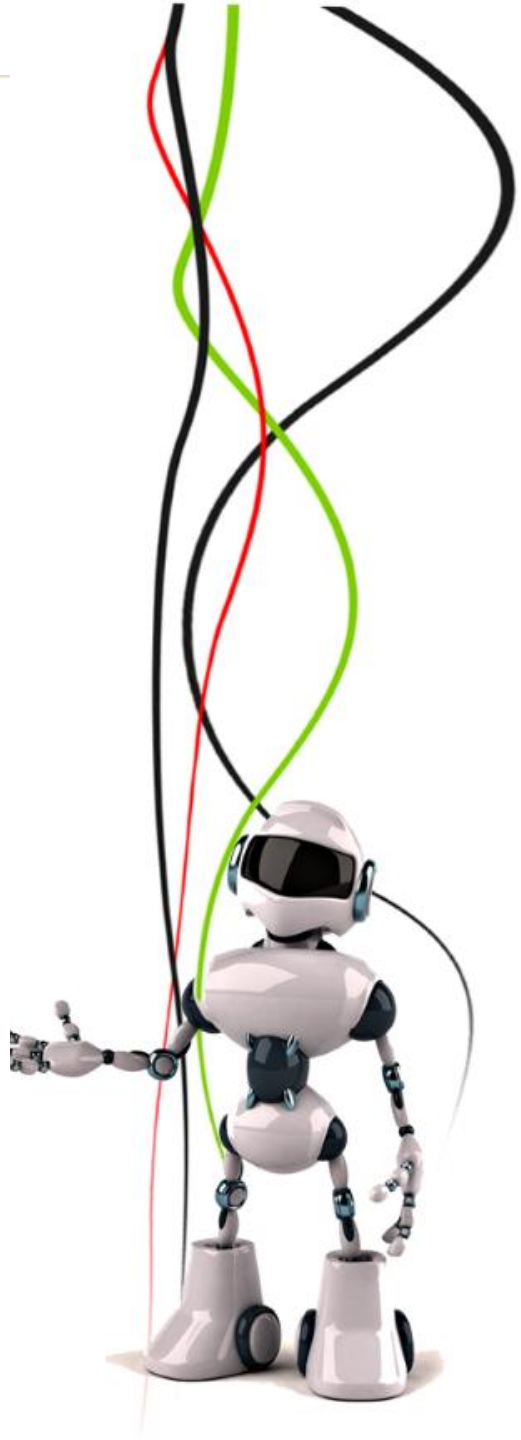
محیط برنامه‌نویسی (IDE)

- اکثر تولید کنندگان کامپایلرها، محیط‌هایی را برای برنامه نویسی ارائه کرده اند که کلیه مراحل لازم جهت تبدیل کد مبدا به زبان ماشین را به‌صورت خودکار انجام می‌دهند.
- به این محیط‌ها IDE (Integrated Development Environment) یا محیط مجتمع توسعه نرم افزار گفته می شود.
- انواع IDEها با ویژگی‌های مختلف برای برنامه‌نویسی به زبان C++ توسط شرکت‌های مختلف ارائه شده است مانند:
  - Turbo C++
  - Borland C++
  - Visual C++
  - و غیره

در این دوره استفاده از نرم‌افزار wxDev-C++ پیشنهاد می‌شود.




wxDev-C++ - [C:\Documents and Settings\samir\My Documents\dev\C++\helloworld.cpp]
File Edit Search View Project Execute Debug Tools CVS Window Help
New Insert Toggle
helloworld.cpp
//this is our first program!
#include <iostream>
int main()
{
std::cout << "Hello World!\n" ;
std::cin.get();
return 0;
}
Compiler
Compiler Resources Compile Log Debug Find Results To-Do List
Line File Message
Ready.
C:\Documents and Settings\samir\My Documents\dev\h...
Hello World!
ویرایشگر متن wxDev-C++
نمایشگر خروجی wxDev-C++



ساختار برنامه‌ی C++

- برنامه‌های زبان C++ از مجموعه‌ای از دستورات و تعدادی تابع تشکیل شده‌اند.
- بدنه‌ی اصلی برنامه، تابعی به نام main() است.

