CHƯƠNG 4 :

HÀM VÀ CHƯƠNG TRÌNH.

4.1 Mở đầu

4.2 Hàm

4.3 Tổ chức chương trình C++ bằng Win32 Console Application

4.4 Một số thư viện trong C++

4.1 Mở đầu

4.1.1 Cấu trúc chung của 1 chương trình C++

4.1.2 Vai trò của hàm

4.1.3 Một số quy tắc cần nhớ khi viết mã chương trình

4.1.1 Cấu trúc chung của 1 chương trình C++

- Một chương trình C++ có thể gồm nhiều tập tin chương trình nguồn, mỗi tập tin chương trình là một văn bản chứa một dãy các chỉ thị và các chỉ thị điều khiển biên dịch.

- Các chỉ thị được phân thành 2 loại :
  - ❖Chỉ thị kiểu :

    Bao gồm định nghĩa các kiểu dữ liệu mới, khai báo biến, hằng và hàm.
  - ❖Chỉ thị thực hiện:

    Được định nghĩa bằng những phép toán hay việc xử lý thực hiện trên các biến của chương trình.

Tất cả các chỉ thị đều phải kết thúc bằng dấu ;

- Cả 2 loại chỉ thị này có thể hợp với nhau bằng một cú pháp qui định để hình thành một chỉ thị duy nhất được gọi là khối lệnh .

  Một khối lệnh được đặt trong cặp dấu ngoặc nhọn :

  { các chỉ thị }

Dạng tổng quát của chương trình C++:

```
// Các chỉ thị điều khiển biên dịch
#include
#define
```

```
// Các định nghĩa toàn cục
```

```
// Khai báo nguyên mẫu các hàm
```

```
// Hàm main
int main() // void main()
{
// dãy tuần tự các lệnh
}
```

```
//Phần định nghĩa hàm.
Kdl   Ham(Danh_Sach_Cac_Doi)
{
// dãy tuần tự các lệnh
}
.   .   .   .   .
```

- Một chương trình C++ có thể chỉ có một hàm hoặc bao gồm nhiều hàm và các hàm này không được phép lồng nhau, Trong đó hàm main() luôn luôn xuất hiện.

  Chương trình bắt đầu thực hiện tại điểm đầu hàm main(). Nghĩa là hàm main() sẽ khởi động các hàm khác để thực hiện công việc.

  Những hàm khác có thể nằm trong cùng một tập tin với main() hoặc tập tin thư viện.

- Hàm main()

  Có 2 dạng khai báo thường dùng :

  - int main() // Không đối
  - void main()

  Các dạng khác như :

  int main(void),

  int main(int argc, char *argv) // Có đối

  ...

  Trong hàm main ta thường dùng câu lệnh return 1 hay return 0 (trả về giá trị 1 hay 0) để kết thúc chương trình.

- Các hàm liên lạc với nhau thông qua danh sách các đối (tham số)của hàm.

  Hàm được khởi động bằng cách gọi tên nó.
- Các chỉ thị tiền xử lý :

  #include , #define, . . .

- Các định nghĩa toàn cục:

  - Định nghĩa các hằng.
  - Khai báo các đối tượng dữ liệu ngoài (biến, mảng , cấu trúc ,. . .)

- Khai báo nguyên mẫu các hàm (prototype).

- Phần định nghĩa các hàm.

  Hàm main() có thể đặt sau, đầu, hay xen giữa các định nghĩa hàm .

4.1.2 Vai trò của hàm:

- C/ C++ đưa ra khái niệm hàm và trở thành công cụ mạnh mẽ để hỗ trợ cho phương pháp lập trình có cấu trúc.

- Hàm có thể xem là một đơn vị độc lập của chương trình. Các hàm có vai trò ngang nhau, vì vậy không cho phép xây dựng một hàm bên trong các hàm khác.

- Các hàm thường che dấu những chi tiết thực hiện đối với các phần khác trong chương trình, do đó làm chương trình sáng sủa , dễ sửa đổi.

4.1.2 Vai trò của hàm:

Một hàm có thể là:

- Nằm ngay trong modul văn bản (có các khai báo, các lệnh...) hoặc được đưa một cách tự động vào văn bản của chương trình (bằng đường dẫn #include) hay được dịch riêng rẽ (sẽ được nối kết vào chương trình trong giai đoạn liên kết).
- Được gọi từ hàm chính, hoặc từ một hàm khác, hoặc từ chính nó (đệ qui).
  - Có hay không có đối (tham số hình thức).
  - Có hay không có giá trị trả về.
  - . . . .
- Một hàm :
  - Chỉ có 1 đầu vào ({).
  - Có thể có nhiều điểm ra (return hay }).

4.1.3 Một số quy tắc cần nhớ khi viết mã chương trình

- Quy tắc 1 :

  Mỗi dòng có thể viết 1 hay nhiều chỉ thị.

- Quy tắc 2 :

  Mỗi chỉ thị phải kết thúc bằng dấu chấm phảy (;).

- Quy tắc 3 :

  Quy tắc viết lời giải thích. Các lời giải thích viết :

  - Trên nhiều dòng, một dòng hoặc trên 1 phần của dòng phải đặt vào giữa các dấu /* và */ .
  - Trên một dòng hoặc trên phần còn lại của một dòng phải đặt sau //

  Các lời giải thích được trình biên dịch bỏ qua.

- Quy tắc 4 (Sử dụng các thư viện)

  C++ có nhiều thư viện chứa các hàm, lớp...mà giao diện của nó lưu trử trong các tập tin - ta thường gọi là tập tin tiêu đề.

  Khi ta sử dụng hàm nào thì dùng chỉ thị tiền xử lý #include để chỉ hàm đó nằm trong tập tin tiêu đề nào, với cách viết:

  #include<TênTệp>

  Chẳng hạn:

  #include<iostream>

4.2 Hàm

4.2.1 Cấu trúc của 1 hàm

4.2.2 Các ví dụ về tạo hàm

4.2.3 Sử dụng hàm

4.2.4 Khai báo nguyên mẫu của hàm (prototype)

4.2.5 Hoạt động của hàm

4.2.6 Truyền tham số

4.2.1 Cấu trúc của 1 hàm :

Mọi hàm đều có dạng :

```
[Kdl]    TenHam ([danh_sách_kiểu_và_Đối])  // Dòng tiêu đề
{
      // Các định nghĩa, khai báo biến
      // Các câu lệnh;
         [return [biểu_thức];]
}
```

Trong đó :

a) Dòng tiêu đề :

Chứa các thông tin :

Kdl : Kiểu dữ liệu của hàm,

TenHam : Tên của hàm.

