Въведение. Структура на програмата

Вероятно най-добрият начин да започнете да учите език за програмиране е от написването на програмата. Ето защо, тук е нашата първа програма:

```
1  // my first program in C++
2  #include <iostream>;
3  using namespace std;
4
5  int main()
6  {
7    cout << "Hello World!" ;
8    return 0;
9  }
10
```

```
Hello World!
```

Първият панел (в светло синьо) показва изходния код на нашата първа програма. Вторият (в светло сиво) показва резултата от програмата, след като съставен и се изпълнява. На ляво, на сивите цифри представляват номерата на редовете - те не са част от програмата, и са показани тук само за информационни цели.

Начинът, по който да редактирате и съставя програма зависи от компилатор, който използвате. В зависимост от това дали има развитие Interface или не и на неговата версия. Консултирайте се с компилатори точка и ръководство или помощ, включена с вашия компилатор, ако имате съмнения за това как да съставят C + + конзола програма.

Предишната програма е типичната програма, която програмист чираци пиша за първи път, и неговия резултат е печат на екрана на "Hello World!" изречение. Той е един от най-простите програми, които могат да бъдат дадени в C + +, но вече съдържа основните компоненти, които всеки C + + програма има. Ние ще разгледаме ред по ред в кода, който току-що са дадени:

```
// my first program in C++
```

Това е коментар на линия. Всички редове, които започват с два знака наклонена черта (`//`) се считат за коментари и

не оказва влияние върху поведението на програмата. Програмистът може да ги използвате, за да включват кратки обяснения или забележки в рамките на източник самия код. В този случай, на ред е кратко описание на това, което ни е програмата.

```
#include <iostream>
```

Редовете, които започват със знака диез (#) са директиви за Preprocessor. Те не са редовни линии код с изрази, но индикациите за Препроцесорът на компилатор. В този случай директивата `#include <iostream>` разказва Preprocessor да включват iostream стандартен файл. Този специфичен файл (iostream) включва декларациите на основен стандарт на входно-изходната библиотека в C++, и е включена, защото нейната функционалност ще бъдат използвани по-късно в програмата.

```
using namespace std;
```

Всички елементи на стандартната C++ библиотека са обявени в рамките на това, което се нарича пространство от имена, пространството от имена с името *STD.* Така че, за да получите достъп до неговата функционалност ние заявяваме с този израз, че ще използваме тези субекти. Тази линия е много често в C++ програми, които използват стандартната библиотека, а в действителност той ще бъде включен в повечето от изходните кодове, включени в тези уроци.

```
int main ()
```

Тази линия съответства на началото на определянето на основната функция. Основната функция е точката, от която всички C + + програми започне тяхното изпълнение, независимо от местоположението си в рамките на програмния код. Няма значение дали има други функции с други имена, определени преди или след него - указанията, съдържащи се в дефиницията на тази функция винаги ще бъде първите, които трябва да бъдат изпълнени във всички C++ програма. По същата причина, от съществено значение е, че всички C + + програми имат *основна* функция.

Думата `main` е последвано в кодекса от чифт скоби (`()`). Това е така, защото тя е функция декларация: В C++, което отличава функция декларация от другите видове изрази са тези скоби, които следват неговото име. По желание, тези скоби могат да приложат списък на параметрите в тях.

Веднага след тези скоби можем да открием тялото на основната функция поставени във фигурни скоби (`{}`). Това, което се съдържа в тези скоби е това, което прави функцията, когато се изпълнява.

```
cout << "Hello World!";
```

Тази линия е C + + изявление. В изявление е прост или съставен израз, който може действително да произвежда някакъв ефект. В действителност, това твърдение извършва само действия, които генерира видим ефект в първата ни програма.

`cout` е името на стандартния изходен поток в C++, и смисъла на цялата декларация е да въведете поредица от символи (в този случай `Hello World` поредица от символи) в стандартния изходен поток *(**cout**,* която обикновено съответства на екрана).

Cout е обявен в стандартния файл *iostream* в рамките на пространство от имена *STD,* така че защо ни трябваше да се включи, че определен файл и да обяви, че щяхме да се използва този конкретен Именно пространство по-рано в нашия код.