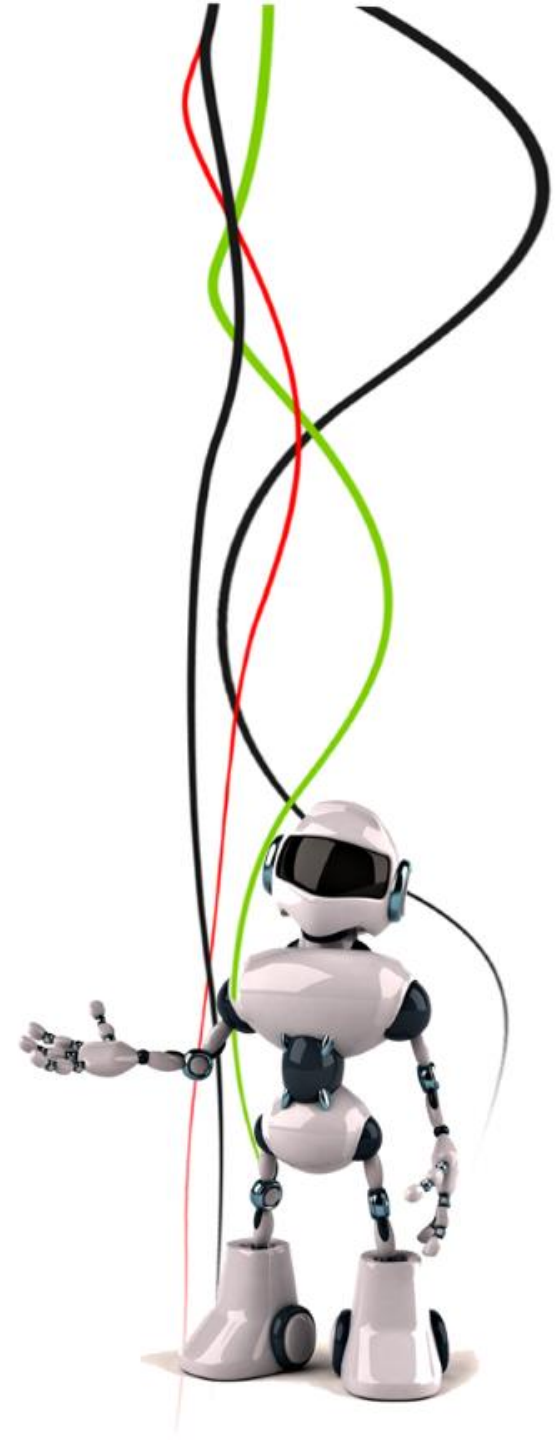
```
# include <فایل سرآیند>
int  main()
{
        اعلان متغیرها

        دستورات اجرایی

return 0;
}
```

هیچ برنامه‌ای وجود ندارد که فاقد این تابع باشد.

- علاوه بر تابع main()، توابع دیگری برای انجام عملیات مختلف از قبل نوشته شده و همراه کامپایلر ارائه می‌شوند.
- این توابع و سایر اطلاعاتی که کامپایلر برای ترجمه به آن‌ها نیاز دارد در فایل‌هایی به عنوان فایل‌های سرآیند (header file) قرار دارند.

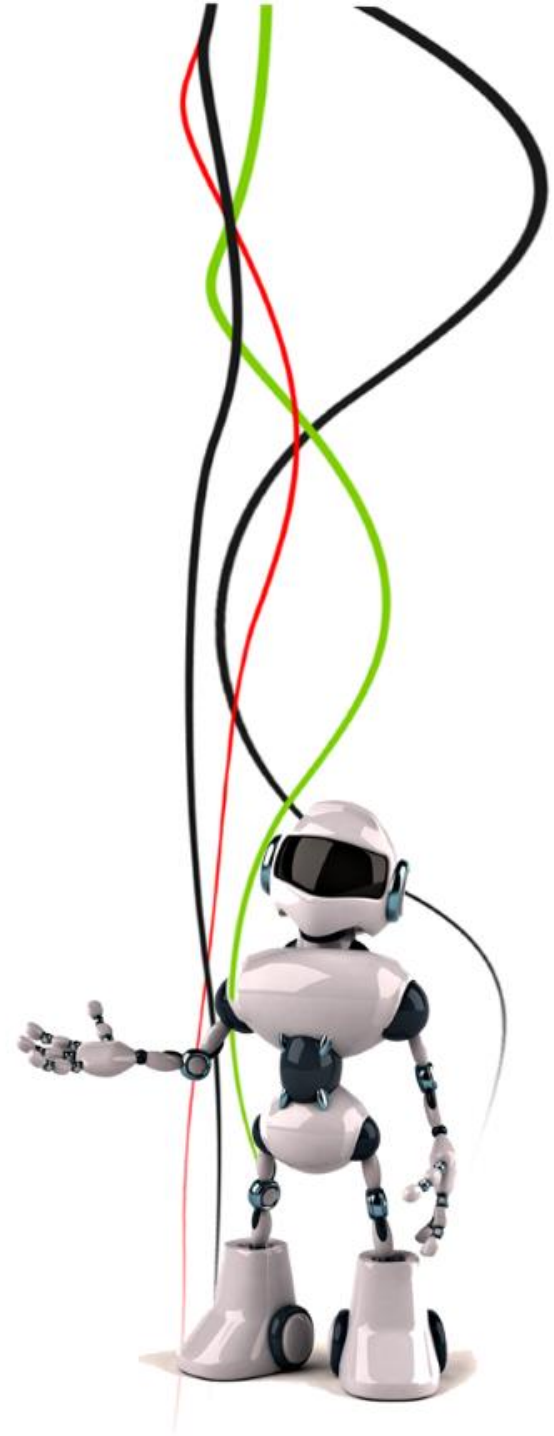


اولین برنامه به زبان C++

- اولین برنامه‌ای که می‌نویسیم به محض تولد، به دنیا سلام می‌کند!

```
//this is our first program!
#include <iostream>
int main()
{
std::cout << "Hello World!\n" ;
return 0;
}
```

