

TRƯỜNG CAO ĐẲNG KỸ THUẬT CAO THẮNG  
KHOA CÔNG NGHỆ THÔNG TIN



# TÀI LIỆU THỰC HÀNH NHẬP MÔN LẬP TRÌNH



(Lưu hành nội bộ)

TP. HỒ CHÍ MINH, 2020

# MỤC LỤC

BÀI THỰC HÀNH TUẦN 01 – BUỔI 1 GIỚI THIỆU VỀ LẬP TRÌNH.....	1
I. Giới thiệu về lập trình máy tính và tiến trình biên dịch .....	1
1. Giới thiệu về lập trình .....	1
2. Tiến trình biên dịch.....	2
II. Giới thiệu một số loại lỗi trong lập trình.....	4
1. Biên dịch chương trình với lỗi cú pháp .....	4
2. Chạy chương trình với lỗi thực thi.....	5
3. Làm việc với lỗi logic .....	6
BÀI THỰC HÀNH TUẦN 01 – BUỔI 02 GIỚI THIỆU NGÔN NGỮ LẬP TRÌNH C++ .....	8
I. Ngôn ngữ lập trình C++ .....	8
1. Cấu trúc một chương trình C++ .....	8
2. Một số toán tử toán học trong C++.....	8
II. Chuyển lưu đồ thuật toán sang chương trình .....	9
III. Bài tập: .....	11
BÀI THỰC HÀNH TUẦN 02 – BUỔI 01 LỆNH NHẬP, XUẤT VÀ PHÉP GÁN.....	15
I. Làm việc với lệnh nhập/ xuất.....	15
1. Câu lệnh nhập/ xuất (cin/cout).....	15
2. Biến và hằng số .....	16
II. Toán tử gán.....	16
1. Toán tử gán .....	16
2. Viết gọn toán tử gán.....	17
III. Bài tập: .....	17
BÀI THỰC HÀNH TUẦN 02 – BUỔI 02 CÁC BIỂU THỨC VÀ CHUYỂN ĐỔI KIỂU DỮ LIỆU .....	21
I. Biểu thức .....	21
1. Một số hàm toán học .....	21
2. Chuyển đổi biểu thức đại số sang C++ .....	21
3. Chuyển đổi kiểu dữ liệu .....	22

II. Toán tử điều kiện.....	22
III. Bài tập: .....	23
BÀI THỰC HÀNH TUẦN 03 – BUỔI 01 BÀI THỰC HÀNH TỔNG HỢP 01 .....	26
I. Mô hình lập trình tuần tự.....	26
II. Kiểu dữ liệu, toán tử và biểu thức .....	27
1. Kiểu dữ liệu.....	27
2. Toán tử .....	28
III. Bài tập: .....	28
BÀI THỰC HÀNH TUẦN 03 – BUỔI 02 CÂU LỆNH ĐIỀU KIỆN <i>if</i> .....	33
I. Các toán tử quan hệ .....	33
II. Toán tử logic.....	34
III. Câu lệnh <i>if</i> .....	34
1. Câu lệnh <i>if</i> .....	34
2. Câu lệnh <i>if...else</i> .....	36
IV. Bài tập: .....	38
BÀI THỰC HÀNH TUẦN 04 – BUỔI 01 CÂU LỆNH ĐIỀU KHIỂN <i>if</i> LỒNG NHAU .....	43
I. Câu lệnh <i>if</i> lồng .....	43
II. Sửa một số lỗi liên quan câu lệnh <i>if</i> lồng nhau .....	45
1. Lỗi cú pháp.....	45
2. Lỗi logic .....	45
III. Bài tập: .....	47
BÀI THỰC HÀNH TUẦN 04 – BUỔI 02 CÂU LỆNH ĐIỀU KHIỂN <i>switch</i> .....	52
I. Câu lệnh <i>switch-case</i> .....	52
II. Câu lệnh <i>switch-case</i> lồng nhau .....	54
III. Bài tập: .....	56
BÀI THỰC HÀNH TUẦN 05 – BUỔI 01 BÀI THỰC HÀNH TỔNG HỢP 02 .....	61
I. Câu lệnh <i>if/else</i> .....	61
II. Câu lệnh <i>switch-case</i> .....	62
III. Bài tập: .....	62
BÀI THỰC HÀNH TUẦN 05 – BUỔI 02 CÂU LỆNH LẶP <i>WHILE</i> .....	67

I. Toán tử tăng và giảm.....	67
II. Câu lệnh <i>while</i> .....	67
III. Các bộ đếm.....	69
IV. Giá trị lính canh.....	70
V. Bài tập:.....	71
BÀI THỰC HÀNH TUẦN 06 – BUỔI 01 CÂU LỆNH LẶP <i>DO-WHILE</i> .....	74
I. Câu lệnh lặp <i>do-while</i> .....	74
II. Chuyển đổi giữa câu lệnh <i>while</i> và <i>do-while</i> .....	75
III. Bài tập: .....	77
BÀI THỰC HÀNH TUẦN 06 – BUỔI 02 CÂU LỆNH LẶP <i>FOR</i> .....	80
I. Câu lệnh lặp <i>for</i> .....	80
II. Chuyển đổi giữa câu lệnh <i>while</i> và <i>for</i> .....	83
III. Bài tập: .....	84
BÀI THỰC HÀNH TUẦN 07 – BUỔI 01 CÁC CÂU LỆNH THOÁT KHỎI VÒNG LẶP.....	87
I. Các câu lệnh <i>break</i> và <i>continue</i> .....	87
1. Câu lệnh <i>break</i> .....	87
2. Câu lệnh <i>continue</i> .....	87
II. Kiểm tra dữ liệu.....	88
III. Bài tập: .....	89
BÀI THỰC HÀNH TUẦN 07 – BUỔI 02 CÂU LỆNH LẶP LỒNG NHAU .....	92
I. Câu lệnh lặp lồng nhau.....	92
II. Bài tập:.....	94
BÀI THỰC HÀNH TUẦN 08 – BUỔI 01 BÀI THỰC HÀNH TỔNG HỢP 03.....	99
I. Câu lệnh <i>while</i> .....	99
II. Câu lệnh <i>do-while</i> .....	99
III. Câu lệnh <i>for</i> .....	99
IV. Bài tập: .....	102
BÀI THỰC HÀNH TUẦN 08 – BUỔI 02 HÀM (FUNCTION) .....	104
I. Định nghĩa hàm .....	104
II. Truyền tham trị (Pass by value) .....	105

III. Truyền tham chiếu (pass by reference).....	106
IV. Bài tập: .....	107
BÀI THỰC HÀNH TUẦN 09 – BUỔI 01 HÀM (FUNCTION) .....	110
I. Phạm vi.....	110
II. Đối số mặc định.....	111
III. Hàm có giá trị trả về.....	113
IV. Bài tập: .....	116
BÀI THỰC HÀNH TUẦN 09 – BUỔI 02 ĐỆ QUY.....	118
I. Một số ví dụ.....	118
II. Bài tập:.....	120
BÀI THỰC HÀNH TUẦN 10 – BUỔI 01 BÀI THỰC HÀNH TỔNG HỢP 04.....	123
I. Khái niệm hàm .....	123
II. Phạm vi của biến .....	124
III. Các cách truyền tham số .....	124
IV. Bài tập: .....	124
BÀI THỰC HÀNH TUẦN 10 – BUỔI 02 MẢNG MỘT CHIỀU .....	126
I. Mảng một chiều.....	126
II. Khởi tạo mảng .....	127
III. Truyền tham số mảng cho hàm.....	127
IV. Bài tập: .....	129
BÀI THỰC HÀNH TUẦN 11 – BUỔI 01 CHUỖI – MẢNG KÝ TỰ .....	132
I. Hằng chuỗi.....	132
II. Lưu trữ chuỗi trong mảng.....	132
III. Các hàm thư viện xử lý chuỗi .....	133
IV. Bài tập: .....	136
BÀI THỰC HÀNH TUẦN 11 – BUỔI 02 CHUỖI – LỚP STRING TRONG C++ .....	139
I. Lớp string .....	139
II. Các hàm thành viên trong lớp string .....	140
III. Bài tập: .....	142
BÀI THỰC HÀNH TUẦN 12 – BUỔI 01 MẢNG HAI CHIỀU .....	144

I. Mảng hai chiều .....	144
II. Xử lý mảng hai chiều .....	144
III. Bài tập: .....	147
BÀI THỰC HÀNH TUẦN 12 – BUỔI 02 BÀI THỰC HÀNH TỔNG HỢP 05 .....	150
I. Mảng 1 chiều .....	150
II. Mảng 2 chiều .....	151
III. Chuỗi .....	152
IV. Bài tập: .....	152
BÀI THỰC HÀNH TUẦN 13 – BUỔI 01 KIỂU DỮ LIỆU CẤU TRÚC - STRUCT ..	156
I. Dữ liệu kiểu cấu trúc - struct .....	156
II. Truy xuất các thành viên cấu trúc .....	157
III. Khởi tạo cấu trúc .....	159
IV. Bài tập: .....	159
BÀI THỰC HÀNH TUẦN 13 – BUỔI 02 KIỂU DỮ LIỆU CẤU TRÚC - STRUCT ..	161
I. Mảng cấu trúc .....	161
II. Cấu trúc lồng nhau .....	161
III. Bài tập: .....	163
BÀI THỰC HÀNH TUẦN 14 – BUỔI 01 CON TRỎ - POINTER .....	164
I. Các biến con trỏ .....	164
II. Mảng và con trỏ .....	165
III. Các biến động .....	166
IV. Bài tập: .....	169
BÀI THỰC HÀNH TUẦN 14 – BUỔI 02 CON TRỎ - POINTER .....	171
I. Con trỏ tới kiểu dữ liệu cấu trúc .....	171
II. Cấu trúc tự trỏ .....	172
III. Bài tập: .....	174
BÀI THỰC HÀNH TUẦN 15 – BUỔI 01 TẬP TIN - FILE .....	175
I. Tập tin văn bản .....	175
II. Các thao tác tập tin .....	175
III. Truyền các tập tin như là các tham số cho hàm .....	178

IV. Tập tin nhị phân .....	179
V. Bài tập:.....	181
BÀI THỰC HÀNH TUẦN 15 – BUỔI 02 BÀI THỰC HÀNH TỔNG HỢP 06.....	182
I. Kiểu dữ liệu cấu trúc .....	182
II. Xử lý tập tin.....	182
III. Các thao tác tập tin.....	183
IV. Truyền các tập tin như là các tham số cho hàm.....	185
V. Tập tin nhị phân.....	185
VI. Bài tập: .....	185
PHỤ LỤC HƯỚNG DẪN SỬ DỤNG VISUAL STUDIO 2010 .....	187
I. Tạo project Visual C++ .....	187
II. Biên dịch và thực thi chương trình.....	189
III. Sửa một số lỗi lập trình.....	192
1. Lỗi cú pháp.....	192
2. Lỗi logic .....	193