Danh_sách_kiểu_và_Đối : Là các đối và kiểu tương ứng của nó.

Cuối dòng tiêu đề không có dấu chấm phảy (;) như kết thúc câu lệnh.

1. Kdl : Có thể có hoặc không.

- *Trương hợp không* :

Không có kiểu dữ liệu của hàm, Một cách ngầm định C++ coi đó làkiểu int.

- *Trường hợp có* :

- Kdl là kiểu dữ liệu của hàm, có thể là bất kỳ kiểu dữ liệu nào ngoại trừ mảng. C++ sẽ dùng từ khóa void để chỉ kiểu dữ liệu của hàm trong trường hợp hàm không trả về giá trị nào cả. Cần lưu ý là :

* Nếu hàm trả về kiểu int, ta có thể không khai báo (ngầm định là kiểu int).

* Nếu trả về khác kiểu int, ta phải khai báo tường minh kiểu dữ liệu của hàm.

2. TenHam : Là tên của hàm do người lập trình tự đặt theo quy tắc đặt tên.

   Qui ước đặt tên hàm : Tên hàm có thể có nhiều từ, ký đầu của mỗi tử viết HOA, còn lại sẽ viết thường. Nhiều khi để dễ đọc, có thể dùng ký tự gạch dưới để nối các từ. Từ đầu sẽ đặt bằng một động từ.

3. Danh sách kiểu và đối :

   Có thể có hoặc không.

- *Trương hợp không* : Nếu không có đối thì vẫn phải giữ các dấu ngoặc tròn ().Trong trường hợp này có thể thay bằng từ khóa void.
- *Trường hợp có* : Đó là các đối của hàm. Trước mỗi đối có kiểu dữ liêu tương ứng của nó, nếu có nhiều đối và kiểu tương ứng của nó, thì chúng phải cách nhau dấu phảy (,).

Chẳng hạn :

```
float Max(float a, float b, float c)
double F(double x, int n)
```

b. Thân hàm :

Thân hàm bắt đấu bằng dấu ngoặc nhọn mở {, tiếp theo là các chỉ thị về kiểu (nếu có) , các câu lệnh trong đó có thể có hay không câu lệnh return và kết thúc bằng dấu ngoặc nhọn đóng.

c. Câu lệnh return :

Trong thân hàm có thể có hoặc không có câu lệnh này. Trong trường hợp có , có thể có một câu lệnh return, hoặc nhiều câu lệnh return ở những nơi khác nhau,

Nếu không có câu lệnh return thì chương trình sẽ ra khỏi hàm khi gặp dấu ngoặc nhọn đóng cuối cùng } của thân hàm để trở về nơi gọi nó.

Nếu có, Câu lệnh "return" là cơ chế chuyển giá trị từ hàm được gọi về nơi gọi . Khi gặp câu lệnh return máy sẽ không thực hiện các câu lệnh sau nó trong hàm chứa câu lệnh này.

Dạng tổng quát của câu lệnh này là :

return [Bt];

Ý nghĩa của các dạng có thể minh họa như sau :

Dạng 1 : return;

Dùng để thoát ra khỏi 1 hàm và trở về hàm đã gọi nó, mà không trả về một giá trị nào. Trường hợp này khai báo kiểu dữ liệu của hàm là void. Câu lệnh return có thể dùng để ra khỏi thân switch, các vòng lặp.

Dạng 2 : return Bt;

hoặc return (Bt);

Giá trị của biểu thức Bt sẽ được chuyển kiểu cho phù hợp với kiểu của hàm trước khi gán cho hàm, và ra khỏi hàm chuyển về nơi gọi nó.

4.2.2 Các ví dụ về tạo hàm.

Ví dụ 1: // Tính max cua 2 so

//Input a,b

//Output Max(a,b)

Ghi chú:

Hàm trả về một trị, nên lưu trử trị trả về trong một biến. Có thể tổ chức dưới dạng :

```
Kdl TenHam(DS_Cac_Kieu_Va_Doi)
{
    Kdl kq;
     . . .
    return kq;
}
```

Ví dụ 2 :

Viết hàm đổi ký số thành số

//Input x

//Output Doi_KS_So(x)

Ví dụ 3:

Viết hàm : $F(x) = \begin{cases} 1; x \text{ lẻ} \\ 0; x \text{ chẵn} \end{cases}$

Ví dụ 4:

Viết hàm tính diện tích của một tam giác khi biết 3 cạnh a, b, c của tam giác.

Dùng công thức:

$$s = \sqrt{p(p-a)(p-b)(p-c)};$$

$$p = \frac{a+b+c}{2}$$

//Input a, b, c // Giả sử thỏa giá trị 3 cạnh của tam giác.
//Output s

Ví dụ 5 :

Hàm xuất ra màn hình các chức năng của chương trình.

```
void XuatMenu()
{
    cout<<" \n ****Bảng Menu ********** ";
    cout<<" \n0.****  Thoát  ********** ";
    cout<<" \n1.**chức năng 1  ********** ";
    cout<<" \n2.**chức năng 2  ********** ";
    cout<<" \n3.**chức năng 3  ********** ";
    cout<<"\n ************************** *";
}
```

Ví dụ 6:

Viết hàm chọn menu.

Input soMenu

output stt

Ví dụ 7:

Tính giá trị tuyệt đối của số nguyên.

input a

output | a |

4.2.3 Sử dụng hàm.

Hàm được sử dụng thông qua lời gọi hàm.

- Lời gọi hàm được viết như sau :

TenHam([Danh sách các tham số thực])

Trong đó :

- Số tham số thực phải bằng số các đối.
- Kiểu của tham số thực phải phù hợp với kiểu của đối tương ứng.

Ví dụ 8 (về lời gọi hàm):

Max(4.0, 3.2)

Doi_KS_So(‘5’)

F(5)

Tinh_DT_TG(3,4, 5)

Xuatmenu();

- Cách sử dụng như sau :

✓ Nếu kiểu dữ liệu của hàm là void thì lời gọi hàm được sử dụng như câu lệnh (thêm dấu ; cuối cùng), tức là :

TenHam([Danh sách các tham số thực]);

✓ Nếu kiểu dữ liệu của hàm khác kiểu void, lời gọi hàm được sử dụng :

- Như một toán hạng trong biểu thức.
- Vế phải câu lệnh gán.
- In giá trị của hàm .