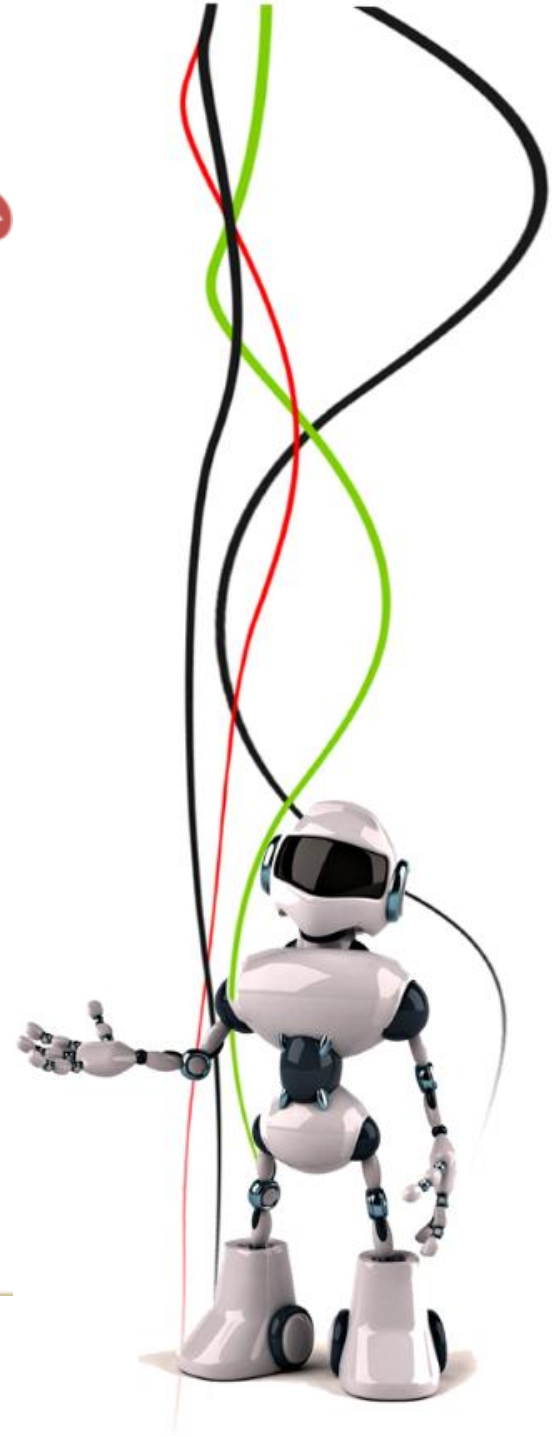
یک عادت خوب برنامه نویسی:
هر برنامه بهتر است با یک توضیح آغاز شود که هدف برنامه، نویسنده برنامه و تاریخ را مشخص کند.



دستورات پیش‌پردازنده

- پیش‌پردازنده را می‌توان برنامه‌ی جداگانه‌ای در نظر گرفت که قبل از کامپایلر واقعی اجرا می‌گردد.
- تمامی فرامین پیش‌پردازنده با ’#‘ آغاز می‌گردند.
- برخلاف دستورات C++ که به ; ختم می‌شوند، پایان جمله‌های پیش پردازنده با خط جدید مشخص می‌شود.

```
//this is our first program!
#include <iostream>
int main()
{
std::cout << "Hello World!\n" ;
return 0;
}
```