# BÀI THỰC HÀNH TUẦN 01 – BUỔI 1

## GIỚI THIỆU VỀ LẬP TRÌNH

### ❖ Mục đích:

1. Giới thiệu về lập trình và tiến trình biên dịch
2. Tìm hiểu, nhận diện và sửa ba loại lỗi máy tính: lỗi cú pháp (syntax errors), lỗi thực thi (run time errors) và lỗi logic (logic errors).
3. Viết được mã lệnh và chạy được chương trình đơn giản từ đầu

### I. Giới thiệu về lập trình máy tính và tiến trình biên dịch

#### 1. Giới thiệu về lập trình

Một chương trình máy tính là một loạt các hướng dẫn được viết bằng một ngôn ngữ lập trình để thực hiện một tác vụ cụ thể. Thông thường khi các sinh viên mới bắt đầu học lập trình, họ chỉ tập trung vào mã của ngôn ngữ lập trình (language code). Tuy nhiên, một phần mềm chất lượng chỉ đạt được sau khi có một bản thiết kế cẩn thận, mà ở đó xác định đầy đủ dữ liệu (data), quy trình xử lý (process) và kết quả mong muốn. Vì lý do này, điều quan trọng là sinh viên cần học các kỹ thuật thiết kế tốt trước khi cố gắng tạo ra một chương trình chất lượng. Thiết kế được hướng dẫn bởi một thuật toán, đó là một kế hoạch giải quyết một số vấn đề (bài toán).

Bài toán ví dụ: Phát triển một thuật toán tính điểm trung bình của 5 bài thi.

Một thuật toán thường bắt đầu với phát biểu chung chung của bài toán.

Tính điểm trung bình 5 bài thi

Từ đây, chúng ta có thể tinh chỉnh phát biểu này bằng cách liệt kê các lệnh sẽ được thực hiện để đạt được mục tiêu.

Đọc các điểm

Tính điểm trung bình

Viết ra điểm trung bình

Mỗi hộp (được gọi là một nút) có thể hoặc không thể được tinh chỉnh thêm tùy thuộc vào độ rõ ràng của nó với người dùng. Ví dụ: Tác vụ **tính điểm trung bình**, đối



với lập trình viên có kinh nghiệm là nhiệm vụ cần hoàn thành để đạt được mục tiêu. Đối với sinh viên mới học là tính trung bình như thế nào hơn là hoàn thành mục tiêu như thế nào. Quy trình tinh chỉnh này tiếp tục cho đến khi chúng ta có một danh sách các bước dễ hiểu với người dùng để hoàn thành nhiệm vụ. Ví dụ, **Tính điểm trung bình** có thể được tinh chỉnh vào hai nút sau:

Tổng = tổng cộng 5 điểm

Điểm trung bình = Tổng/5

Bắt đầu từ trái sang phải, một nút không có tinh chỉnh sẽ trở thành một phần của thuật toán. Thuật toán thực sự (các bước để giải quyết chương trình trên) được liệt kê trong phần in đậm.

Tính trung bình của 5 điểm thi

**Đọc các điểm**

Tính điểm trung bình

**Tổng = tổng cộng 5 điểm**

**Điểm trung bình = Tổng/5**

**Viết ra điểm trung bình**

Từ thuật toán trên, một chương trình có thể được viết bằng C++.

## 2. Tiến trình biên dịch

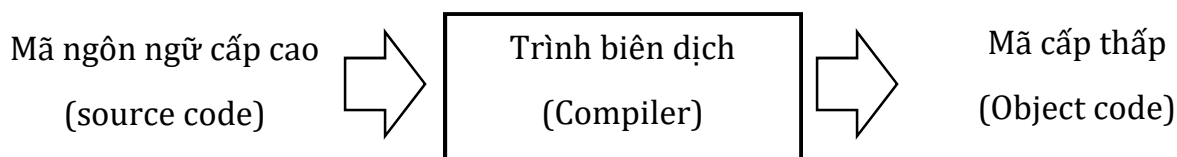
Máy vi tính chỉ có thể hiểu một chuỗi các số 1 và 0. Chuỗi giá trị dưới đây có thể vô nghĩa với chúng ta, trên thực tế, đó là cách máy tính đọc và thực hiện mọi thứ mà nó làm.

10010001111010101110010001110001000

Bởi vì máy tính chỉ sử dụng hai số (1 và 0), đây được gọi là **mã nhị phân**. Có thể tưởng tượng lập trình phức tạp như thế nào nếu chúng ta phải học ngôn ngữ rất phức tạp này. Thực tế, đó là cách lập trình được thực hiện cách đây nhiều năm. Tuy nhiên, ngày nay chúng ta may mắn có được **ngôn ngữ cấp cao** (high level language) như

C++. Những ngôn ngữ này được hướng nhiều hơn cho sự hiểu biết của con người và do đó nhiệm vụ lập trình trở nên dễ dàng hơn nhiều. Tuy nhiên, vì máy tính chỉ hiểu mã nhị phân cấp thấp (thường được gọi là mã máy), nên phải có quy trình dịch để chuyển đổi các ngôn ngữ cấp cao này thành mã máy. Điều này thường được thực hiện bởi một **trình biên dịch**, là một gói phần mềm dịch các ngôn ngữ cấp cao sang mã máy. Nếu không có nó, chúng tôi không thể chạy các chương trình của chúng tôi. Hình bên dưới minh họa vai trò của trình biên dịch.

Trình biên dịch dịch mã nguồn thành mã đối tượng. Loại mã thường được phản ánh trong tên tiện ích mở rộng của tệp nơi tệp được đặt.



Ví dụ: Chúng ta sẽ viết mã nguồn (source code – high level language) trong C++ và tất cả các tệp tin kết thúc với phần mở rộng .cpp, như sau:

*firstprogram.cpp*

*secondprogram.cpp*

Khi những chương trình này được biên dịch, một tệp tin mới (tệp tin đối tượng) sẽ được tạo ra và có phần mở rộng là .obj, như sau:

*firstprogram.obj*

*secondprogram.obj*

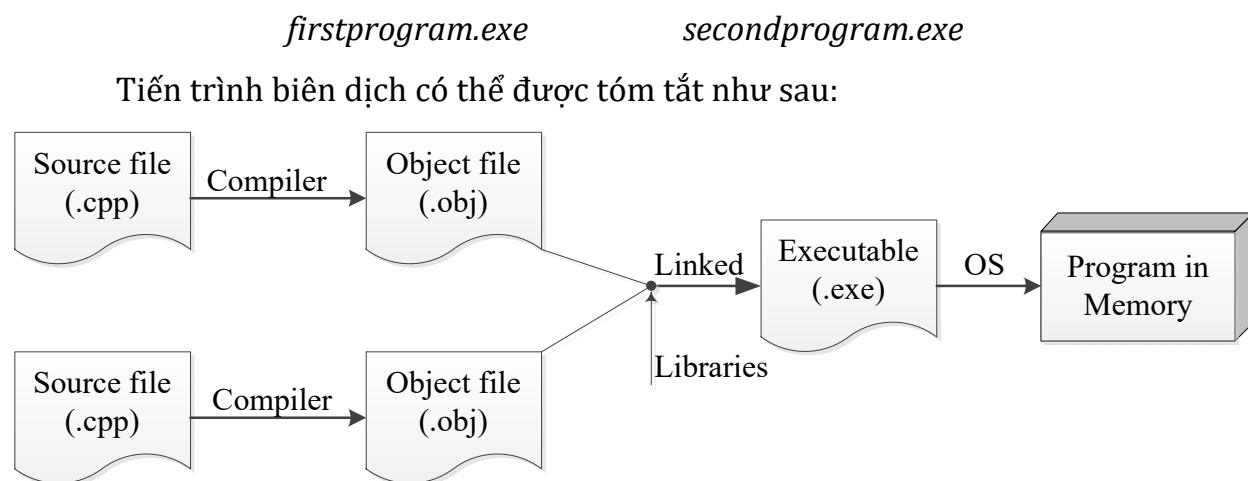
Trình biên dịch cũng bắt lỗi ngữ pháp được gọi là **lỗi cú pháp (syntax errors)** trong mã nguồn. Ví dụ, trong C++ các lệnh kết thúc bằng dấu chấm phẩy. Câu lệnh sau đây sẽ có lỗi cú pháp:

```
cout << "Hi there" << endl
```

Vì không có dấu chấm phẩy ở cuối, trình biên dịch sẽ chỉ ra một lỗi, mà phải được sửa lại như sau:

```
cout << "Hi there" << endl;
```

Sau khi quá trình biên dịch hoàn thành, máy tính phải thực hiện thêm việc **liên kết chương trình (linking)**. Trình liên kết (**linker**) kết hợp các mô-đun đối tượng được tạo bởi trình biên dịch từ các tập tin mã nguồn, thêm các mô-đun mã được yêu cầu từ thư viện chuẩn được cung cấp như là một phần của C++ và kết nối tất cả mọi thứ vào một tập tin thực thi thường có phần mở rộng là *.exe*. Trình liên kết cũng phát hiện và báo cáo lỗi; ví dụ: nếu một phần của chương trình của bạn bị thiếu hoặc thành phần thư viện được tham chiếu không tồn tại.



## II. Giới thiệu một số loại lỗi trong lập trình

### 1. Biên dịch chương trình với lỗi cú pháp

- Tạo một chương trình mới với code như sau:

---

```

//Day la chuong trinh minh hoa loi cu phap
//Them ten cua ban o day

//Khai bao thu vien
#include<iostream>
using namespace std;

//Ham chinh
void main()
{
    int So;
    float Tong;

    cout<<"Hom nay la mot ngay tuyet voi de thuc hanh"
  
```

---

---

```
    cout<<endl<<"Hay nhap mot so ban thich: "<<endl;
    cin>>So;
    Tong = So*2;
    cout<<Tong<<" la 2 lan so ban nhap."<<endl;
}
```

---

- Biên dịch chương trình bạn sẽ thấy thông báo lỗi.
- Sửa lỗi, biên dịch và chạy lại chương trình trên.