. . . .

Ví dụ 9 (về cách sử dụng lời gọi hàm):

```
Xuatmenu();
cout<<Max(3.5, 6.4);
double kq  = 3*Max(3.5, 6.4);
cout<<Doi_KS_So('0');
```

4.2.4 Khai báo nguyên mẫu của hàm (prototype).

Trước khi sử dụng một hàm, ta có thể khai báo hoặc không khai báo nguyên mẫu của hàm trong chương trình.Vị trí khai báo có thể là ngoài tất cả các hàm, hoặc trong hàm. . . thường là ở đầu chương trình.

Dạng khai báo nguyên mẫu hàm là :

Kdl tên_hàm (Danh_sách_Kiểu_và _Đối); // Có dấu ;

Ví dụ :

int Doi_KS_So(char x);

Hoặc double Max(double a, double b);

Đối với C++ , Các trường hợp sau nhất thiết phải khai báo nguyên mẫu:

- Vị trí của hàm đặt sau hàm main().
- Các hàm gọi lẫn nhau.

Dù rằng có các trường hợp không nhất thiết phải khai báo nguyên mẫu, nhưng tốt hơn cả là ta vẫn khai báo để trình biên dịch dễ phát hiện lỗi khi gọi hàm.

4.2.5 Hoạt động của hàm.

Khi gặp một lời gọi hàm thì hàm bắt đầu thực hiện. Quá trình diễn ra theo trình tự :

a) Cấp phát bộ nhớ cho các đối và các biến địa phương.

b) Gán giá trị của các tham số thực cho các đối tương ứng.

c) Thực hiện các câu lệnh trong thân hàm.

d) Khi gặp câu lệnh return hoặc dấu } cuối cùng của thân hàm thì máy sẽ xóa các đối, các biến địa phương và ra khỏi hàm.

4.2.6 Truyền tham số

Có 2 cách :

- Truyền bằng trị
- Truyền bằng biến.

Phần này giới thiệu truyền bằng trị.

- Truyền bằng trị.

  - Hình thức thực hiện : Gán giá trị cùng kiểu cho đối.

(Tham số của hàm luôn được truyền theo tham trị .
Điều này có nghĩa là các gia trị thực
(tham số thực) không bị thay đổi giá trị
khi truyền cho các đối)

4.3 Tổ chức chương trình C++ bằng Win32 Console Application

4.3.1 Tập tin đề án

4.3.2 Cách tạo đề án

4.3.3 Các ví dụ

4.3.1 Tập tin đề án.

Một chương trình trong Windows tạo bằng Win32 Console Application theo VC++ trong mơi trường VS 2010 là một đề án bao gồm các thành phần sau:

a. Chương trình nguồn: Các tập tin dạng .*cpp, .*h. . .

- Tập tin *.h chứa các định nghĩa hằng số, biến, hàm,...

Đó là các tập tin tiêu đề của các thư viện, do ta tự xây dựng hoặc của Windows

- Tập tin *.CPP có thể có một hay nhiều, nhưng trong đó phải có một tâp tin chứa hàm main.

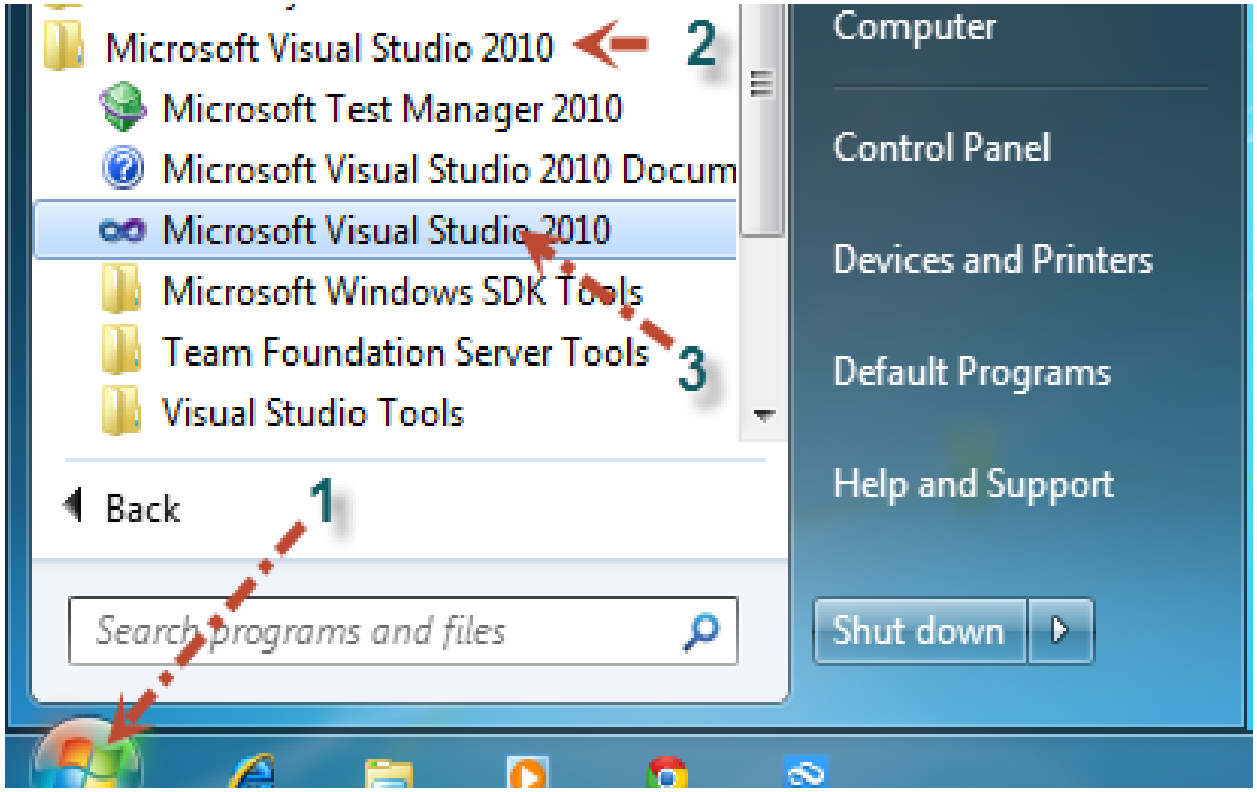
Tập tin *.cpp chứa các xử lý hoặc cài đặt hàm chức năng.

