MÔI TRƯỜNG LẬP TRÌNH C++

Để làm việc với C++ bạn cần phải có trình soạn thảo và cài đặt môi trường để biên dịch mã. Ở đây, có một số công cụ hoàn toàn miễn phí và sử dụng phổ biến như Dev C++, Eclipse, Code Block, Visual Studio… Các phần mềm này có sự hỗ trợ cho việc soạn thảo cũng như tạo sẵn môi trường để biên dịch mã C++.

Để bắt đầu với C++, chúng ta đến với chương trình Hello World:

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

Trong đó:

- Dòng #include <iostream> **là thư viện được khai báo trong mỗi chương trình C++. Đây là môi trường nhập / xuất (input / output) dữ liệu trong C++. Ngoài ra, với các bài toán chúng ta cần quan tâm đến 2 thư viện toán học đặc biệt quan trọng là** cmath **và** algorithm

- Dòng using namespace std **là không gian tên. Nếu ko có dòng này bạn cần nhập** std::cout << "Hello World!";

- Toàn bộ chương trình chính sẽ nằm trong phần:

```
int main()
{
 // Nội dung lệnh
 return 0;
 }
```

Lưu ý:

- Mọi câu lệnh trong C++ đều kết thúc bằng dấu ";".

- Để nhập dữ liệu ta sử dụng cin, xuất dữ liệu ta sử dụng cout.

Ví dụ:

```
#include <iostream>
using namespace std;
int main()
{
 int a;
 cin >> a;
 cout << "a = " << a;
 return 0;
 }
```

- Nếu bạn bỏ qua không gian tên using namespace std **thì câu lệnh nhập / xuất dữ liệu sẽ cần khai báo std:**

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

- Hai câu lệnh return 0 và return 1 là hai giá trị trả về duy nhất của hàm main() trong C++.

Hai giá trị trả về này của hàm int main() **có ý nghĩa như sau:**

Chúng ta chỉ đinh return 0 để kết thúc chương trình theo cách bình thường (normal termination). Điều đó có nghĩa là kể cả chương trình có xảy ra lỗi hay không, thì C++ vẫn ngầm định là chương trình đã được kết thúc mà không có lỗi xảy ra.

Chúng ta chỉ đinh return 1 để kết thúc chương trình theo cách bất thường (abnormal termination). Điều đó có nghĩa là khi chương trình xảy ra lỗi, thì lỗi này sẽ được trả về khi kết thúc chương trình.

1. Các kiểu dữ liệu

Với các bài toán xử lý trên C++, chúng ta cần quan tâm một số kiểu dữ liệu như sau:

- **Kiểu số nguyên** int **là kiểu dữ liệu thường xuyên sử dụng**
- **Kiểu** string, char**: khai báo trong những bài toán về xâu ký tự**
- **Kiểu** bool**: Trả về** true **(1) hoặc** false **(0)**
- **Kiểu** float**,** double**: trả về kiểu số thực**
- **Kiểu** long**: Trả về kiểu số nguyên lớn hơn** int

Type name	Bytes	Range of values
bool	1	false or true
char	1	-128 to 127
int	4	-2147483648 to 2147483647
unsigned int	4	0 to 4294967295
__int16	2	-32768 to 32767
__int32	4	-2147483648 to 2147483647
__int64	8	-9223372036854775808 to 9223372036854775807
float	4	3.4E +/- 38 (7 digits)
double	8	1.7E +/- 308 (15 digits)

Tùy theo tình huống và dạng bài tập mà chúng ta sử dụng dữ liệu làm sao cho hợp lý. Có những bài toán yêu cầu về số nguyên lớn ta có thể sử dụng int8_t, int32_t, int64_t **để tránh trường hợp file test với số nguyên lớn làm bạn mất điểm do bị tràn số. Còn với những trường hợp không yêu cầu về số nguyên lớn ta có thể sử dụng kiểu thông dụng là** int**. Các kiểu dữ liệu thường dùng là** int, char, string, float, double

2. Biến, hằng, các phép toán và toán tử

- Khai báo biến: Việc khai báo biến trong C++ rất đơn giản, cấu trúc chung thường là:

Kiểu_dữ_liệu Tên_biến;

Hoặc: Kiểu_dữ_liệu biến1, biến2, biến3;

Ví dụ: int a, b, c;

Bạn hoàn toàn có thể khai báo biến đồng thời với việc gán một giá trị cụ thể nào đó cho biến.