## 2. Chạy chương trình với lỗi thực thi

- Tạo một chương trình mới với code như sau:

---

```
//Day la chuong trinh nhan mot so va chia no cho 2

//Nhap ten ban o day

//Khai bao thu vien
#include<iostream>
using namespace std;

//Ham chinh
void main()
{
    int SoBiChia;
    int SoChia;
    float Thuong;

    cout<<"Nhap mot so va sau do nhan Enter"<<endl;
    cin >>SoBiChia;

    Thuong = SoBiChia/SoChia;

    cout<<"Ket qua: "<<Thuong<<endl;
}
```

---

- Biên dịch và chạy chương trình bạn sẽ thấy **không** có thông báo lỗi cú pháp. Ở đây có **lỗi thời điểm thực thi** chương trình. Do SoChia chưa được khởi tạo giá trị.
- Sửa lại chương trình trên bằng cách gán cho SoChia một giá trị là 2.
- Biên dịch và chạy lại chương trình. Nhập 9 khi chương trình yêu cầu nhập. Ghi nhận lại kết quả được xuất lên màn hình.

### 3. Làm việc với lỗi logic

- Tạo một chương trình mới với code như sau:

---

```
//Day la chuong trinh nhan hai gia tri tu nguoi dung
//va sau do hoan vi chung truooc khi xuat len man hinh

//Nhap ten ban o day

//Khai bao thu vien
#include<iostream>
using namespace std;

//Ham chinh
void main()
{
    float SoThuNhat;
    float SoThuHai;

    //Thong bao nguoi dung nhap so thu nhat
    cout<<"Nhap so thu nhat va sau do nhan Enter"<<endl;
    cin >>SoThuNhat;

    //Thong bao nguoi dung nhap so thu hai
    cout<<"Nhap so thu hai va sau do nhan Enter"<<endl;
    cin >>SoThuHai;

    //Hien thi gia tri nhap
    cout<<"Ban nhap cac so la "<<SoThuNhat<<" va "
        <<SoThuHai<<endl;

    //Hoan vi hai so
    SoThuNhat = SoThuHai;
    SoThuHai = SoThuNhat;

    //Xuat cac gia tri len man hinh
    cout<<"Sau khi hoan vi cac gia tri cua hai so la "
        <<SoThuNhat<<" va "<<SoThuHai<<endl;
}
```

---

- Biên dịch chương trình. Bạn sẽ **không** thấy lỗi cú pháp.
- Chạy chương trình. Những giá trị được xuất lên màn hình?
- Hãy chỉ ra lỗi của chương trình trên và đề xuất cách sửa lỗi.



# BÀI THỰC HÀNH TUẦN 01 – BUỔI 02

## GIỚI THIỆU NGÔN NGỮ LẬP TRÌNH C++

### ❖ Mục đích:

1. Giới thiệu ngắn gọn ngôn ngữ lập trình C++
2. Chuyển lưu đồ thuật toán sang chương trình C++
3. Minh họa sử dụng một số toán tử trong C++

### I. Ngôn ngữ lập trình C++

#### 1. Cấu trúc một chương trình C++

**Header  
(Global  
section)**

```
//Đây là chương trình minh họa
//một chương trình C++ đơn giản

//Nhập tên bạn ở đây

//Khái báo thu vien
#include<iostream>
using namespace std;
const double PI = 3.14; //Hàng số
```

**Hàm  
chính  
(main)**

```
//Hàm chính
void main()
{
    //Khái báo biến và khởi tạo giá trị
    float fBanKinh = 4.0;

    cout<<"PI = "<<PI<<endl;
    cout<<"Ban Kinh = "<<fBanKinh<<endl;
    cout<<"Chu vi = "<<2*PI*fBanKinh<<endl;
}
```

#### 2. Một số toán tử toán học trong C++

Trong lập trình có các toán tử toán học truyền thống như sau:

Phép toán	Ký hiệu toán học	Ký hiệu C++
Cộng	+	+
Trừ	-	-
Nhân	×	*
Chia	÷	/
Chia lấy dư	mod	%

---

```
//Đây là chương trình minh họa các toán tử trong C++
//Thêm tên của bạn ở đây

//Khai báo thư viện
#include<iostream>

using namespace std;

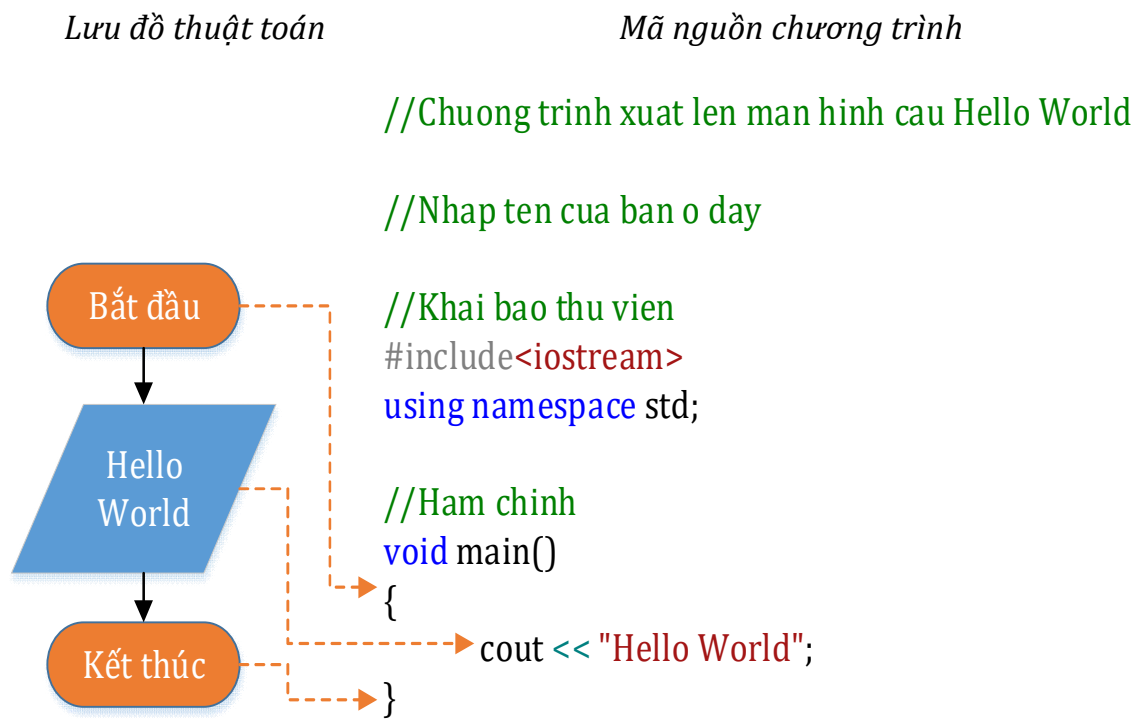
//Hàm chính
void main()
{
    int iSoBiChia = 9;
    int iSoChia = 2;
    int iSoDu;
    int iThuong;

    iThuong = iSoBiChia/iSoChia;
    iSoDu = iSoBiChia%iSoChia;
    cout<<"Thuong = "<<iThuong<<endl;
    cout<<"So du = "<<iSoDu<<endl;
}
```

---

## II. Chuyển lưu đồ thuật toán sang chương trình

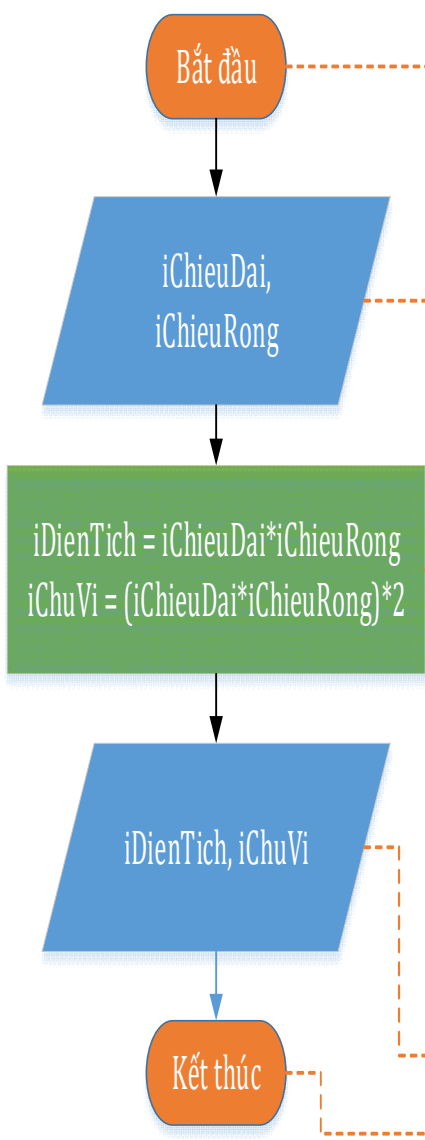
**Ví dụ 1:** *Viết chương trình in ra màn hình câu "Hello world".*





**Ví dụ 2: Viết chương trình nhập vào chiều dài và chiều rộng của hình chữ nhật. Tính và in ra màn hình chu vi và diện tích của hình chữ nhật.**