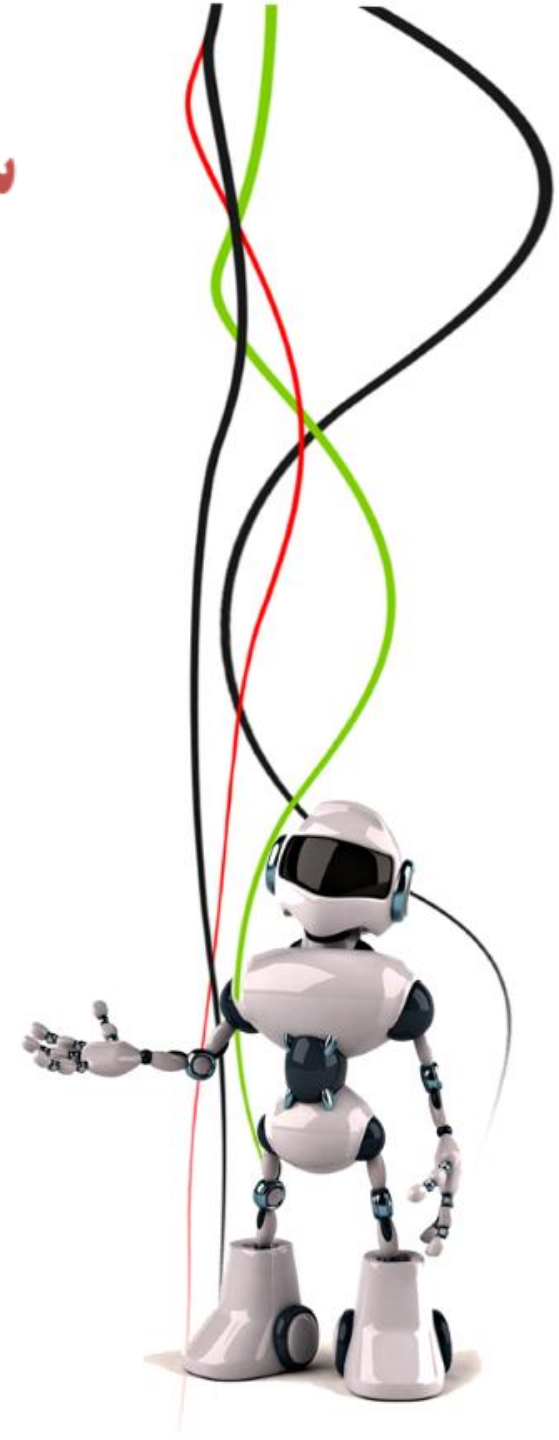


ساختار برنامه‌ی C++

- برای استفاده از توابع آماده‌ی زبان C++، باید بدانیم هر تابع درکدام فایل سرآیند وجود دارد و همان را به برنامه اضافه کنیم.
- برای اتصال فایل‌های سرآیند از دستوری به نام #include استفاده می‌شود. (که یک دستور پیش‌پردازنده است)

```
//this is our first program!
#include <iostream>
int main()
{
std::cout << "Hello World!\n" ;
return 0;
}
```

توابعی که برای ورود و خروج داده‌ها استفاده می‌شوند در سر فایل iostream قرار دارند.

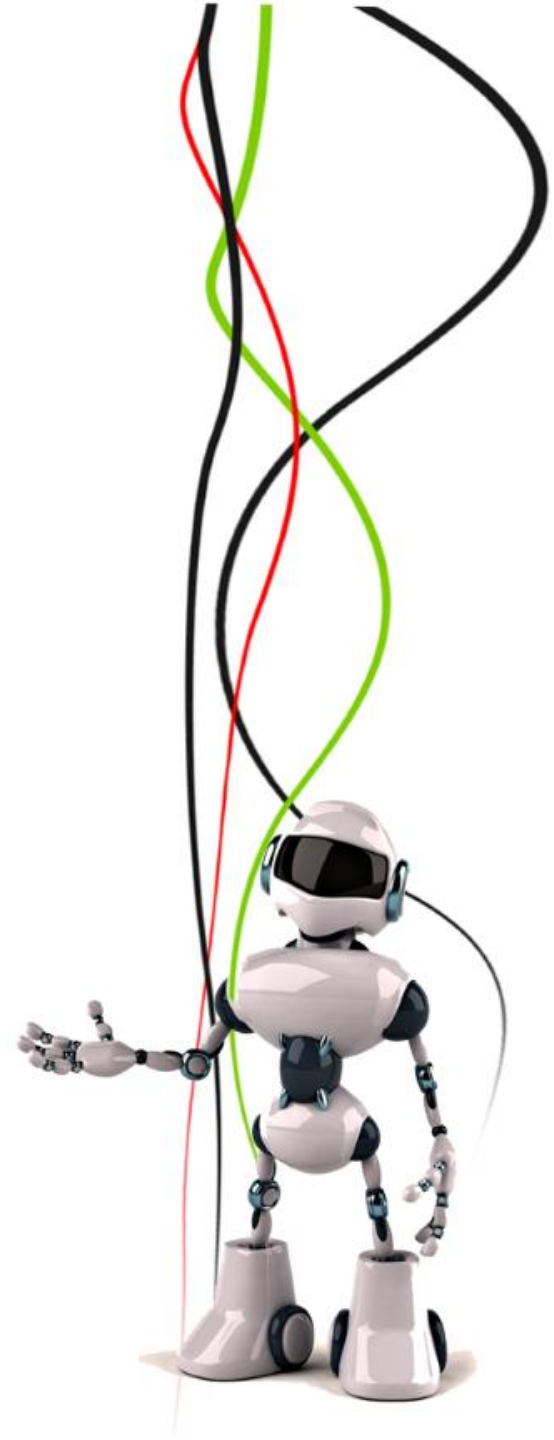


نکات دستور #include

- دستور #include قبل از تابع main و به شکل زیر به کار می‌رود.