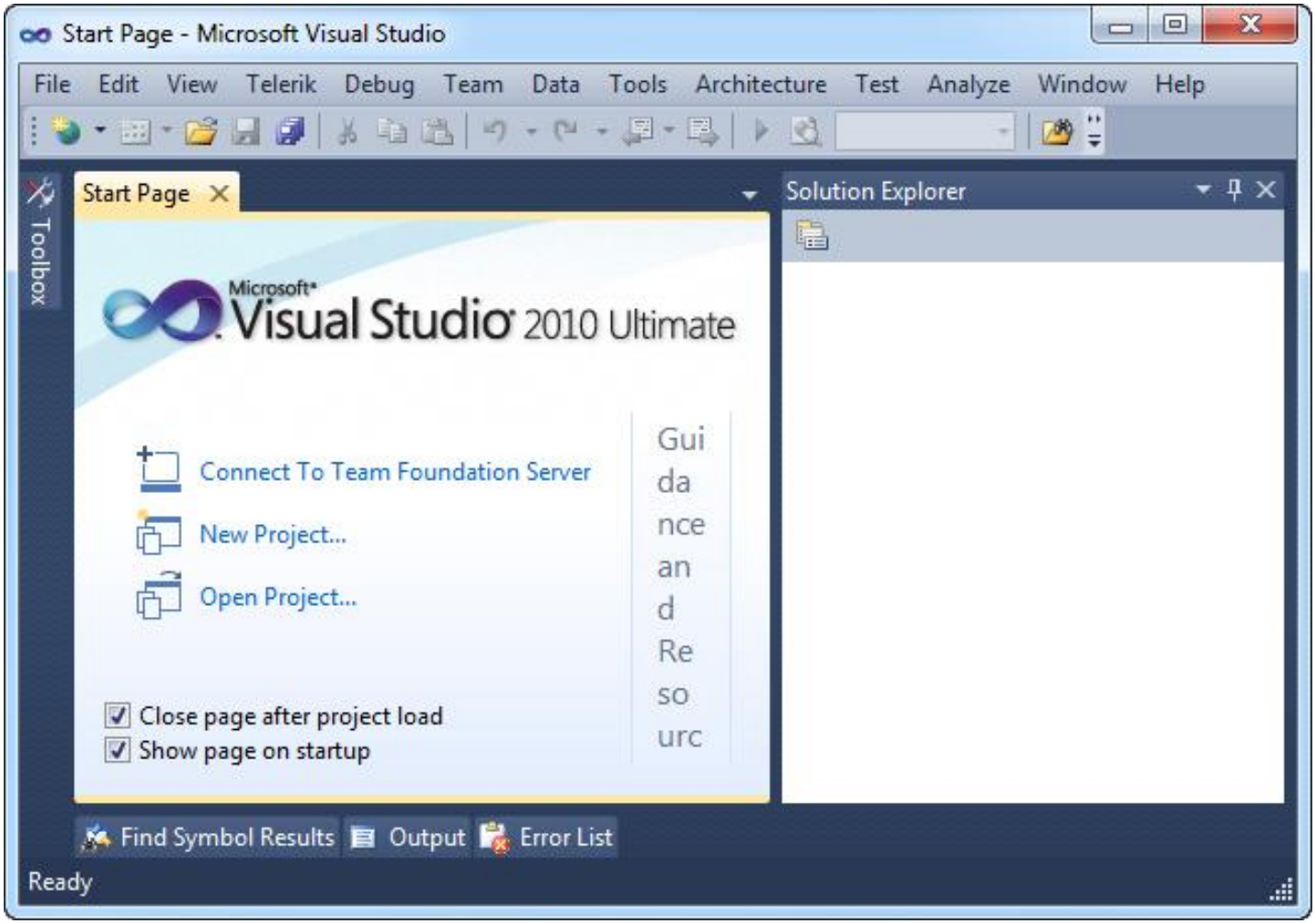
b.Tập khai báo tài nguyên:Tất cả các tài nguyên được mô tả tóm tắt trong tập tin có tên phụ là .RC