Lưu đồ thuật toán



Mã nguồn chương trình

```

//Chương trình tính diện tích và chu vi hình chữ nhật

//Nhập tên của bạn ở đây

//Khai báo thư viện
#include<iostream>
using namespace std;

//Hàm chính
void main()
{
    //Khai báo biến
    int iChieuDai, iChieuRong;
    int iDienTich, iChuVi;

    cout<<"CHƯƠNG TRÌNH TÍNH DIỆN TÍCH, CHU VI HÌNH CHỮ NHẬT"<<endl;
    cout<<"===== "<<endl;
    //Nhập dữ liệu
    cout<<"Hay nhập chiều dài: "<<endl;
    cin>>iChieuDai;
    cout<<"Hay nhập chiều rộng: " << endl;
    cin>>iChieuRong;

    //Tính toán
    iDienTich = iChieuDai * iChieuRong;
    iChuVi = (iChieuDai + iChieuRong) * 2;

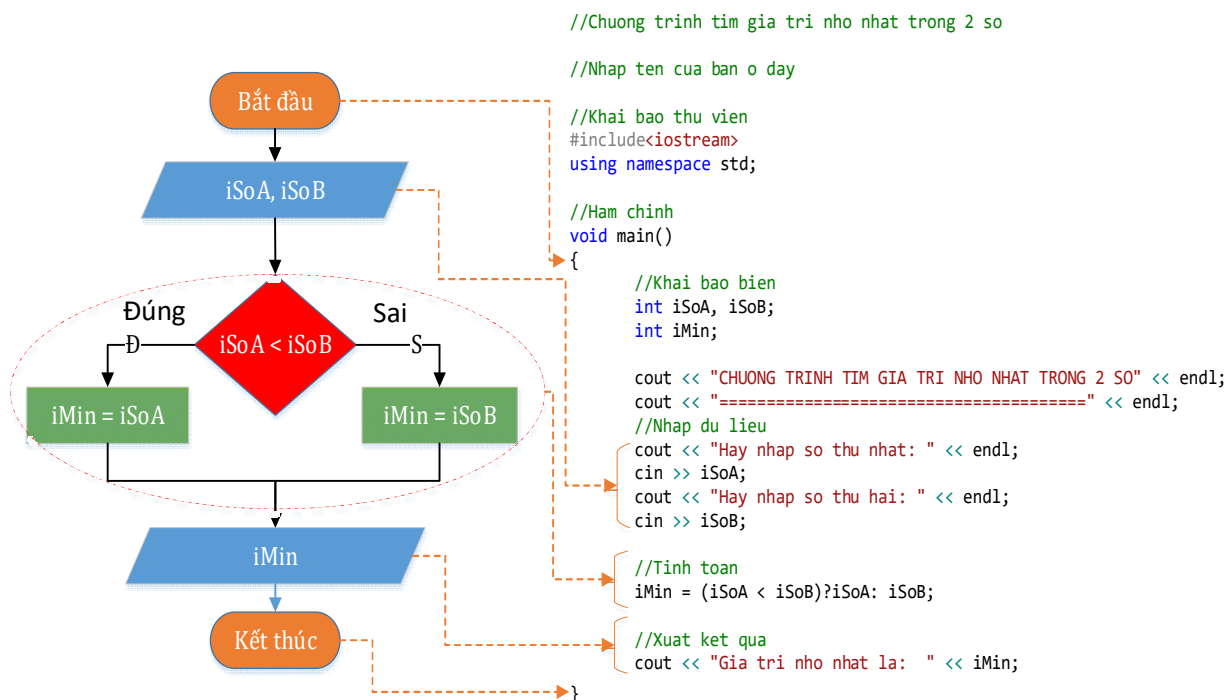
    //Xuất kết quả
    cout<<"Diện tích: " << iDienTich << endl;
    cout<<"Chu vi: " <<iChuVi;
}
  
```

The code is a C++ program that calculates the area and perimeter of a rectangle. It includes comments in Vietnamese. The program starts with a main function that declares variables for length, width, area, and perimeter. It then prompts the user to input the length and width. After calculating the area and perimeter, it outputs the results. The flowchart on the left maps the steps of the code to its graphical representation.

**Ví dụ 3: Viết chương trình nhập vào 2 số nguyên, tìm và in số nhỏ nhất trong 2 số ra màn hình.**

Lưu đồ thuật toán

Mã nguồn chương trình



### III. Bài tập:

Yêu cầu: Chuyển lưu đồ thuật toán sang chương trình C++.

**Bài 1: (Xuất bài thơ)** Viết chương trình xuất lên màn hình đoạn thơ (không dấu), được định dạng như sau:

THƠ DUYÊN  
Xuân Diệu  
Chiều mong hoa thơ trên nhánh duyên,  
Cây me riu rít cặp chim truyện.  
Đổ trời xanh ngọc qua muôn lá,  
Thu đến nơi nơi động tiếng huyền.  
Con đường nhỏ nhỏ, gió xiêu xiêu,  
Lả lả cành hoang nắng trở chiều.  
Buổi ấy lòng ta nghe ý bạn,  
Lần đầu rung động nỗi thương yêu.  
Em bước điềm nhiên không vương chân,  
Anh đi lững dưng chẳng theo gần.  
Vô tâm nhưng giữa bài thơ dịu,  
Anh với em như một cặp vần.

Lưu đồ:

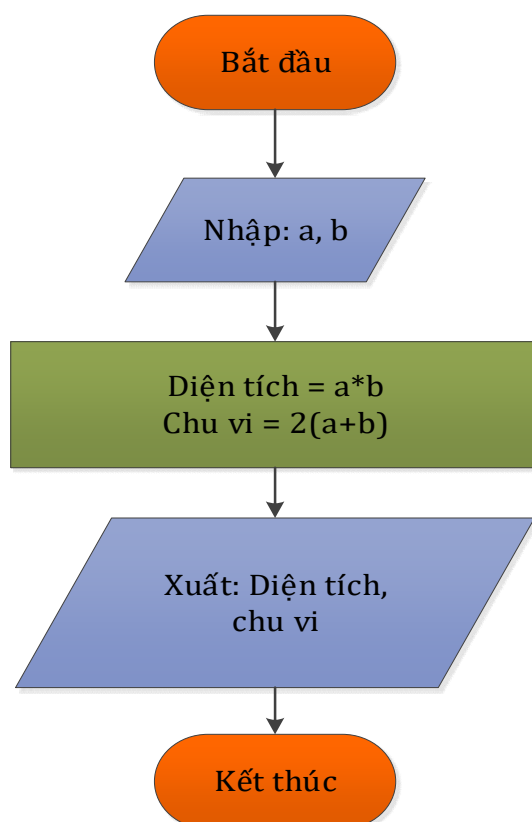
- Input: Không có
- Output: Đoạn thơ



**Bài 2: (Tính diện tích và chu vi thửa ruộng)** Giả sử nhà bạn có một thửa ruộng hình chữ nhật có chiều dài là  $a$ , chiều rộng là  $b$ . Bạn hãy mô tả thuật toán để tính diện tích và chu vi của thửa ruộng.

Lưu đồ:

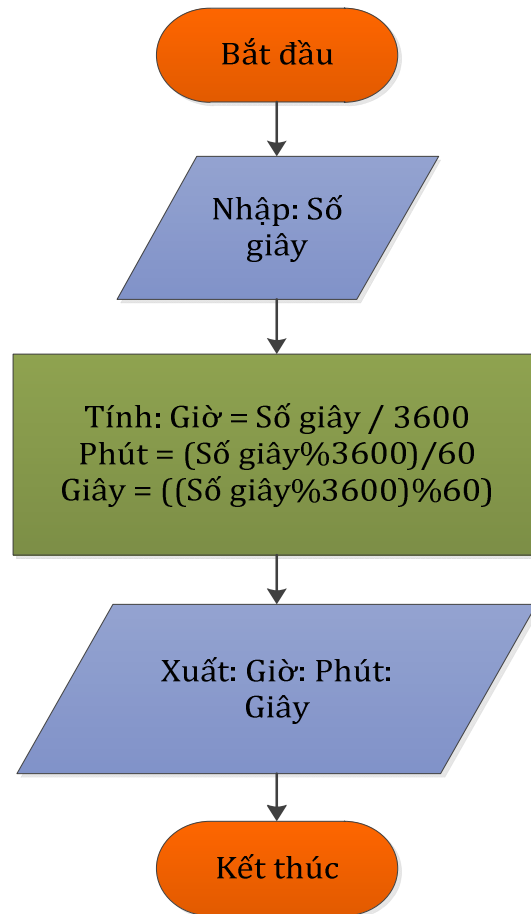
- Input:  $a, b \in \mathbb{R}$
- Output: Diện tích và chu vi hình chữ nhật



**Bài 3: (Đổi số giây sang giờ, phút, giây)** Viết chương trình nhập vào tổng số giây là một số nguyên dương. Hãy đổi tổng số giây sang số giờ, số phút và số giây (còn dư). Xuất kết quả lên màn hình theo định dạng giờ:phút:giây

Lưu đồ:

- Input: Số giây  $\in \mathbb{N}$
- Output: Giờ, phút, giây tương ứng



**Bài 4:** (Xác định giá trị hàng trăm, chục, đơn vị) Viết chương trình nhập vào một số nguyên dương. Hãy xác định và xuất lên màn hình các giá trị thuộc hàng trăm, hàng chục và hàng đơn vị.

# BÀI THỰC HÀNH TUẦN 02 – BUỔI 01

## LỆNH NHẬP, XUẤT VÀ PHÉP GÁN

### ❖ Mục đích:

1. Học câu lệnh nhập và định dạng xuất
2. Giới thiệu biến và hằng số
3. Chỉ ra việc sử dụng bộ nhớ trong lập trình
4. Giới thiệu các kiểu dữ liệu khác nhau
5. Giới thiệu phép gán

### I. Làm việc với lệnh nhập/ xuất

#### 1. Câu lệnh nhập/ xuất (cin/cout)

```
//Chương trình minh họa định dạng xuất
//Thêm tên của bạn ở đây
//Khai báo thu viên
#include <iostream> // để sử dụng cout <<
#include <iomanip> // để sử dụng các định dạng
using namespace std;
void main()
{
    cout << "TINH DOANH THU" << endl << "=====" << endl;
    //khai báo biến
    int iSoLuong;
    float fpDonGia;
    float fpThanhTien;
    //Nhập dữ liệu
    cout << "Hay nhap so luong, ENTER: " << endl;
    cin >> iSoLuong;
    cout << "Hay nhap don gia, ENTER: " << endl;
    cin >> fpDonGia;
    //Tính toán
    fpThanhTien = iSoLuong * fpDonGia;
    //Xuất kết quả
    cout << setprecision(2) << fixed;
    cout << "So luong: " << setw(20) << iSoLuong << endl;
    cout << "Don gia: " << setw(20) << fpDonGia << endl;
    cout << "Thanh tien: " << setw(15) << fpThanhTien << endl;
}
```

## 2. Biến và hằng số

---

```
//Chương trình minh họa biến và hằng số
//Thêm tên của bạn ở đây
//Khai báo thư viện
#include <iostream>
#include <iomanip> // để sử dụng các định dạng
using namespace std;
//Định nghĩa hằng số cách 1
#define HangSoF 33.8
void main()
{
    cout << "CHƯƠNG TRÌNH ĐỔI NHIỆT ĐỘ C SANG ĐỘ F, ĐỘ K" <<
endl
    << "=====" << endl;
    //khai báo biến
    float fpDoC;
    float fpDoF = 0, fpDoK = 0;
    //Định nghĩa hằng số cách 2
    const double HangSoK = 273.15;

    //Nhập dữ liệu
    cout << "Hay nhập nhiệt độ C, ENTER: " << endl;
    cin >> fpDoC;
    //Tính toán
    fpDoK = fpDoC + HangSoK;
    fpDoF = fpDoC + HangSoF;
    //Xuất kết quả
    cout << setprecision(2) << fixed;
    cout << "Nhiệt độ C: " << setw(20) << fpDoC << endl;
    cout << "Nhiệt độ F: " << setw(20) << fpDoF << endl;
    cout << "Nhiệt độ K: " << setw(20) << fpDoK << endl;
}
```

---

## II. Toán tử gán

### 1. Toán tử gán

Ví dụ:

---

```
int count;
int total;
total = 10; //10 được lưu vào vùng nhớ có tên là total
count = 3 + 4; //Phía bên phải được tính là 7
                // count được gán giá trị của 7
total = total + count; //Phía bên phải được tính là 10 + 7
                // và 17 được đưa vào vùng nhớ có tên là total
```

---

Chú ý rằng, `total` đã được khởi tạo giá trị ban đầu là 10, ở dòng lệnh cuối cùng giá trị này bị thay đổi thành 17.