Забележете, че изявлението завършва с точка и запетая характер (`;`). Този символ се използва, за да отбележи края на изложението и в действителност тя трябва да бъде включена в края на всички израз отчети във всички C++ програми (един от най-често срещаните грешки, синтаксис е наистина да забрави да включи някои запетая след изявление).

```
return 0;
```

Връщането изявление причинява основната функция, за да продължи. връщане може да бъде последван от кода на връщане (в нашия пример е последван от кода на връщане на стойност, равна на нула). А код на връщане на *0* в *полза* на *основната* функция обикновено се тълкува като програмата работи както се очаква, без никакви грешки по време на неговото изпълнение. Това е най-обикновен начин да сложи край на C++ програма конзола.

Може би сте забелязали, че не всички линии на тази програма, извършване на действия, когато кодът се изпълнява. Имаше редове, които съдържат само коментари (тези, започващи с `//`). Имаше линии с директивите за Препроцесорът на компилатор (тези, започващи с `#`). Тогава там са линии, които са започнали на декларация на функция (в този случай, основната функция) и накрая линии с отчети (като вмъкване в *Cout),* които са включени в рамките на блок ограничено от скоби (`{}`) на чиято основна функция.

Програмата е структурирана по различни линии, за да бъдат по-разбираеми, но в C + +, ние нямаме строги правила за това как да се разделят инструкции в различни линии. Например, вместо

```
1  int main()
2  {
3    cout << "Hello World!";
4    return 0;
5  }
```

Ние би могъл да напише:

```
int main() { cout << "Hello World!" ; return 0; }
```

Всички само в една линия и това би трябвало точно същото значение, както в предишния код.

В C + +, разстоянието между отчети е посочено завършва с точка и запетая (;) в края на всеки един, така че отделянето на различни линии код няма значение на всички за тази цел. Можем да напишем много изявления на ред или напишете един отчет, който взема много редове код. Разделението на код в различни линии служи само за да стане по-четливи и схеми за хората, които могат да го прочетат.

Нека да се добави допълнителна инструкция за първата ни програма:

```
1  // my second program in C++
2  #include <iostream>;
3  using namespace std;
4
5  int main()
6  {
7    cout << "Hello World! " ;
8    cout << "I am a C++ program" ;
9    return 0;
10 }
11
12
```

```
Hello World!I am a C++ program
```

В този случай, ние извършихме две вмъквания в Cout в две различни изявления. За пореден път, разделянето в различни редове код е направено само за да се даде по-голяма четливост на програмата, тъй като *основната* можеше да бъде напълно валидни определени по следния начин:

```
int main() { cout << " Hello World! " ; cout << " I'm a C++ program " ; return 0; }
```

Ние също бяха свободни да се разделят на код в повече линии, ако счете за по-удобно:

```
1 int main()
2 {
3   cout << "Hello World!" ;
4   cout << "I am a C++ program" ;
5   return 0;
6 }
7
8
```

И резултатът отново ще са точно същите, както в предишните примери.

Preprocessor директиви (тези, които започват с `#`), са от това общо правило, тъй като те не са отчети. Те са линии четат и обработват от Preprocessor и не произвежда никакъв код от себе си. Preprocessor директиви трябва да бъдат посочени в собствената си линия и не трябва да завършва с точка и запетая (`;`).

Коментари

Коментари са части от изходния код пренебрегнати от компилатора. Те просто не правете нищо. Тяхната цел е само да се даде възможност на програмиста да вмъкнете бележки или описания вградена в изходния код.

C++ поддържа два начина за вмъкване на коментари:

```
// line comment
/* block
comment */

```

Първият от тях, известен като линия за коментар, изхвърля всичко от където двойката признаци наклонена черта (`//`) се намира до края на същата линия. Вторият, известен като блока си, изхвърля всичко между `/*` символи и първата поява на `*/` герои, с възможност за включване на повече от един ред.
Ние ще добавят коментари към втория нашата програма:

```
/* my second program in C++ with more comments */
#include <iostream>;
using namespace std;

int main()
{
  cout << "Hello World! " ; // prints Hello World!
  cout << "I am a C++ program" ; // prints I am a C++ program
  return 0;
}


```

```
Hello World!I am a C++ program
```

Ако включите коментари в рамките на изходния код на програмите си, без да използвате коментар герои комбинации `//` , `/*` или `*/` , компилаторът ще ги вземе, сякаш те са C + + изрази, най-вероятно причинява едно или няколко съобщения за грешка, когато го събират .