include <فایل سرآیند> ⟵ # include <iostream>

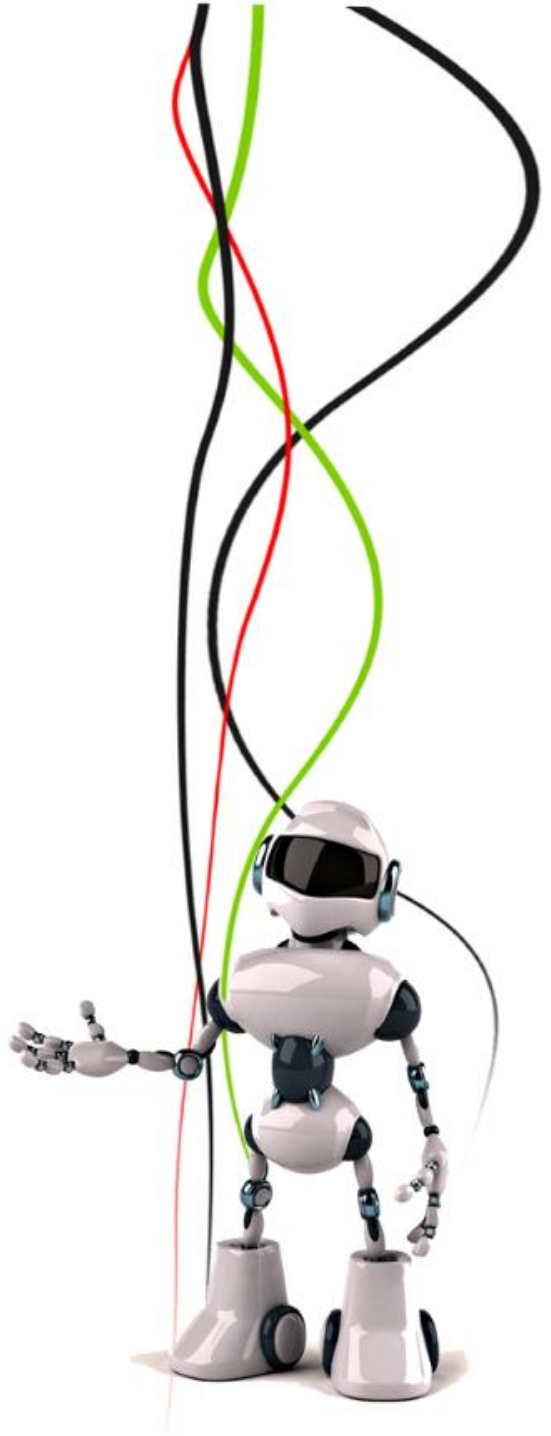
- برای اضافه کردن چند فایل سرآیند، برای هریک باید یک دستور #include جداگانه نوشت.
- بین # و include نباید فاصله‌ای وجود داشته باشد.
- بین نام فایل و علائم < و > نباید فاصله‌ای باشد.
- ذکر علائم < و > ضروری است.



دستور int main()

- این دستور در تمامی برنامه‌های C++ وجود دارد.
- پرانتزهای پس از main تعیین می‌کنند که این دستور یک قسمت بسیار مهم از برنامه به نام تابع است.
- یک برنامه‌ی C++ حاوی یک یا تعداد بیشتری تابع است. (وجود تابع main ضروری است)
- اجرای برنامه‌های C++ در تابع main آغاز می‌گردد، حتی اگر این تابع نخستین تابع موجود در برنامه نباشد.
- لغت کلیدی int در سمت چپ main مشخص می‌کند که main یک مقدار int را برمی‌گرداند.
- آغازگر بدنه‌ی هر تابع آکولاد باز و انتهای تابع آکولاد بسته می‌باشد.

```
//this is our first program!
#include <iostream>
int main()
{
std::cout << "Hello World!\n" ;
return 0;
}
```



دستور چاپ

- کل دستورخط ۵ برای چاپ جمله‌ای در خروجی استفاده می‌شود.

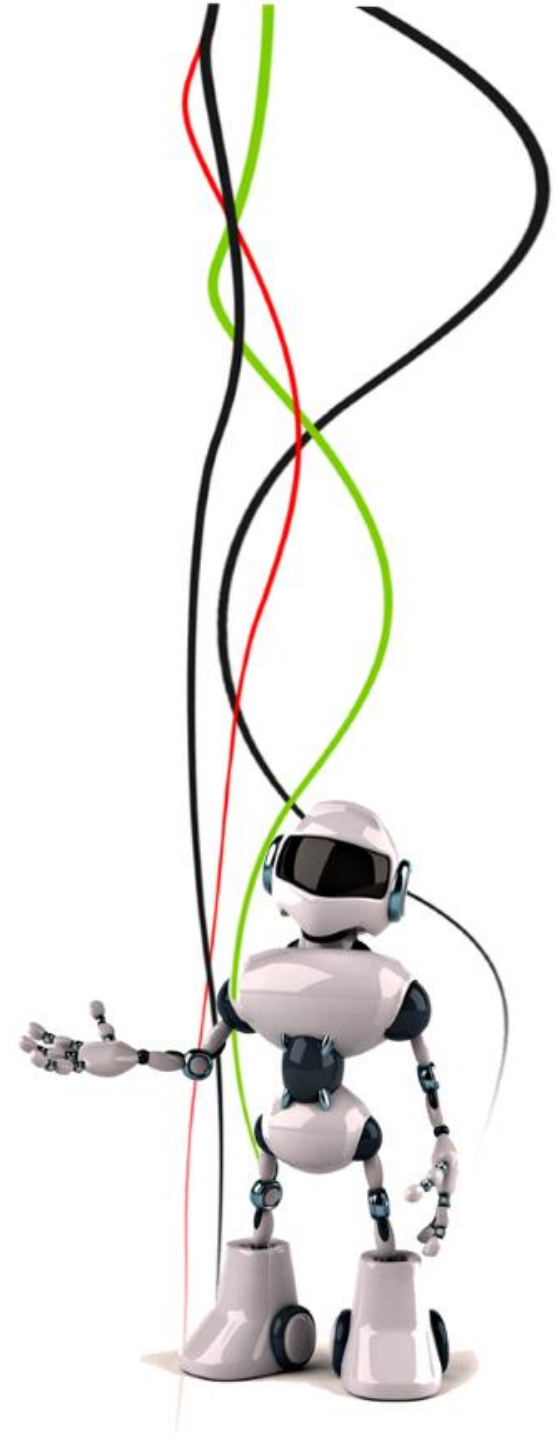
یک کاراکترکنترلی یا رشته‌ی گریز

```
std::cout << "Hello World!\n" ;
```