## 2. Viết gọn toán tử gán

Một phép gán dạng `x = x @ a`; có thể được viết ngắn gọn thành `x @= a`; trong đó `x` là một biến, `@` là các phép toán toán học và `a` là một biểu thức hoặc hằng số hoặc biến. Cách viết này có nhiều thuận lợi khi viết hoặc đọc chương trình nhất là khi tên biến quá dài.

Ví dụ:

---

```
count = count + 3;
count += 3;
total = total + count;
total += count;
```

---

### Các toán tử gán tắt:

Ký hiệu	Phép toán	Ví dụ	Kết quả
<code>+=</code>	Gán cộng (Addition assignment)	<code>i += 3</code>	<code>i = i + 3</code>
<code>-=</code>	Gán trừ (Subtraction assignment)	<code>i -= 3</code>	<code>i = i - 3</code>
<code>*=</code>	Gán nhân (Multiplication assignment)	<code>i *= 3</code>	<code>i = i * 3</code>
<code>/=</code>	Gán chia (Division assignment)	<code>i /= 3</code>	<code>i = i / 3</code>
<code>%=</code>	Gán chia dư (Remainder assignment)	<code>i %= 3</code>	<code>i = i % 3</code>

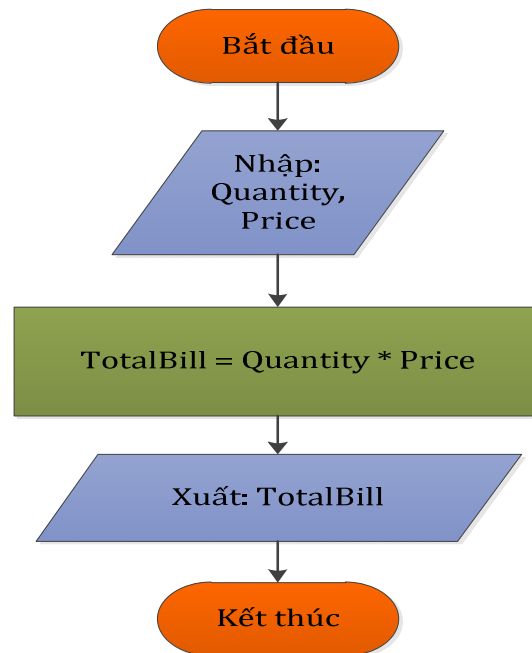
## III. Bài tập:

**Bài 1: (Tính tổng trị giá của hóa đơn)** Viết chương trình cho phép người dùng nhập vào số lượng và đơn giá của một sản phẩm khách hàng mua. Sau đó, xuất lên màn hình trị giá hóa đơn của khách hàng.

Lưu đồ:

- Input: Số lượng (Quantity), Đơn giá (Price)
- Output: Tổng trị giá (TotalBill)





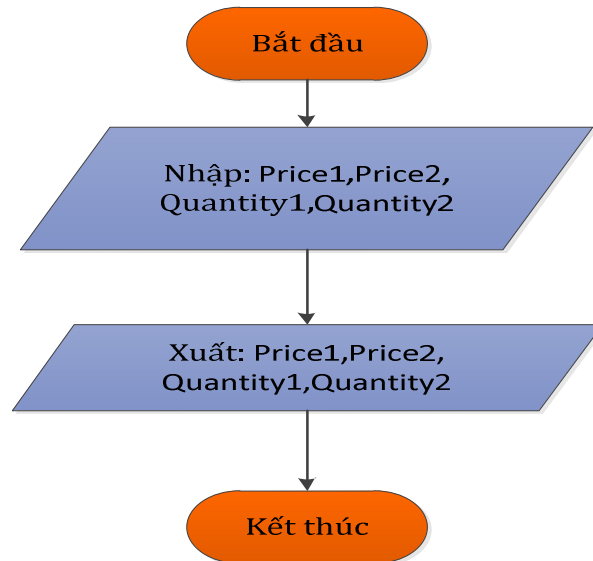
**Bài 2: (Định dạng xuất)** Hãy viết chương trình cho phép người dùng nhập vào đơn giá của hai sản phẩm bất kỳ và số lượng của mỗi sản phẩm. Xuất lên màn hình theo định dạng sau: đơn giá cách lề trái 15 khoảng trắng, số lượng cách điểm kết thúc của đơn giá 12 khoảng trắng.

```
CHUONG TRINH MINH HOA DINH DANG XUAT
=====
Hay nhap so luong 1:
8
Hay nhap don gia 1:
1.95
Hay nhap so luong 2:
9
Hay nhap don gia 2:
10.89

          PRICE    QUANTITY
          1.95      8
          10.89     9
```

Lưu đồ:

- Input: Đơn giá 1 (Price1), đơn giá 2 (Price2), số lượng 1 (Quantity1), số lượng 2 (Quantity2)
- Output: các đơn giá và số lượng theo định dạng



**Bài 3: (Tính điểm trung bình)** Viết chương trình cho phép người dùng nhập vào 3 điểm số từ bàn phím và xuất lên màn hình điểm trung bình (lấy chính xác đến 2 số thập phân) của những điểm vừa nhập.

*Chạy chương trình với mẫu sau:*

**Hay nhap vao diem thu nhât:**

9.7

**Hay nhap vao diem thu hai:**

9.8

**Hay nhap vao diem thu ba:**

9.5

**Diem trung binh: 9.67**

**Bài 4: (Tính tổng doanh thu)** Công ty đồ gỗ Woody bán ba loại ghế sau:

Kiểu (Style)	Giá (Price Per Chair)
Thuộc địa Mỹ	\$85.00
Hiện đại	\$57.50
Pháp cổ điển	\$127.75

Viết chương trình nhập vào số lượng bán mỗi loại. Tính tổng số tiền mỗi loại và tổng doanh thu của tất cả các loại. Sau đó, xuất lên màn hình lấy chính xác đến hai số thập phân.

*Hãy chạy chương trình với mẫu sau:*

**Hay nhập vào số ghe đã bán loại Thuộc địa Mỹ:**

**20**

**Hay nhập vào số ghe đã bán loại Hien dai:**

**15**

**Hay nhập vào số ghe đã bán loại Phap co dien:**

**5**

**Tong doanh thu loại Thuoc dia My: \$1700.00**

**Tong doanh thu loại Hien dai: \$862.50**

**Tong doanh thu loại Phap co dien: \$638.75**

**Tong doanh thu tat ca cac loai: \$3201.25**

**Bài 5: (Tính thuế phải đóng)** Viết chương trình nhập vào tổng doanh số bán hàng (doanh thu cộng thuế) mà một doanh nghiệp có được trong một tháng cụ thể. Chương trình cũng nhận về phần trăm thuế bán hàng. Hãy xuất lên màn hình tổng doanh thu cộng với thuế phải đóng với độ chính xác 2 số thập phân.

*Hãy chạy chương trình với các mẫu sau:*

**Hay nhập tổng doanh thu trong tháng:**

**1080**

**Hay nhập vào thuế theo dạng thập phân (0.02 cho 2%):**

**0.06**

**Tong doanh thu trong thang: \$1080.00**

**Thuê phải đóng trong tháng: \$64.80**

## BÀI THỰC HÀNH TUẦN 02 – BUỔI 02

### CÁC BIỂU THỨC VÀ CHUYỂN ĐỔI KIỂU DỮ LIỆU

#### ❖ Mục đích:

1. Giới thiệu về biểu thức
2. Học cách chuyển đổi kiểu dữ liệu
3. Làm việc với hằng số và các hàm toán học
4. Làm việc với toán tử điều kiện

#### I. Biểu thức

##### 1. Một số hàm toán học

C++ cung cấp một số hàm toán học được định nghĩa trong thư viện **cmath**. Để sử dụng các hàm toán học này, ta cần khai báo thư viện **#include<cmath>**.

Hàm	Mô tả	Ví dụ	Kết quả
<code>pow(x, y)</code>	Tính giá trị $x^y$	$N = \text{pow}(2, 3)$	$N = 8$
<code>sqrt(x)</code>	Tính căn bậc 2 của $x$	$N = \text{sqrt}(4)$	$N = 2$
<code>cbrt(x)</code>	Tính căn bậc 3 của $x$	$N = \text{cbrt}(8)$	$N = 2$
<code>abs(x)</code>	Trị tuyệt đối của số nguyên $x$	$N = \text{abs}(-3)$	$N = 3$
<code>fabs(x)</code>	Trị tuyệt đối của số thực $x$	$N = \text{fabs}(-5.6)$	$N = 5.6$
<code>ceil(x)</code>	Làm tròn lên của $x$	$N = \text{ceil}(-8.8)$	$N = -8.0$
<code>floor(x)</code>	Làm tròn xuống của $x$	$N = \text{floor}(-8.8)$	$N = -9.0$

##### 2. Chuyển đổi biểu thức đại số sang C++

Một trong những thách thức của việc học một ngôn ngữ lập trình mới đó là việc chuyển đổi các biểu thức đại số sang mã lệnh tương ứng.

Ví dụ:  $4y(3-2)y + 7$

Biểu thức đại số này được triển khai như thế nào trong C++?