c. Tập tin đề án có dạng .dsw, *.sln

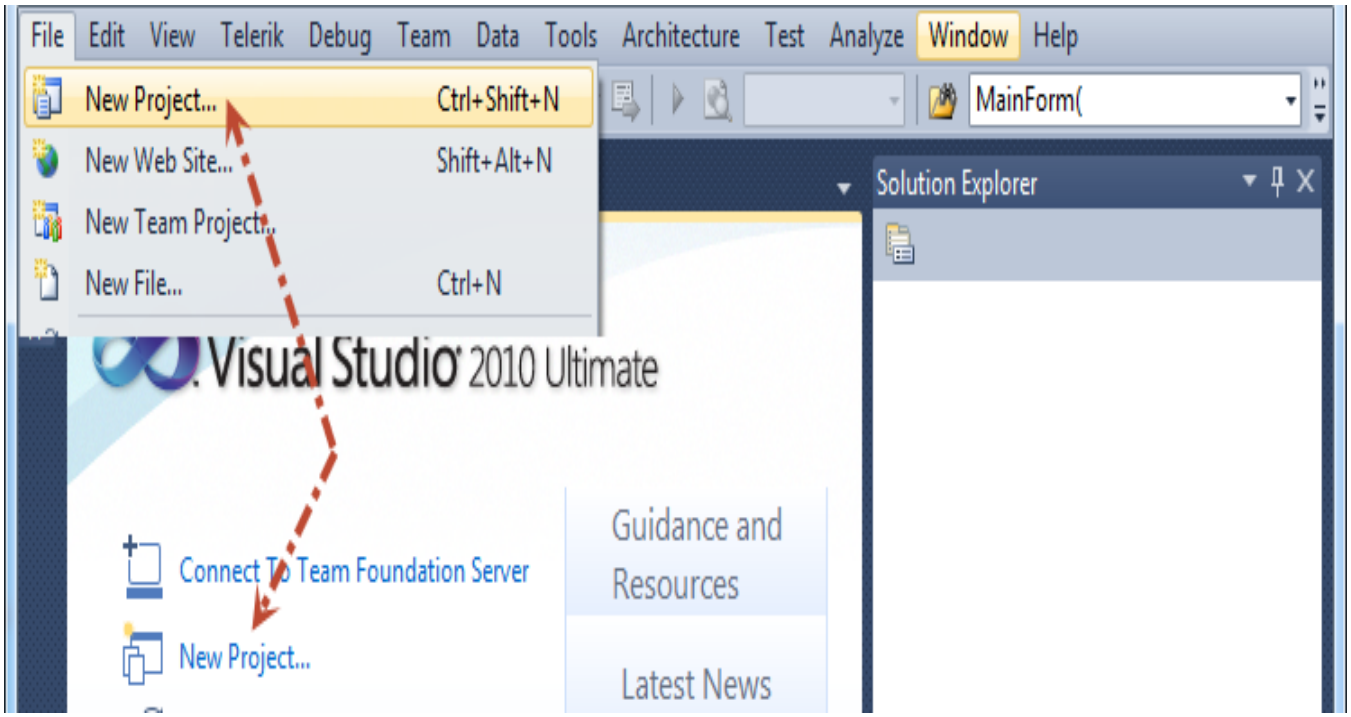
4.3.2 Tạo đề án trong Visual 2010

Bước 1: Mở Visual Studio bằng cách vào menu **Start** > **All Programs** > **Microsoft Visual Studio 2010** rồi chọn **Microsoft Visual Studio 2010** như trong hình sau:

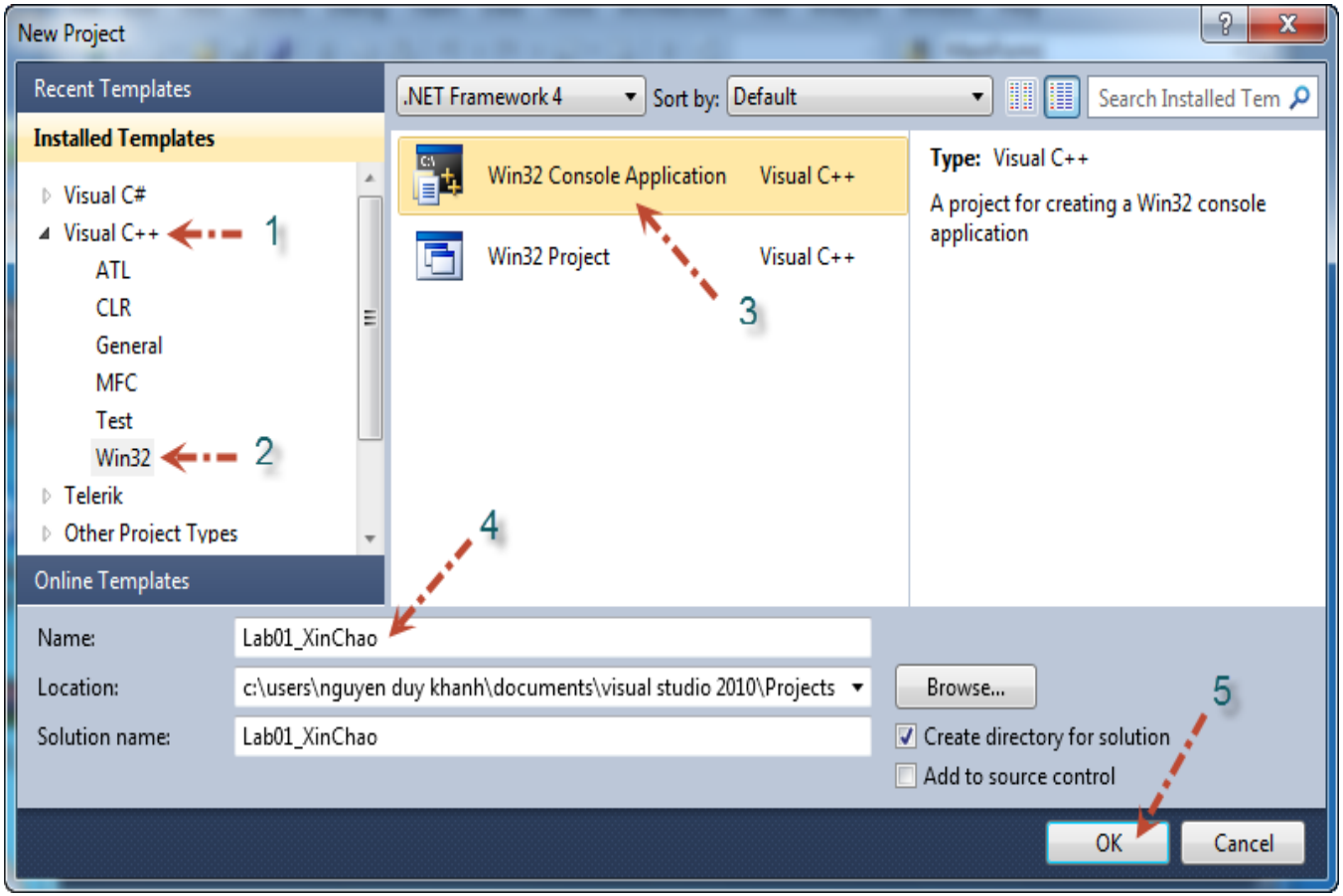



Start Page - Microsoft Visual Studio
File Edit View Telerik Debug Team Data Tools Architecture Test Analyze Window Help
Toolbox
Start Page
Solution Explorer
Microsoft Visual Studio 2010 Ultimate
Connect To Team Foundation Server
New Project...
Open Project...
Gui
da
nce
an
d
Re
so
urc
Close page after project load
Show page on startup
Find Symbol Results
Output
Error List
Ready