عملگر درج در جریان (خروجی)

رشته کاراکتر

- این دستور رشته کاراکتر مورد نظر را به فرآیند خروجی std::cout میفرستد که معمولا این خروجی صفحه نمایش میباشد.
- کاراکتر \n در خروجی چاپ نخواهد شد بلکه وجود این کاراکتر باعث میشود مکان‌نما به خط بعدی صفحه نمایش پرش نماید.
- وجود std:: برای دستوراتی که توسط پیش‌پردازنده‌ی <iostream> وارد برنامه می‌شوند الزامیست.

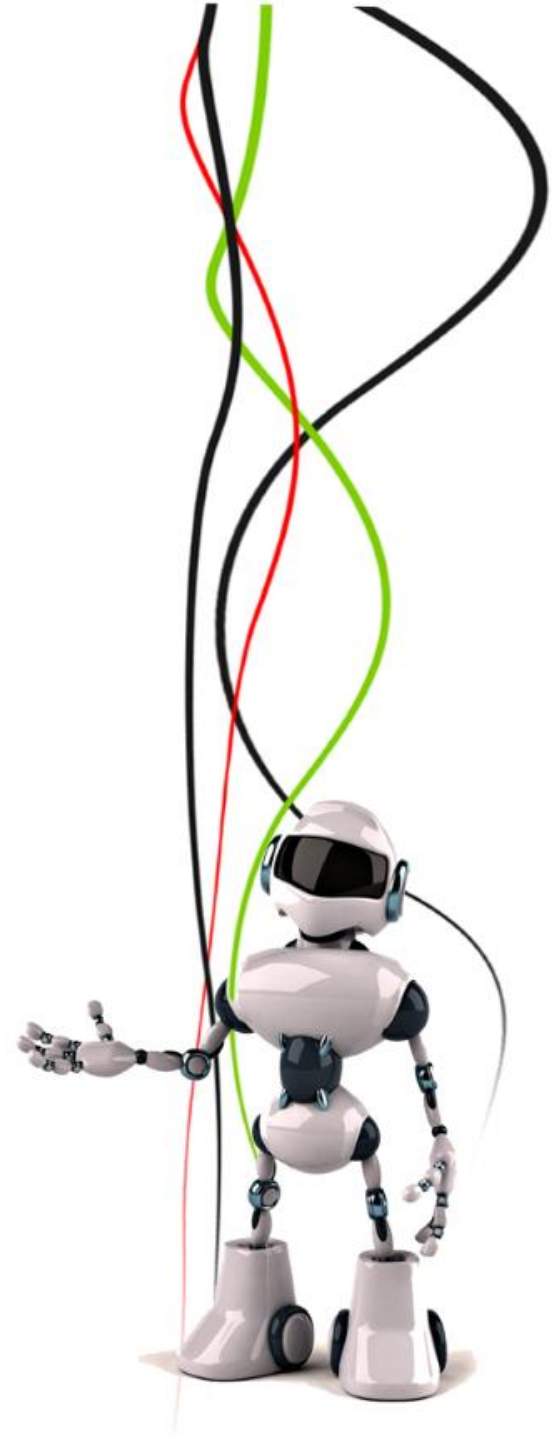


دستور return 0

- سیستم عامل که اجرا کننده‌ی تابع main() است می‌خواهد بداند آیا این تابع با موفقیت به پایان رسیده‌است یا خیر.
- برای این منظور بهتر است در پایان اجرای برنامه، مقداری به سیستم عامل برگردانده شود.
- آخرین دستور برنامه، return 0، مقدار 0 را برمی‌گرداند.
- این روش یکی از راه‌های خروج از یک تابع می‌باشد.

