

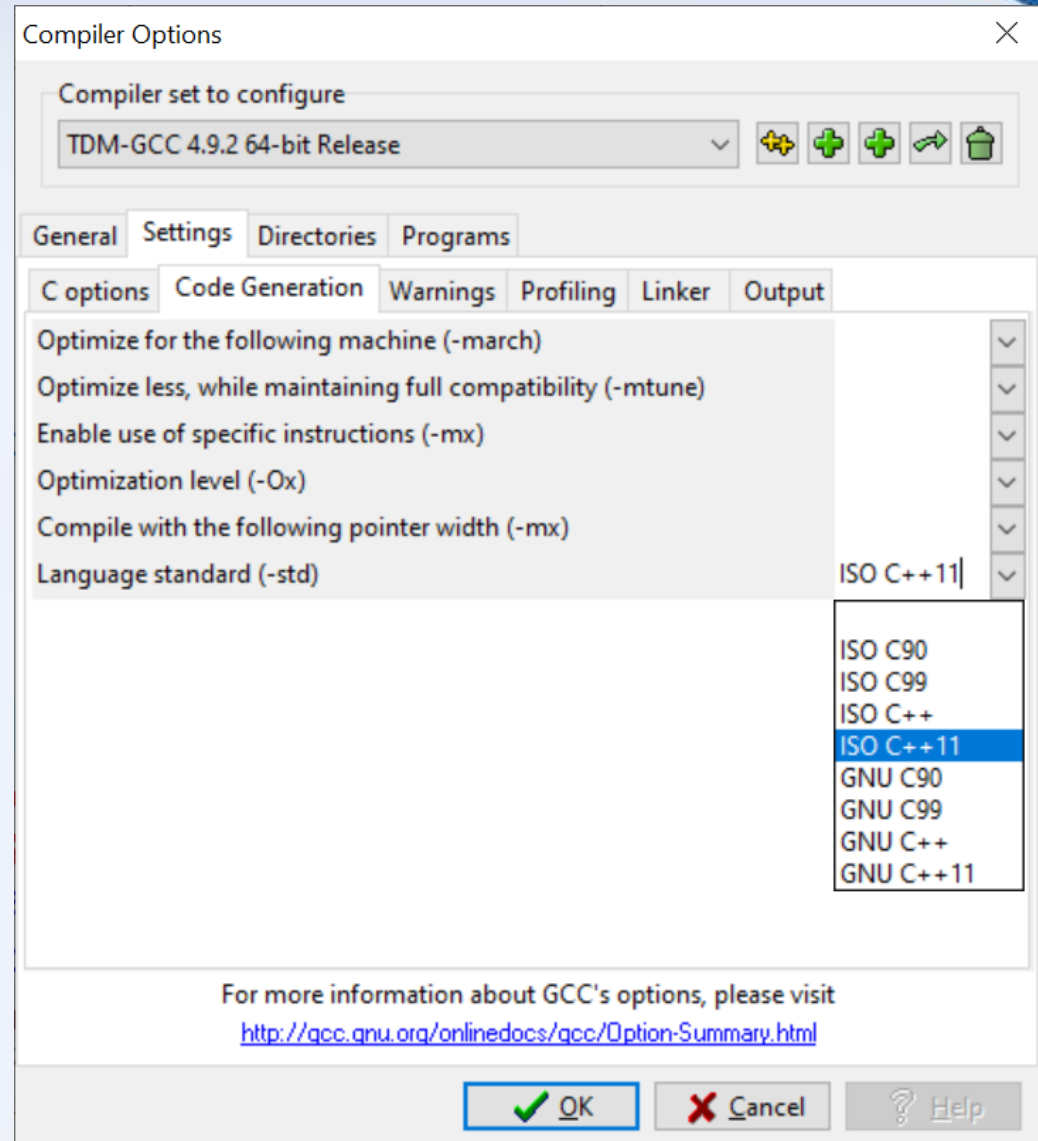
Bài 2. Ngôn ngữ lập trình C++

I. Giới thiệu

- Ngôn ngữ lập trình C++ là ngôn ngữ được phát triển dựa trên ngôn ngữ lập trình C.
- Do đó về cơ bản, cú pháp của C++ giống với cú pháp của C. Tuy nhiên nó có một số mở rộng sau đây:
 - Nhập, xuất dữ liệu (cout, cin)
 - Hàm có đối mặc định, hàm có đối tham chiếu
 - Nạp chồng hàm (hay tải bội hàm – overload function)
 - Hàm mẫu
 - Lớp (có khả năng xây dựng các chương trình HĐT)