$4 * y * (3-2) * y + 7$

Hãy xem biểu thức sau:

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad (1)$$

Vậy (1) sẽ được viết thành:

$$x_1 = (-b + \text{sqrt}(\text{pow}(b, 2) - (4 * a * b))) / (2 * a)$$

$$x_2 = (-b - \text{sqrt}(\text{pow}(b, 2) - (4 * a * b))) / (2 * a)$$

### 3. Chuyển đổi kiểu dữ liệu

Ví dụ:

---

```
int iSoSVDau = 10;
int iTongSoSV=50;
float fPhanTram;
fPhanTram = float (iSoSVDau)/iTongSoSV;
```

---

## II. Toán tử điều kiện

**Cú pháp:** *expr1 ? expr2 : expr3*

Biểu thức *expr1* được tính trước. Nếu nó cho **giá trị khác 0 (đúng)**, thì biểu thức *expr2* được tính, và đó là giá trị của cả biểu thức. Ngược lại, biểu thức *expr3* được tính, và đó là giá trị của cả biểu thức. **Chỉ một biểu thức *expr2* hoặc *expr3* được tính.**

Ví dụ: Tính f(x) theo x với công thức sau:

$$f(x) = \begin{cases} 3x + \sqrt{x}, & x > 0 \\ e^x + 4, & x \leq 0 \end{cases}$$

---

```
//Day la chuong trinh minh hoa toan tu dieu kien:
//Chuong trinh tinh f(x) theo x

//Nhap ten ban o day

//Khai bao thu vien
#include<iostream>
#include<cmath>
using namespace std;

//Ham chinh
```

---

---

```

void main()
{
    float fX;
    float fFx;

    cout<<"Hay nhap gia tri cua x: "<<endl;
    cin>>x;
    fFx = (fX>0)? 3*fX + sqrt(fX) : exp(fX) + 4;
    cout<<"Gia tri f(x) = "<<fFx<<endl;
}

```

---

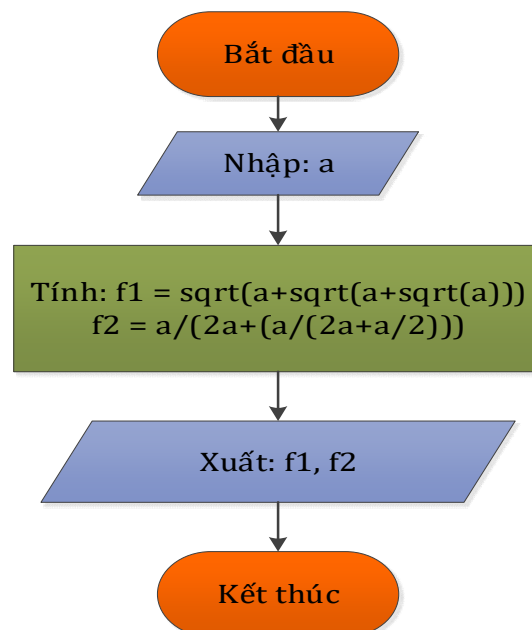
### III. Bài tập:

**Bài 1: (Tính giá trị các biểu thức)** Viết chương trình cho phép người dùng nhập vào một số nguyên  $a$ . Tính và xuất kết quả lên màn hình giá trị của các biểu thức theo  $a$  với độ chính xác 2 số thập phân:

$$f_1 = \sqrt{a + \sqrt{a + \sqrt{a}}} \qquad f_2 = \frac{a}{2a + \frac{a}{2a + \frac{a}{2}}}$$

Lưu đồ:

- Input:  $a \in \mathbb{N}$
- Output:  $f_1, f_2$



**Bài 2: (Tính chu vi, diện tích, bán kính đường tròn nội, ngoại tiếp)** Hãy viết chương trình cho phép người dùng nhập vào chiều dài ba cạnh của một tam giác (không cần kiểm tra điều kiện). Tính chu vi, diện tích, bán kính đường tròn nội tiếp, ngoại tiếp theo công thức:

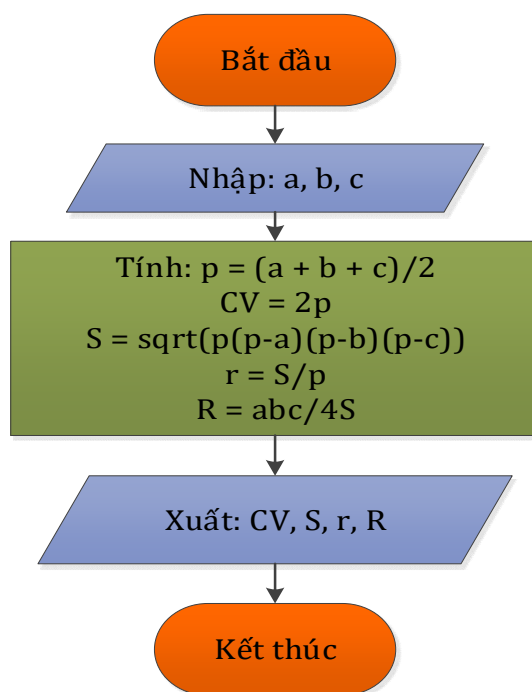
$$C = 2p = (a + b + c) \quad S = \sqrt{p(p-a)(p-b)(p-c)}$$

$$r = \frac{S}{p} \quad R = \frac{abc}{4S}$$

Xuất kết quả lên màn hình với độ chính xác 3 số lẻ.

Lưu đồ:

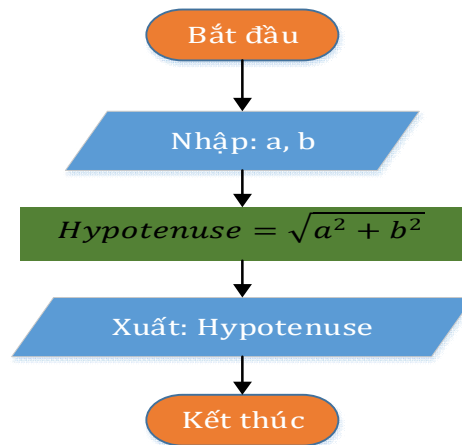
- Input:  $a, b, c \in \mathbb{R}$
- Output:  $CV, S, r, R$



**Bài 3: (Tính chiều dài cạnh huyền)** Viết chương trình cho phép người dùng nhập vào chiều dài hai cạnh của một tam giác vuông. Tính và xuất lên màn hình chiều dài cạnh huyền của tam giác đó.

Lưu đồ:

- Input:  $a, b \in \mathbb{R}$
- Output: Chiều dài cạnh huyền (Hypotenuse)

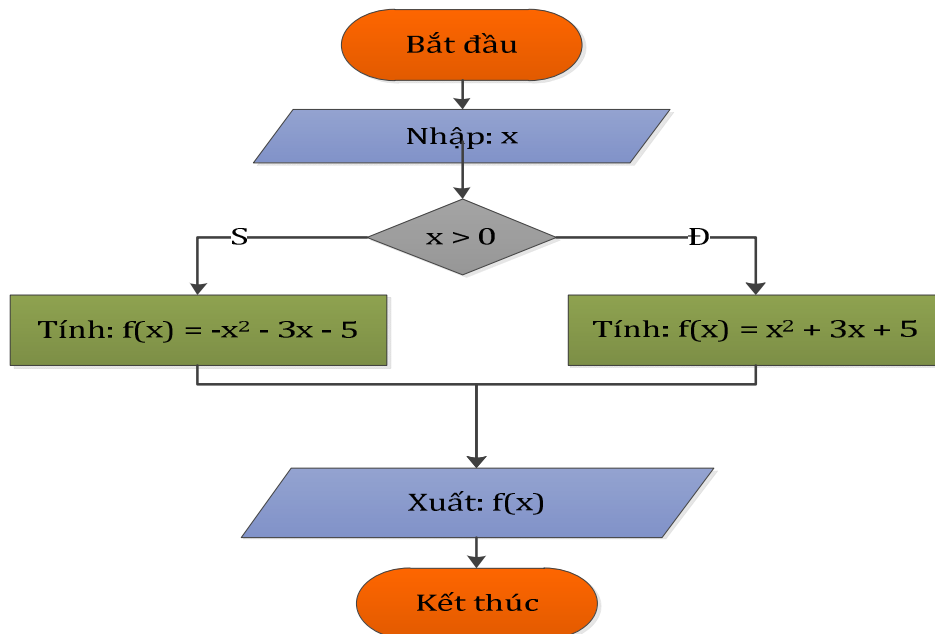


**Bài 4: (Tính giá trị biểu thức)** Viết chương trình cho phép người dùng nhập vào số thực  $x$ . Sử dụng toán tử điều kiện để tính  $f(x)$  theo giá trị  $x$  với công thức sau:

$$f(x) = \begin{cases} x^2 + 3x + 5, & x > 0 \\ -x^2 - 3x - 5, & x \leq 0 \end{cases}$$

Lưu đồ:

- Input:  $x \in \mathbb{R}$
- Output:  $f(x)$





## BÀI THỰC HÀNH TUẦN 03 – BUỔI 01

### BÀI THỰC HÀNH TỔNG HỢP 01

#### ❖ Mục đích:

1. Ôn tập mô hình lập trình tuần tự
2. Ôn lại kiến thức về kiểu dữ liệu, các toán tử và các biểu thức
3. Ôn tập cách sửa một số lỗi liên quan
4. Vận dụng cài đặt các bài toán liên quan trên máy tính.

#### I. Mô hình lập trình tuần tự

Xét một chương trình C++ sau: Nhập vào một số nguyên N và một số thực X. Tính và in ra màn hình:

$$P = (X^2 + 1)^N$$

---

```
/* Thông tin chương trình và tác giả
   Đây là chương trình minh họa mô hình tuần tự
   Họ tên: Nhập tên bạn ở đây
   MSSV:
   Lớp:
   Ngày:
   */

//Khai báo thư viện
#include<iostream>
#include<cmath>
using namespace std;

//Hàm chính
void main()
{
    //Khai báo biến
    int iN;
    float fpX;
    float fpP;

    //Nhập dữ liệu
    cout<<"Hay nhập số nguyên: "<<endl;
    cin>>iN;
```

---

---

```

cout<<"Hay nhap mot so thuc: "<<endl;
cin>>fpX;

//Tinh toan
fpP = pow( (pow(fpX, 2) + 1), iN);

//Xuat ket qua
cout<<"P = " <<fpP<<endl;
}

```

---

Một chương trình C++ bao gồm các thành phần: khai báo thư viện và hàm main. Phần khai báo thư viện hay vùng toàn cục (global section) chứa các khai báo thư viện cần thiết và các ghi chú mô tả mục đích của chương trình cũng như thông tin về lập trình viên. Mỗi chương trình C++ có **một và chỉ một hàm main** cho biết điểm bắt đầu thực thi chương trình. Mỗi hàm main phải bắt đầu với dấu ngoặc nhọn trái { và kết thúc với dấu ngoặc nhọn phải }. Các câu lệnh bên trong các dấu ngoặc này được thực thi **tuần tự từ trên xuống**.