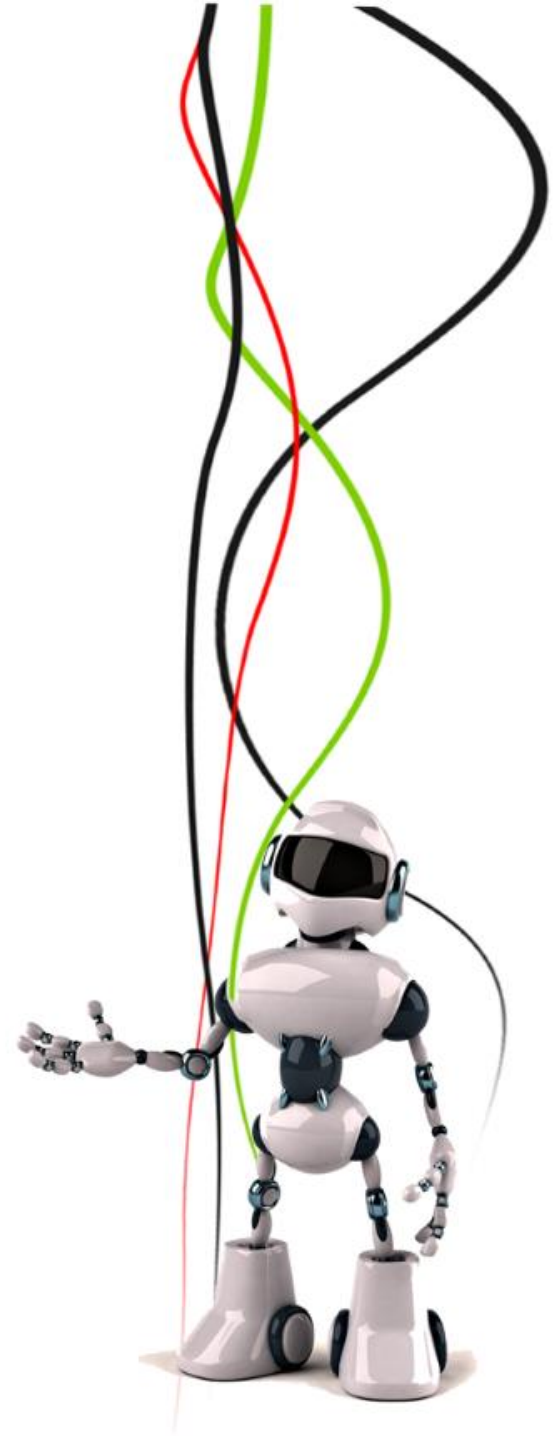
```
//this is our first program!
#include <iostream>
int main()
{
std::cout << "Hello World!\n" ;
return 0;
}
```

نکته مهم: حتما باید در پایان هر دستور، علامت سمیکولن ; قرار دهید



استفاده از فضای نام (namespace)

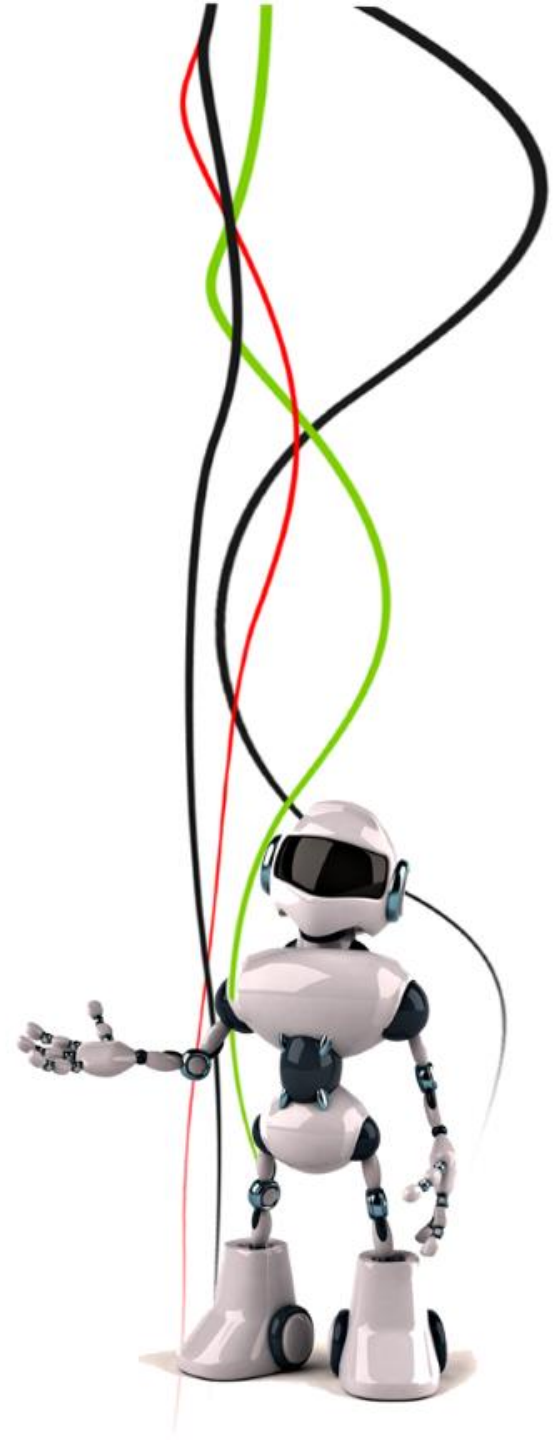
```
#include <iostream>
using namespace std;
int main()
{
cout << "Hello  World!\n" ;
return 0;
}
```