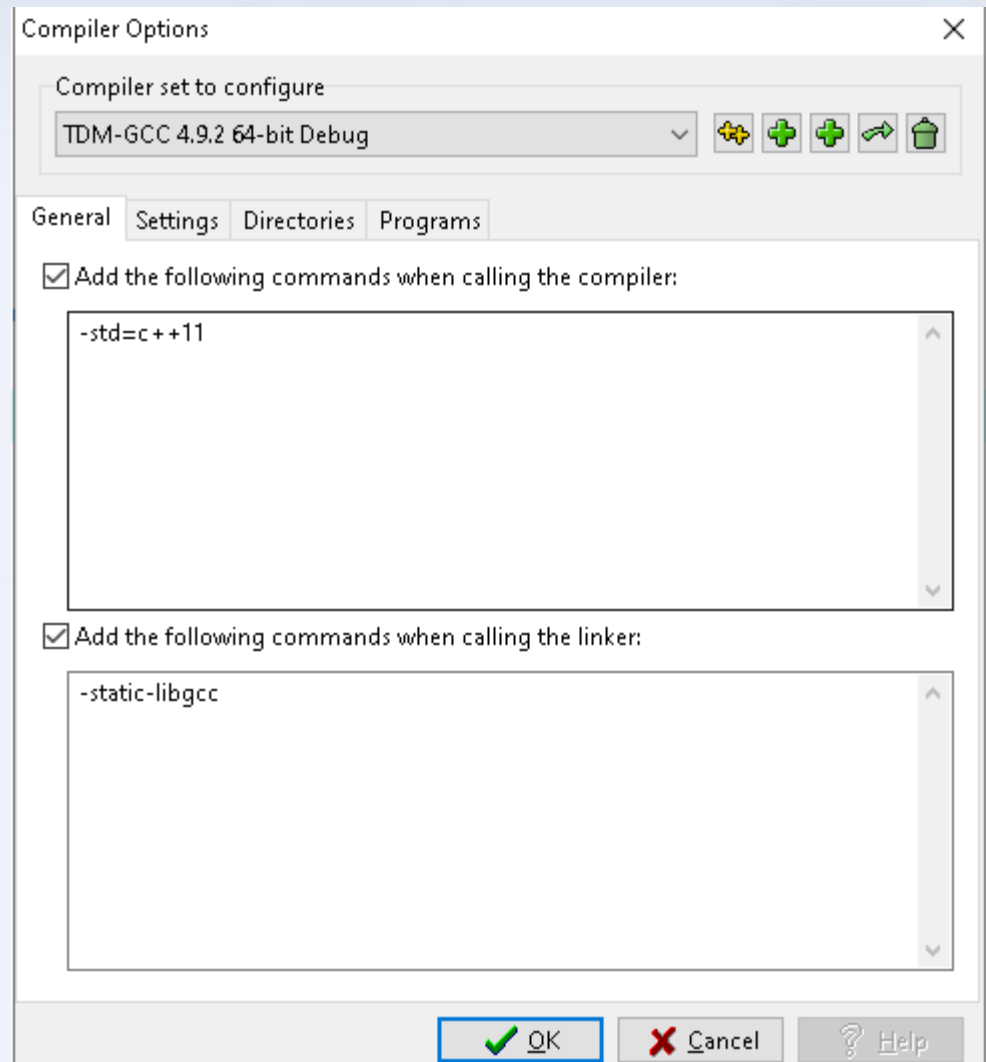
Thiết lập biên dịch C++11 cho dev C++

- Tools-> Compiler options -> Settings -> Code Generation
- Chọn Language standard (-std) có giá trị:
ISO C++11



Thiết lập biên dịch C++11 cho dev C++

- Tools-> Compiler options -> General
- Check vào box
Add the following commands when calling the compiler
- Soạn vào
`-std=c++11`



II. Nhập xuất dữ liệu

- Nhập dữ liệu **kiểu số**

cin>>Tênbiến1>>Tênbiến2>>...>>Tênbiếnn;

Ví dụ:

float x,y;

int m, n;

cin>>x>>y;

cin>>m;

cin>>n;

- Nhập dữ liệu kiểu **xâu ký tự**

```
cin.ignore(0);
```

```
cin.get(Tênbiến, n); //n là số ký tự tối đa cần  
gán cho biến
```

Ví dụ:

```
char ht[30];
```

```
char w[10];
```

```
cin.ignore(0);
```

```
cin.get(ht, 30);
```

```
cin.ignore(0);
```

```
cin.get(w, 5);
```

- Xuất dữ liệu

```
cout<<Bthức1<<Bthức2<<...<<Bthứcn;
```

Ví dụ:

```
#include "iostream.h"
```

```
void main()
```

```
{
```

```
    float x, y=10;
```

```
    cout<<"Nhập x=";
```

```
    cin>>x;
```

```
    cout<<"x+y="<<x+y;
```

```
    cout<<"x-y="<<x-y;
```

```
}
```