## II. Kiểu dữ liệu, toán tử và biểu thức

### 1. Kiểu dữ liệu

C++ cung cấp một số kiểu dữ liệu cơ sở như sau:

Kiểu dữ liệu		Từ khóa
Số nguyên	Có dấu	int short long
	Không dấu	unsigned int unsigned short unsigned long
Số thực		float double
Ký tự		char
Chuỗi		string
Luận lý		bool

## 2. Toán tử

Trong C++ cung cấp sẵn một số toán tử toán học cơ bản. Các toán tử này có độ ưu tiên thực hiện khác nhau. Thứ tự ưu tiên được liệt kê như bảng dưới đây, các toán tử có độ ưu tiên từ trên xuống.

Toán tử	Độ ưu tiên
<code>() [] -&gt;</code>	Trái sang phải
<code>! ++ -- - + * (cast) &amp; sizeof</code>	Phải sang trái
<code>* / %</code>	Trái sang phải
<code>+ -</code>	Trái sang phải
<code>&lt;&lt; &gt;&gt;</code>	Trái sang phải
<code>&lt; &lt;= &gt; &gt;=</code>	Trái sang phải
<code>== !=</code>	Trái sang phải
<code>&amp;</code>	Trái sang phải
<code> </code>	Trái sang phải
<code>^</code>	Trái sang phải
<code>&amp;&amp;</code>	Trái sang phải
<code>  </code>	Trái sang phải
<code>?:</code>	Phải sang trái
<code>= += -= *= /= %=</code>	Phải sang trái
<code>,</code>	Phải sang trái

### III. Bài tập:

**Bài 1: (Tìm lỗi)** Hãy viết lại chính xác chương trình dưới đây.

---

```

/* Thông tin chương trình và tác giả
Đây là chương trình minh họa lỗi cú pháp
Họ tên: Nhập tên bạn ở đây
MSSV:
Lớp:
Ngày:
*/

```

---

---

```
//Khai bao thu vien
#include<iostream>
using namespace std;

//Ham chinh
void Main()

    //Khai bao bien
    int iSoNguyen;

    //Nhap du lieu
    cout<<"Hay nhap so nguyen: "<<endl;
    cin>>iSoNguyen;

    //Tinh toan
    iDonVi = iSoNguyen % 10;
    iHangChuc = (iSoNguyen/10)%10;
    iHangTram = iSoNguyen/100;

    int iSum = iDonVi + iHangChuc + iHangTram;

    //Xuat ket qua
    cout<<"Tong cac chu so = "<<iSum<<endl;
}
```

---

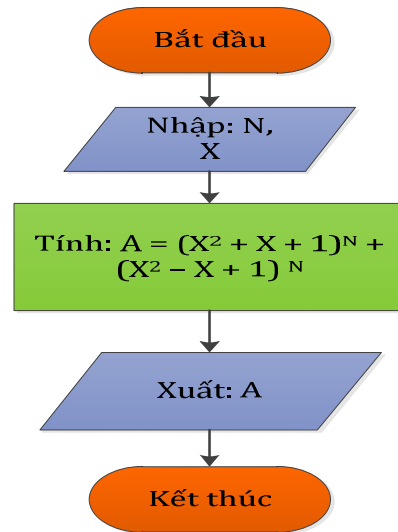
Biên dịch chương trình và chụp lại màn hình thông báo lỗi (Error List). Hãy sửa tất cả các lỗi phát sinh ở trên.

**Bài 2:** *(Tính giá trị biểu thức)* Nhập vào một số nguyên N và một số thực x. Tính và in ra biểu thức sau:

$$A = (X^2 + X + 1)^N + (X^2 - X + 1)^N$$

Lưu đồ:

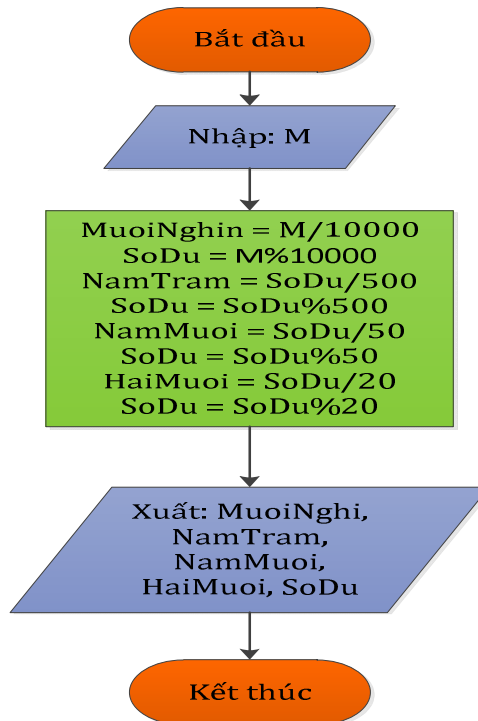
- Input:  $N \in \mathbb{Z}, X \in \mathbb{R}$
- Output: A theo công thức  $A = (X^2 + X + 1)^N + (X^2 - X + 1)^N$



**Bài 3: (Đổi tiền)** Nhập vào một số tiền nguyên dương M. Đổi số tiền này ra các tờ giấy bạc 10.000đ, 500đ, 50đ, 20đ và 1đ. Với giả thiết ưu tiên cho tờ có mệnh giá lớn hơn, hãy in ra màn hình số tờ mỗi loại.

Lưu đồ:

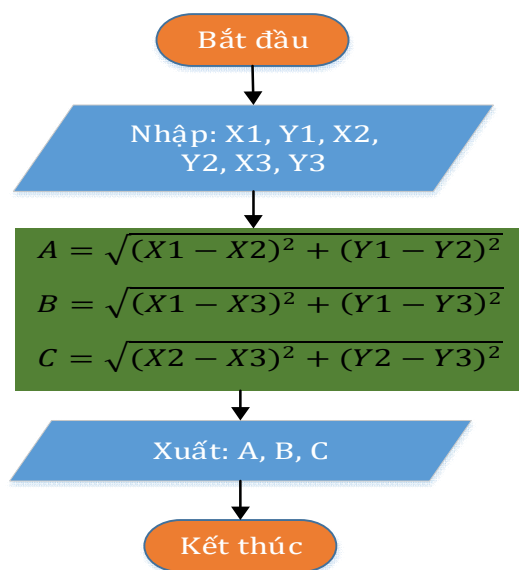
- Input:  $M \in \mathbb{N}$
- Output: Số lượng tiền mỗi mệnh giá



**Bài 4:** Nhập vào tọa độ 3 điểm trong không gian hai chiều. Tính chiều dài 3 cạnh tương ứng với 3 tọa độ trên.

Lưu đồ:

- Input:  $X_1, Y_1, X_2, Y_2, X_3, Y_3 \in \mathbb{R}$
- Output: Độ dài ba cạnh



**Bài 5:** Nhập vào tọa độ 2 điểm trong không gian hai chiều. Tính và xuất lên màn hình tọa độ trung điểm của hai điểm vừa nhập.

**Bài 6: Body Mass Index** hay thường được gọi viết tắt là **BMI** (chỉ số cân nặng, chỉ số khối cơ thể) được dùng để tính tỷ lệ giữa cân nặng và chiều cao của ai đó. Công thức tính BMI như sau:

$$BMI = \frac{\text{Cân nặng (Kg)}}{\text{Chiều cao}^2 (m^2)}$$

Viết chương trình cho phép người dùng nhập vào cân nặng và chiều cao của một người. Tính và xuất lên màn hình BMI của người đó.

**Bài 7:** Vận tốc của một vật được tính theo công thức sau:

$$v = \frac{x_2 - x_1}{t_2 - t_1}$$

Trong đó:

$x_2$ : vị trí cuối cùng  $x_1$ : vị trí ban đầu

$t_2$ : thời điểm cuối  $t_1$ : thời điểm ban đầu

Viết chương trình cho phép người dùng nhập vào vị trí ban đầu, vị trí cuối, thời điểm ban đầu, thời điểm cuối của một vật (Các giá trị nhập là các số nguyên). Tính và xuất lên màn hình vận tốc của vật đó.

## BÀI THỰC HÀNH TUẦN 03 – BUỔI 02

### CÂU LỆNH ĐIỀU KIỆN *if*

#### ❖ Mục đích:

1. Làm việc với các toán tử quan hệ
2. Làm việc với câu lệnh *if*
3. Học và sử dụng các toán tử logic

#### I. Các toán tử quan hệ

C++ cho phép các lập trình viên so sánh các giá trị số bằng các toán tử quan hệ.

Ký hiệu	Ý nghĩa
>	Lớn hơn
<	Nhỏ hơn
>=	Lớn hơn hoặc bằng
<=	Nhỏ hơn hoặc bằng
==	Bằng
!=	Khác

Mỗi kết quả của các toán tử này có kiểu số nguyên. Kết quả của mỗi toán tử là **1** nếu phép so sánh là đúng (*true*) và **0** nếu phép so sánh là sai (*false*).

Hãy xem xét đoạn lệnh sau:

```
int years;  
years = 6; //Cau lenh gan, years duoc gan gia tri 6  
years == 5; //Bieu thuc quan he, khong phai cau lenh gan  
years = years - 1; //Cau lenh gan  
years == 5; //Bieu thuc quan he
```

Bởi vì một toán tử quan hệ tạo ra một kết quả luận lý, nên có thể lưu kết quả này vào một biến kiểu bool. Ví dụ:

```
bool result = 5 < 4; //result sẽ là false
```



## II. Toán tử logic

Với việc sử dụng các toán tử quan hệ C++, các lập trình viên có thể tạo ra các biểu thức quan hệ. Các lập trình viên cũng có thể kết hợp các giá trị chân trị vào một biểu thức duy nhất bằng cách sử dụng **các toán tử logic**.

Phép toán	Ký hiệu C++
AND	&&
OR	
NOT	!