کاراکترهای کنترلی (رشته‌های گریز)

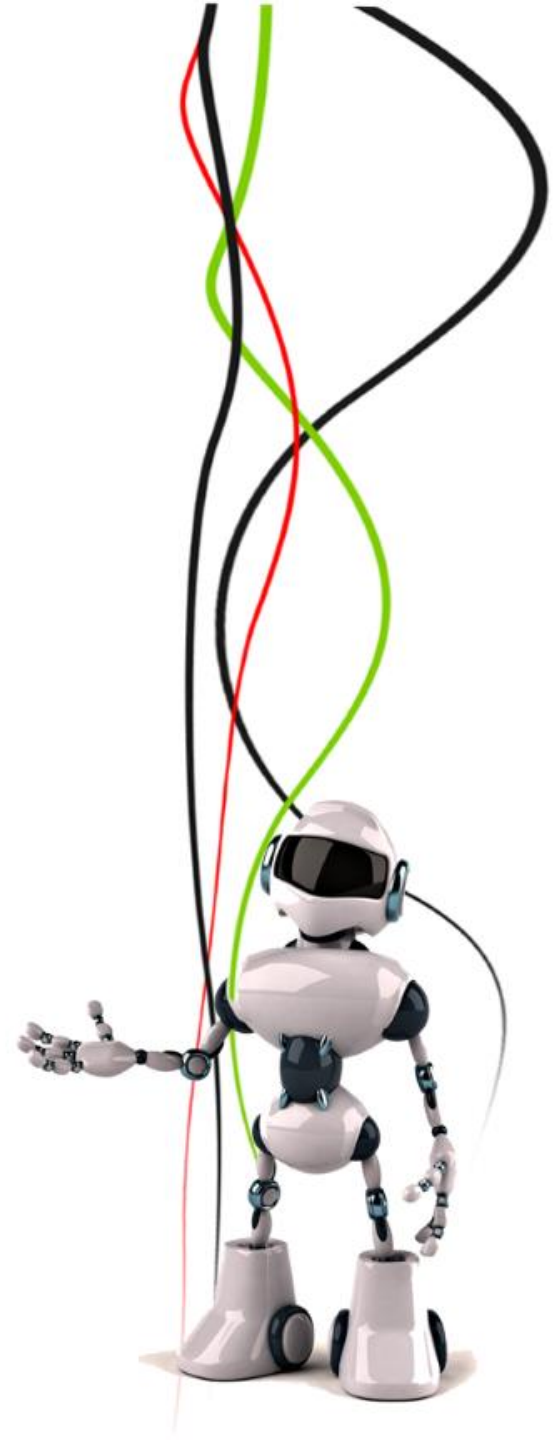
- این کاراکترها شکل خروجی اطلاعات را مشخص می‌کنند.
  - آیا تمام اطلاعات در یک سطر چاپ شوند؟
  - آیا اطلاعات با فاصله‌ی خاصی چاپ شوند؟

- برای مثال \n موجب می‌شود تا از سطر جاری صرفنظر شده و ادامه‌ی اطلاعات در ابتدای سطر بعدی چاپ شوند.



کاراکترهای کنترلی (رشته‌های گریز)

رشته‌های گریز	توضیح
\n	**سطر جدید.** مکان نمای صفحه نمایش را در ابتدای سطر بعد قرار می‌دهد.
\t	**Tab افقی.** مکان نما را به یک tab جلوتر انتقال می‌دهد.
\r	**بازگشت به ابتدای سطر.** مکان نما را به ابتدای سطر جاری برمی‌گرداند.
\a	**هشدار.** بوق سیستم را به صدا در می‌آورد.
\\	**ممیز برعکس.** برای چاپ ممیز برعکس کاربرد دارد.
\’	**چاپ کوتیشن تکی.**
\”	**چاپ کوتیشن دوتایی.**



چاپ یک خط متن با استفاده از چندین دستور

```
#include <iostream>
using namespace std;
int main()
{
cout << "Hello ";
cout<< "World!\n" ;
return 0;
} // end of function main
```

هر دستور درج در جریان، به طور پیش‌فرض چاپ خروجی را از ادامه‌ی خروجی قبلی آغاز می‌کند.

خروجي

```
Hello World!
```

تا زمانی که به طور صریح چاپ خروجی در خط جدید (با \n) اعلام نشود، دستورات چاپ در همان سطر قبل عمل خواهند کرد.