III. Hàm

- Khi xây dựng các hàm ngoài các kiểu hàm như trong C thì C++ còn cho phép xây dựng các kiểu hàm sau đây:
 - Đối tham chiếu
 - Đối mặc định
 - Nạp chồng hàm (overload function)
 - Hàm mẫu (template)

• Hàm có đối tham chiếu

- Khai báo hàm:

`DataType Func_Name(DataType & Arg_Nam,...);`

- Sử dụng hàm: Các đối thực sự tương ứng với đối tham chiếu phải là các biến cùng kiểu với kiểu của đối.
- Sự hoạt động của hàm như hàm có đối con trỏ

Ví dụ: Xây dựng hàm hoán đổi giá trị của hai biến

```
void hoandoi(float &a, float &b)
{
    float tg;
    tg = a;
    a = b;
    b = tg;
}
```

```
void main() {
    float x, y;
    cout<<"Nhap x, y: ";
    cin>>x>>y;
    cout<<"x = "<<x<<" y = "<<y<<endl;
    hoandoi(x,y);
    cout<<"x = "<<x<<" y = "<<y;
    getch();
}
```

```
#include <iostream.h>
```

```
#include <conio.h>
```

```
void duplicate (int& a, int& b, int& c)
```

```
{
```

```
    a = 2; b = 2; c = 2;
```

```
}
```

```
int main (){
```

```
    int x=1, y=3, z=7;
```

```
    duplicate (x, y, z);
```

```
    cout << "x=" << x << ", y=" << y << ", z=" << z;
```

```
    return 0;
```

```
}
```

• Hàm có đối mặc định

- Khai báo hàm

`DataType Func_Name(DataType Arg_Nam1, DataType Arg_Nam2 = value2, ...);`

- Sử dụng hàm: Có thể không truyền đối thực sự cho đối mặc định
- Nếu truyền thì hàm nhận giá trị của đối thực sự, nếu không truyền hàm nhận giá trị mặc định

`Func_Name(Arg1, Arg2);`

`Func_Name(Arg1);`

Ví dụ:

```
#include <iostream.h>
```

```
#include <conio.h>
```

```
int divide (int a=10, int b=2) {
```

```
    int r; r=a/b;
```

```
    return (r);
```

```
}
```

```
int main () {  
    cout<<divide();  
    cout << divide (12);  
    cout <<"\n"<< divide (12, 4);  
    return 0;
```

```
}
```

5

Kết quả

6

3

- **Nạp chồng hàm (overload function)**

- Nạp chồng hàm là khả năng cho phép định nghĩa lại một hàm đã có. Tức là trong một chương trình cho phép nhiều hàm trùng tên nhau.

- **Một số lưu ý khi nạp chồng hàm**

Các hàm phải có ít nhất một trong các đặc điểm sau:

- Khác nhau về số lượng đối
- Khác nhau về kiểu của đối
- Khác nhau kiểu trả về của hàm

- Ví dụ:

- Hàm nhập một dãy số

- `void Nhapday(float *, int);`

- `void Nhapday(int *, int);`

- Hàm tìm uscln của hai số nguyên

- `int uscln(int,int);`

- `long uscln(long, long);`

- `long uscln(long, int);`

Hàm mẫu (template)

- Hàm mẫu là hàm được xây dựng như là một mẫu để thực hiện một chức năng nào đó mà kiểu của các đối vào chưa được xác định.

- **Khai báo**

```
template<class DataType,...>
```

```
DataType Func_Name(DataType Arg_Name,...){
```

```
    các câu lệnh;
```

```
};
```

Trong đó **DataType** là một tên kiểu bất kỳ do người lập trình đặt

Ví dụ 1

```
#include <iostream.h>
#include <conio.h>
template <class T>
T GetMax (T a, T b)
{ T result;
  result = (a>b)? a : b;
  return (result);
}
```

```
int main () {
int i=5, j=6, k;
long l=10, m=5, n;
k = GetMax(i,j);
n = GetMax(l,m);
cout << k << endl;
cout << n << endl;
return 0;
}
```

Ví dụ 2

Xây dựng hàm nhập, in một dãy số có kiểu bất kỳ

```
template<class T>
void Nhapday(T *a, int n, char ch){
    for(int i=0; i<n; i++){
        cout<<ch<< "["<<i<<"]="";
        cin>>a[i];
    }
}
```

```
template<class D>
void Inday(D *a, int n){
    for(int i=0; i<n; i++)
        cout<<a[i] <<" ";
}
```

```
void main(){
    int m,n;
    float a[100];
    long b[100];
    cout<<"Nhap m,n:";
    cin>>m>>n;
    Nhapday(a,m,'a');
    Nhapday(b,n,'b');
    Inday(a,m);
    Inday(b,n);
}
```


Hết