Ví dụ: int a = 50, b = 100;

- Khai báo hằng: Việc khai báo hằng cũng giống kiểu khai báo biến

Ví dụ: const int a = 50;

- Các phép toán: C++ sử dụng các phép toán cơ bản giống như các ngôn ngữ lập trình khác:

+ Phép cộng: +
+ Phép trừ: -
+ Phép nhân: *
+ Phép chia: /
+ Phép chia có dư: %

- Toán tử: Chúng ta cần lưu ý một số toán tử thông dụng trong C++

Toán tử	**Ý nghĩa**	**Ví dụ**
==	Bằng	x == y

!=	Không bằng	x != y
>	Lớn hơn	x > y
<	Nhỏ hơn	x < y
>=	Lớn hơn hoặc bằng	x >= y
<=	Nhỏ hơn hoặc bằng	x <= y

- Toán tử logic: Chúng ta quan tâm đến 3 toán tử logic thường dùng:

Toán tử	**Ý nghĩa**	**Ví dụ**
&&	AND	x < 5 && x < 10
||	OR	x < 5 || x < 4
!	NOT	!(x < 5 && x < 10)

- Toán tử gán: Giữ vai trò đặc biệt quan trọng trong lập trình. Chúng ta cần chú ý một số phép gán trong C++ dưới đây:

Toán tử	**Ví dụ**	**Tương đương**
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
++	x++	x = x + 1
--	x--	x = x - 1

Lưu ý: luôn tránh những loại lệnh không rõ ràng, ++i, i++, --i, i-- . Chỉ dùng ++, -- khi biết chắc chắn thứ tự của chúng

```
j = ++i  / i = i+1 , j =i   tăng i lên rồi mới lấy i sử dụng
j = i++  / j = i , i = i +1  lấy i sử dụng trước rồi tăng i lên

a=10;   b=a++;   a =11, b=10
a=10;   b=a--;   a =9, b=10
a=10;   b=++a;   a =11, b=11
a=10;   b=--a;   a =9, b=9

++--i thực hiện trước +
```

```
a=10;
b = 1 + a++; -> b = 11    , a = 11
b = 1 + a--; -> b = 11    , a = 9

a=10;
b = 1 + ++a; -> b = 12   a = 11
b = 1 + --a; -> b = 10   a = 9

a=10;
b = 1 + a++; -> b = 11    , a = 11
c = 1 + ++a; -> c = 13   a = 12

------------------------
bản chất ++i  ( --i tương tự )
{
  return i+=1;    // i = i + 1;
}

bản chất i++:   ( i--  tương tự )
{
  int temp;
  temp = i;
  i = i + 1;
  return temp;
}
```

3. Hàm

Hàm là một nhóm các lệnh đi cùng nhau để thực hiện một nhiệm vụ nào đó trong chương trình. Mỗi chương trình C++ có ít nhất một hàm: int main()

Cấu trúc:

Kiểu_trả_về tên_Hàm(Danh sách tham số)
{
Thân hàm
}

Trong đó:

Kiểu trả về: Một hàm có thể trả về một giá trị. Kieu_tra_ve là dạng dữ liệu của giá trị mà hàm trả về. Vài hàm cung cấp các hoạt động và không trả về giá trị nào cả. Đó là hàm void.

Tên hàm: Đây là tên thực sự của hàm. Tên hàm và danh sách tham số cấu tạo nên dấu hiệu hàm.

Danh sách tham số: Khi hàm được gọi, bạn phải truyền vào danh sách các tham số. Một giá trị hướng đến một tham số thực tế. Danh sách tham số có các kiểu, thứ tự và số lượng các tham số của hàm. Các tham số trong hàm là tùy chọn, nghĩa là một hàm có thể không có tham số.

Thân hàm: Phần thân của một hàm bao gồm tập hợp các lệnh xác định những gì mà hàm thực hiện.

Để gọi hàm, đơn giản là chỉ cần truyền các tham số được yêu cầu cùng với tên của hàm và nếu hàm trả về các giá trị, bạn có thể dự trữ các giá trị trả về.