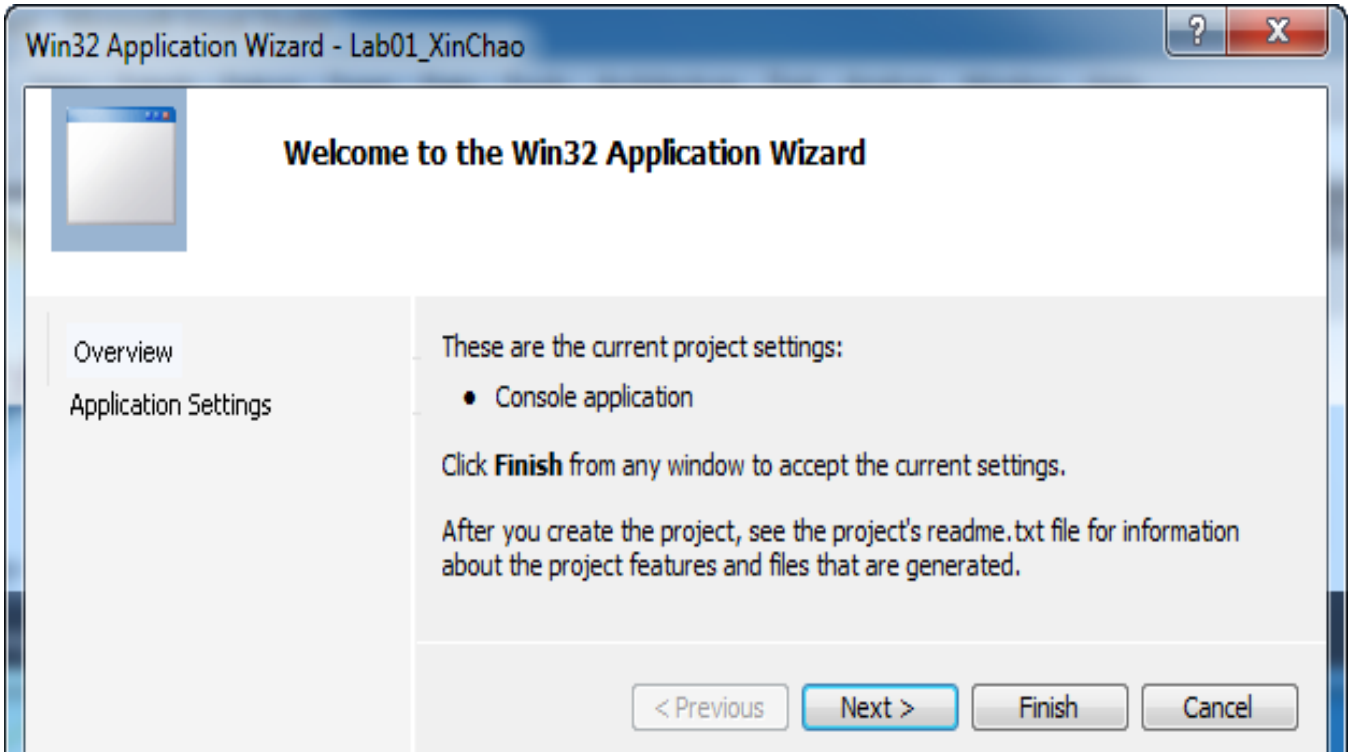
Bước 2: Trong cửa sổ Visual Studio, chọn **New Project**… hoặc vào menu **File** > **New Project** hoặc nhấn tổ hợp phím **Ctrl + Shift + N** để mở cửa sổ tạo dự án.



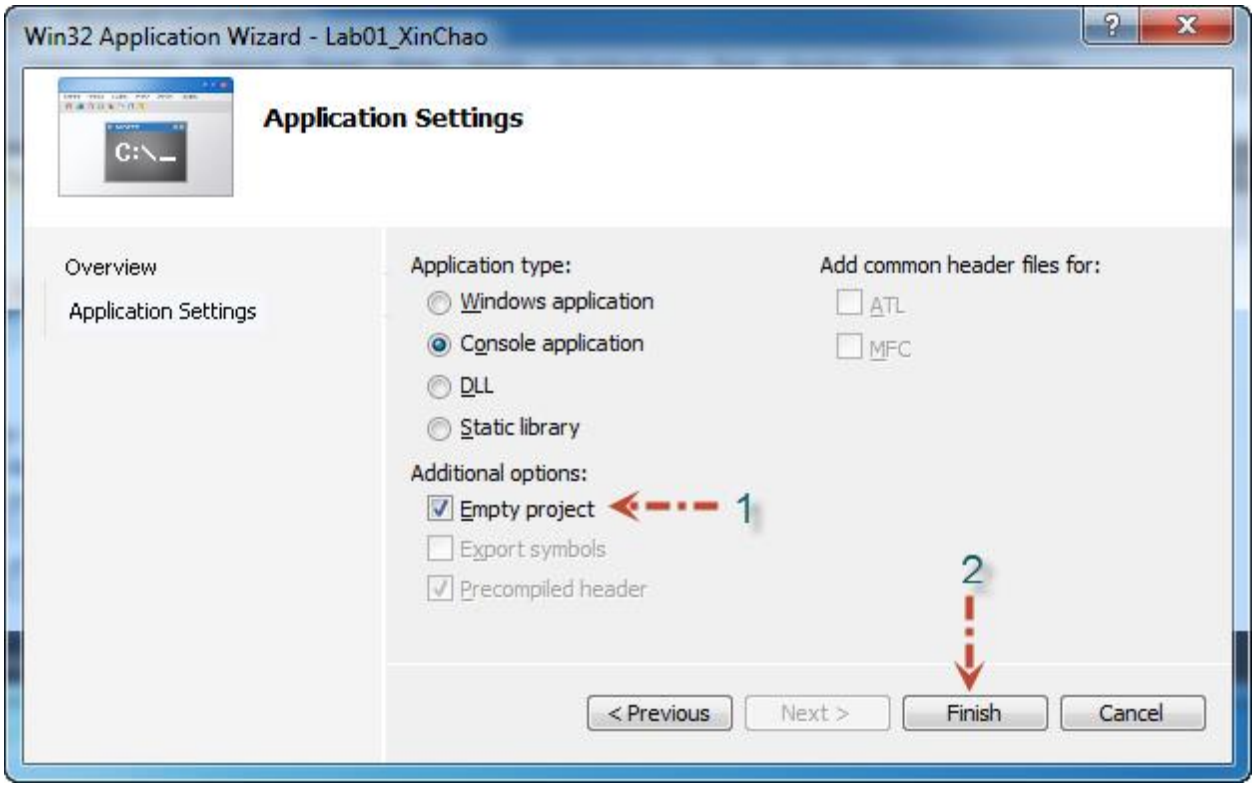
Bước 3:Trong cửa sổ mới, chọn kiểu dự án và **nhập tên của dự án** trong mục Name như hình sau:



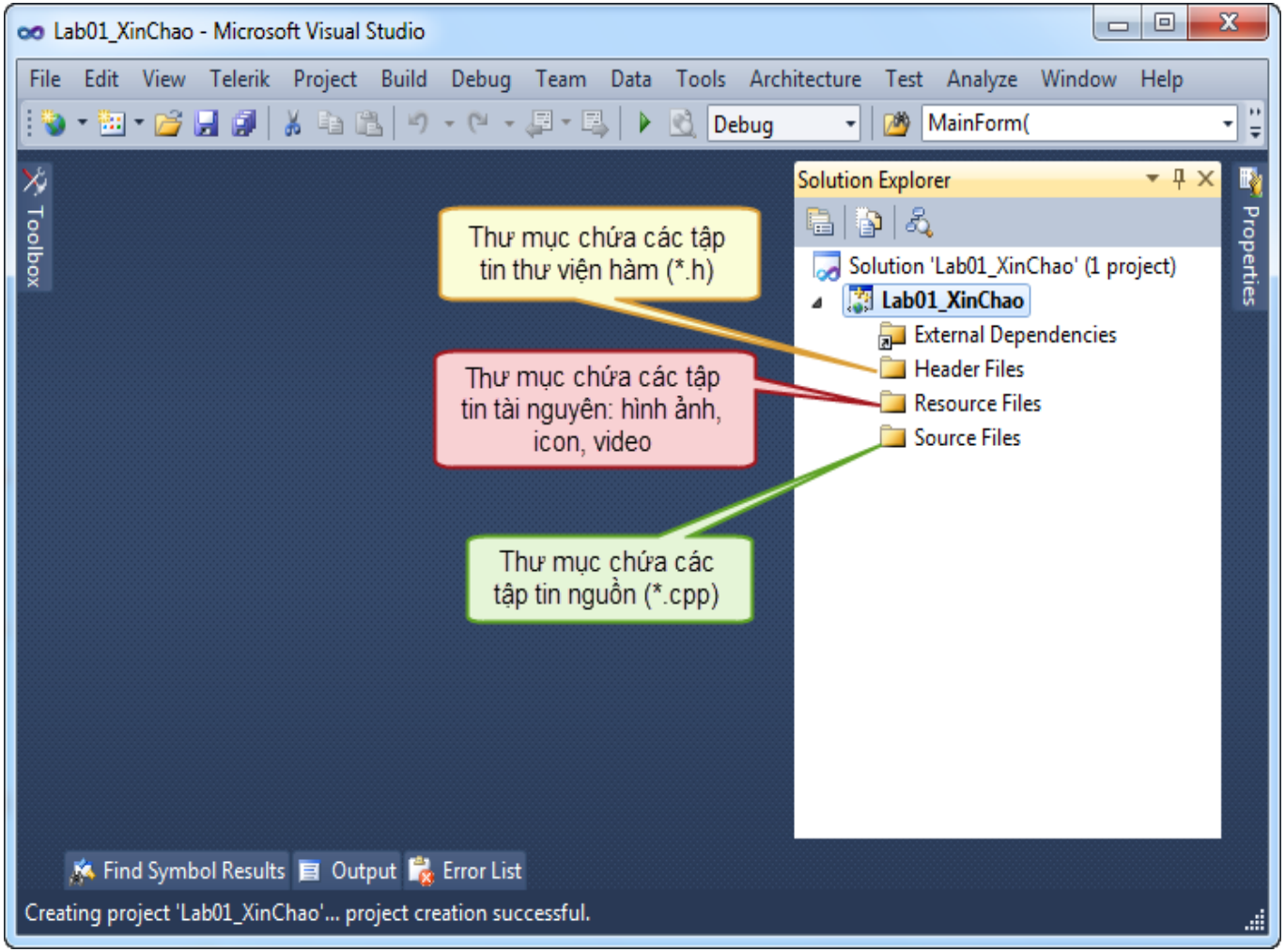
Bước 4:Trong cửa sổ tiếp theo, nhấn nút Next.



Bước 5:Tiếp theo, đánh dấu chọn mục Empty project để tạo một dự án rỗng. Nhấn nút Finish.



Bước 6: Nếu chương trình không hiển thị cửa sổ **Solution Explorer** như hình dưới đây, bạn hãy chọn menu **View** > **Solution Explorer**. Cửa sổ này hiển thị cấu trúc của một dự án C++.



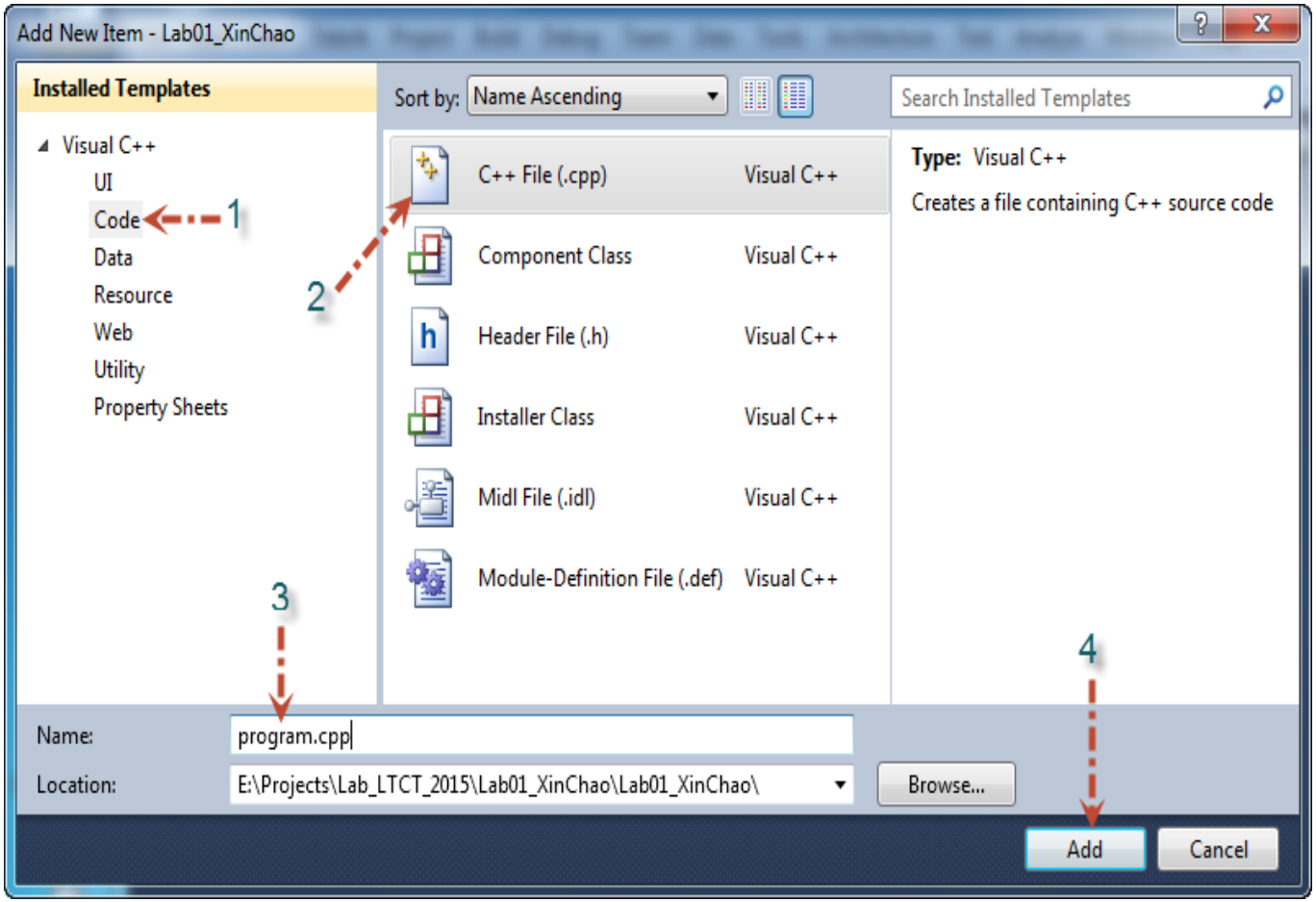
Bước 7:Tiếp theo, ta sẽ tạo một tập tin *.cpp để chứa mã nguồn của chương trình. Để tạo tập tin này, ta có thể thực hiện một trong các bước sau:

Nhắp phải chuột vào thư mục **Source Files**, chọn **Add** > **New Items**…

Chọn từ menu **Projects** > **Add New Items** …

Nhấn tổ hợp phím **Ctrl + Shift + A**

Bước 8:Trong cửa sổ Add New Item, chọn mục Code > C++ File (.cpp), nhập tên tập tin là program.cpp vào mục Name rồi nhấn Add.



Bước 10: Đến đây, dự án đã được tạo xong. Ta có thể viết mã lệnh cho chương trình đầu tiên. Trong phần tiếp theo, ta sẽ viết mã lệnh để xuất ra màn hình một dòng thông báo có nội dung: “Chào mừng các bạn đến với C++”.

4.3.3. Ví dụ:

- Viết chương trình tính diện tích tam giác khi biết chiều dài 3 cạnh tam giác.

  Ta tổ chức đề án theo 2 cách :

  – Cách 1: Đề án chỉ có 1 hàm main()

  – Cách 2 đề án được tổ chức theo nhiều hàm

Cách 1 (Project chỉ có một tập tin .cpp, trong tập tin này chỉ có hàm main()
- Tạo project co tên : DT_Tamgiac
- Thêm vào project tập tin Dttg.cpp trong Source files
//Soạn thảo chương trình trong tập tin Dttg.cpp :

```
#include <iostream>
#include <math.h>
#include <conio.h>
using namespace std;
int main()
{
	double a, b, c, p, s;
	cout << "\nNhap chieu dai 3 canh a, b, c:";
	cout << "\na = "; cin >> a;
	cout << "\nb = "; cin >> b;
	cout << "\nc = "; cin >> c;
	//Tinh nua chu vi
	p = (a + b + c) / 2;
	//Tinh dien tich tam giac
	s = sqrt(p*(p - a)*(p - b)*(p - c));
	cout << "\nDien tich tam giac S = " << s;
	_getch();
	return 1;
}
```

Cách 2: (Project chỉ có một tập tin .cpp, trong tập tin này có nhiều hàm
- Tạo project co tên : DT_Tamgiac
- Thêm vào project tập tin Dttg.cpp trong Source files
//trong tập tin Dttg.cpp này có hàm main() và hàm tính diênj tích tam giác

```
#include <iostream>
#include <math.h>
#include <conio.h>
using namespace std;
//khai báo nguyên mẫu
double  Tinh_DT_TG(double a, double b, double c);
int main()
{
	double a, b, c,  s;
	cout << "\nNhap chieu dai 3 canh a, b, c:";
	cout << "\na = "; cin >> a;
	cout << "\nb = "; cin >> b;
	cout << "\nc = "; cin >> c;
	//Gọi hàm tính diện tích
	s= Tinh_DT_TG(a,b,c);
	cout << "\nDien tich tam giac S = " << s;
	_getch();
	return 1;
}
```

```
//Hàm tính diện tích tam giác
double Tinh_DT_TG(double a, double b, double c)
{
	double s, p;
	p = (a+b+c)/2;
	s = sqrt(p*(p-a)*(p-b)*(p-c));
	return s;
}
```

Cách 2 có thể sửa lại như sau, viết thêm hàm chaychuongtrinh, và hàm main() gọi hàm này để thực hiện công việc

```
#include <iostream>
#include <math.h>
#include <conio.h>
using namespace std;
double DT_TG(double a, double b, double c);
void ChayChuongTrinh();
int main()
{
        ChayChuongTrinh();
        return 1;
}
```

```
//Ham chay chuong trinh viet nhu sau:
void ChayChuongTrinh()
{
    double a, b, c, s;
    cout << "\nNhap chieu dai 3 canh a, b, c:";
    cout << "\na = "; cin >> a;
    cout << "\nb = "; cin >> b;
    cout << "\nc = "; cin >> c;
    s = DT_TG(a, b, c);
    cout << "\nDien tich tam giac S = " << s;
    _getch();
}
```

4.4 Một số thư viện trong C++

Một trong những tính năng thuận tiện của C++ là cung cấp kèm theo một số lượng lớn các thư viện.

Thư viện chứa các hằng, các hàm thực hiện các thao tác chuyên biệt nào đó.

Muốn sử dụng thư viện, ta chỉ cần tham chiếu đến tập tin giao diện của nó gọi là tập tin tiêu đề (header), tập tin có đuôi dạng .h, bằng cách sử dụng chỉ thị :

#include <Ten_Tep>

Sau đây là một số tập tin thư viện thường sử dụng :

Tên tập tin tiêu đề	Hàm, Hằng	Công dụng
iostream	Chứa các đối tượng cin, cout, . . .	Nhập xuất chuẩn
iomanip	Chứa các định dạng kết xuất	

math.h	double log(double x)	Lnx
	double pow(double x, double y)	x^y
	double sqrt(double x)	$\sqrt{x}$
	double exp(double x)	E^x
	. . .	

Xem MSDN . . .