Xét một ví dụ đơn giản về hàm tính tổng hai số:

```
#include <iostream>
using namespace std;
int sum(int a, int b=20)
{
  int ketqua;
```

```
  ketqua = a + b;
  return (ketqua);
}
int main ()
{
   // Khai bao bien cuc bo:
  int a = 100;
  int b = 200;
  int ketqua;
  // goi ham de tinh tong hai so.
  ketqua = sum(a, b);
  cout << "Tong gia tri la: " << ketqua << endl;
  // goi ham mot lan nua.
   ketqua = sum(a);
  cout << "Tong gia tri la: " << ketqua << endl;
  return 0;
}
```

4. Tệp và ghi tệp

Với việc ghi và đọc file chúng ta có thể sử dụng một trong 2 cách:

Cách 1: Sử dụng thư viện fstream

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
  ifstream fin;
  ofstream fout;
  int a, b, Tong = 0, Tich = 0;
   fin.open("Tong.INP",ios::in);
   fout.open("Tong.OUT",ios::out);
  fin>>a;
  fin>>b;
  fin.close();
  fout << a + b << endl;
  fout << a * b;
  fout.close();
  return 0;
}
```

Cách 2: Sử dụng Hàm freopen:

```
#include <iostream>
using namespace std;
int main()
{
  int a;
  freopen("TongChuSo.INP","r",stdin);
   freopen("TongChuSo.OUT","w",stdout);
  cin >> a;
  int c = a % 10;
  int d = a / 10;
  cout << c + d;
  return 0;
}
```

BÀI TẬP

Bài 1. Viết chương trình tính tổng các chữ số của số nguyên n có hai chữ số

- **Dữ liệu vào**: Cho từ tệp **INPUT.INP** gồm một dòng chứa số n
- **Dữ liệu ra**: Ghi ra tệp **OUTPUT.OUT** gồm một dòng chứa số là tổng các chữ số của số n

Ví dụ:

INPUT.INP	**OUTPUT.OUT**
34	7

Bài 2. Viết chương trình tính tích các chữ số của một số nguyên n có 3 chữ số

- **Dữ liệu vào**: Cho từ tệp **TICH.INP** gồm một dòng chứa số n
- **Dữ liệu ra**: Ghi ra tệp **TICH.OUT** gồm một dòng chứa số là tích các chữ số của số n

Ví dụ:

TICH.INP	**TICH.OUT**
254	**40**

Bài 3. Viết chương trình tính S = n(n+1)(n+2)/3. Với n là số nguyên dương (n <= 2^{32}).

- **Dữ liệu vào**: Cho từ tệp **TS.INP** gồm một dòng chứa số n
- **Dữ liệu ra**: Ghi ra tệp **TS.OUT** gồm một dòng chứa số s

Ví dụ:

TS.INP	**TS.OUT**
7	**168**

Bài 4. Người ta đổi T (giây) thành a (giờ), b (phút), c (giây). Hãy tìm a, b, c.

- **Dữ liệu vào**: Cho từ tệp **TIME.INP** gồm một dòng chứa số T
- **Dữ liệu ra**: Ghi ra tệp **TIME.OUT** gồm một dòng chứa các số a, b và c ngăn cách nhau bởi dấu ":"

Ví dụ:

TIME.INP	**TIME.OUT**
6695	**1:51:35**

Bài 5. Nam có X đồng. Biết rằng tờ tiền có mệnh giá là 100, 50, 20, 10, 5, 2, 1. Hỏi số lượng tờ tiền ít nhất mà Nam cầm trên tay là bao nhiêu? Liệt kê từng loại tiền mà Nam có.

- **Dữ liệu vào**: Cho từ tệp **VND.INP** gồm một dòng chứa số X
- **Dữ liệu ra**: Ghi ra tệp **VND.OUT** gồm:
  + Dòng thứ nhất ghi tổng số tờ tiền,
  + Dòng thứ hai ghi số tờ tiền của từng loại (mỗi loại cách nhau một dấu cách).

Ví dụ:

VND.INP	**VND.OUT**
6577	**69** **65 1 1 0 1 1 0**