Ví dụ:

```
if (age > 12 && age < 20 && savings > 5000)
    cout<<"Ban la mot nguoi tuoi teen giau co."<<endl;
if (a < 10 || b > c || c > 50)
    cout<<"Co it nhat mot dieu kien dung."<<endl;
```

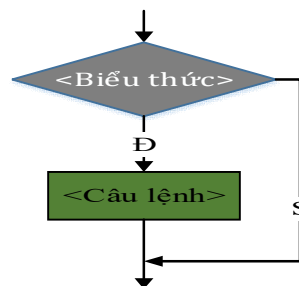
## III. Câu lệnh if

### 1. Câu lệnh if

Cú pháp:

```
if (<Biểu thức>)
    <Câu lệnh>;
```

Lưu đồ:

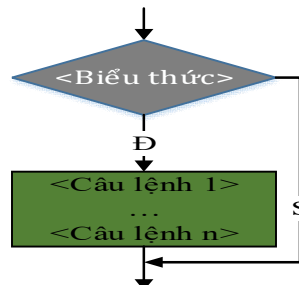


Hoặc:

Cú pháp:

```
if (<Biểu thức>)
{
    <Câu lệnh 1>;
    ...
    <Câu lệnh n>;
}
```

Lưu đồ:



**Ví dụ 3.1:** Viết chương trình nhập vào 2 số nguyên. Hãy cho biết giá trị nhỏ nhất của hai số vừa nhập.

---

```
/*
Chương trình tìm giá trị nhỏ nhất của 2 số
Thêm tên của bạn ở đây
*/
//Khai báo thư viện
#include<iostream>
using namespace std;

//Hàm chính
void main()
{
    cout<<"CHƯƠNG TRÌNH TÌM GIÁ TRỊ NHỎ NHẤT HAI SỐ"<< endl;
    int iNum1, iNum2;
    int imin;

    cout << "Hãy nhập một số nguyên: " << endl;
    cin >> iNum1;
    cout << "Hãy nhập một số nguyên: " << endl;
    cin >> iNum2;

    imin = iNum1;
    if (imin < iNum2)
        imin = iNum2;

    cout << "num1 = " <<iNum1<<" và num2 = " << iNum2 << endl;
    cout << "Giá trị nhỏ nhất của hai số là: "<<imin << endl;
}
```

---

**Ví dụ 3.2:** Viết chương trình nhập vào 2 số nguyên. Hãy hoán vị giá trị của số thứ nhất và số thứ hai nếu hai số có giá trị khác nhau.

---

```
/*
Chương trình hoán vị 2 số
Thêm tên của bạn ở đây
*/
//Khai báo thư viện
#include<iostream>
using namespace std;
//Hàm chính
void main()
{
    cout<<"CHƯƠNG TRÌNH HOÁN VỊ HAI SỐ"<< endl;
```

---

```

int iNum1, iNum2;
int iTam;
cout << "Hay nhap mot so nguyen: " << endl;
cin >> iNum1;
cout << "Hay nhap mot so nguyen: " << endl;
cin >> iNum2;
cout<<"Gia tri cua so thu nhut: iNum1 = "<<iNum1<< endl;
cout << "Gia tri cua so thu hai: iNum2 = "<<iNum2 << endl;

```

```

if (iNum1 != iNum2)
{
    iTam = iNum1;
    iNum1 = iNum2;
    iNum2 = iTam;
}

```

```

cout << "Cac gia tri sau khi hoan vi la: " << endl;
cout << "iNum1 = "<< iNum1 << " va iNum2 = "<<iNum2<< endl;
}

```

## 2. Câu lệnh *if...else*

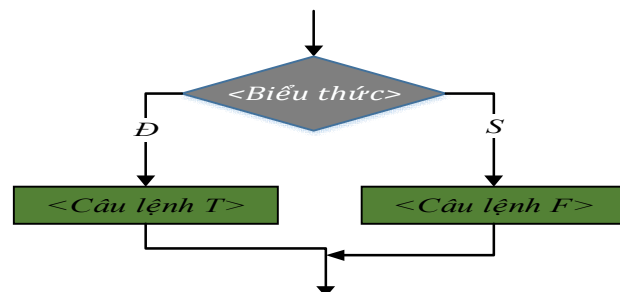
Cú pháp:

```

if (<Biểu thức>)
    <Câu Lệnh T>;
else
    <Câu Lệnh F>;

```

Lưu đồ:



Hoặc:

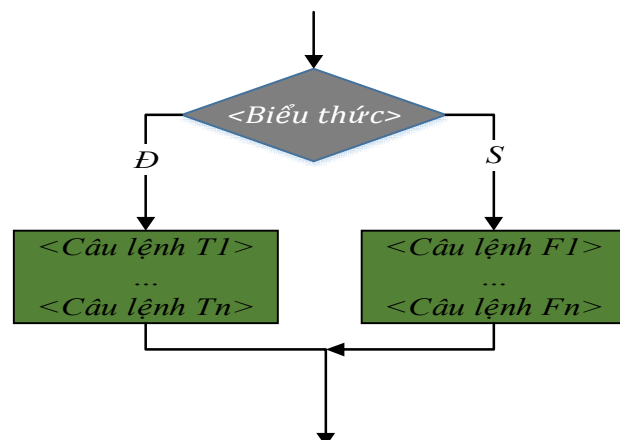
Cú pháp:

```

if (<Biểu thức>)
{
    <Câu Lệnh T1>;
    ...
    <Câu Lệnh Tn>;
}
else
{
    <Câu Lệnh F1>;
    ...
    <Câu Lệnh Fn>;
}

```

Lưu đồ:



**Ví dụ 3.3:** Viết chương trình nhập vào 1 số nguyên. Cho biết số vừa nhập là số chẵn hay số lẻ.

---

```
/*
Chương trình kiểm tra một số là chẵn hay lẻ
Thêm tên của bạn ở đây
*/

//Khai báo thư viện
#include<iostream>
using namespace std;

//Hàm chính
void main()
{
    cout << "CHƯƠNG TRÌNH KIỂM TRA CHẴN LẺ" << endl;
    int iNum;

    cout << "Hãy nhập một số nguyên: " << endl;
    cin >> iNum;

    if (iNum % 2 == 0)
        cout << iNum << " là số chẵn!" << endl;
    else
        cout << iNum << " là số lẻ!" << endl;
}
```

---

**Ví dụ 3.4:** Viết chương trình nhập vào 2 số nguyên. Tìm giá trị lớn nhất và giá trị nhỏ nhất trong 2 số vừa nhập.

---

```
/*
Chương trình tìm giá trị lớn nhất, nhỏ nhất
Thêm tên của bạn ở đây
*/

//Khai báo thư viện
#include<iostream>
using namespace std;

//Hàm chính
void main()
{
    cout<<"CHƯƠNG TRÌNH TÌM GIÁ TRỊ LỚN NHẤT, NHỎ NHẤT"<<
                                                endl;
```

---

---

```
int iNum1, iNum2;
int iMin, iMax;

cout << "Hay nhap mot so nguyen: " << endl;
cin >> iNum1;
cout << "Hay nhap mot so nguyen: " << endl;
cin >> iNum2;

if (iNum1 > iNum2)
{
    iMax = iNum1;
    iMin = iNum2;
}
else
{
    iMax = iNum2;
    iMin = iNum1;
}

cout << "Gia tri lon nhat la: " << iMax << endl;
cout << "Gia tri nho nhat la: " << iMin << endl;
}
```

---

#### IV. Bài tập:

**Bài 1: (Sửa lỗi)** Hãy viết lại chính xác chương trình sau:

---

```
//Day la chuong trinh kiem tra gia tri khoi tao
//la bang hay khong bang gia tri nguoi dung nhap

//Nhap ten ban o day

//Khai bao thu vien
#include<iostream>
using namespace std;

//Ham chinh
void main()
{
    int iNum1; //num1 khong duoc khoi tao
    int iNum2 = 5; //num2 duoc khoi tao gia tri 5

    cout<<"Hay nhap mot so nguyen: "<<endl;
    cin>>iNum1;

    cout<<"num1 = "<<iNum1<<" va num2 = "<<iNum2<<endl;
}
```

---

---

```

if(iNum1 == iNum2)
    cout<<"Hey, Day la mot su trung hop!"<<endl;
if(iNum1 != iNum2)
    cout<<"Cac gia tri la khong giong nhau."<<endl;
}

```

---

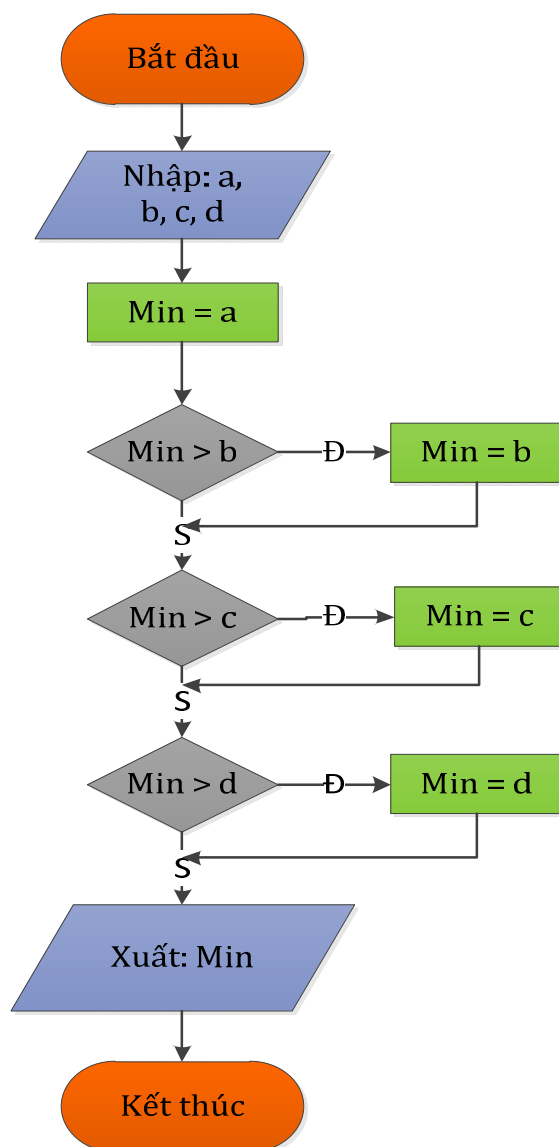
- Chạy chương trình một vài lần với các giá trị nhập khác nhau mỗi lần chạy. Chương trình có thực hiện như mong muốn của bạn? Nếu có, hãy giải thích chương trình đang làm gì. Nếu không, hãy chỉ ra lỗi và sửa chúng.
- Sửa lại chương trình để người dùng **nhập vào hai giá trị** để kiểm tra phép so sánh bằng. Đảm bảo bạn phải thông báo mỗi lần nhập. Kiểm tra chương trình với các cặp giá trị giống nhau và khác nhau.
- Sửa lại chương trình để khi các giá trị là giống nhau nó sẽ in ra các dòng:  
**Cac gia tri la giong nhau.**  
**Hey, day la mot su trung hop!**
- Sửa lại chương trình bằng cách thay câu lệnh **if** bằng câu lệnh **if/else**. Chạy lại chương trình và kiểm tra kết quả.

### ❖ Dạng bài toán tìm kiếm:

**Bài 2: (Tìm giá trị nhỏ nhất)** Viết chương trình nhập vào 4 số nguyên a, b, c, d. Tìm và xuất lên màn hình số nhỏ nhất trong 4 số vừa nhập.

Lưu đồ:

- Input:  $a, b, c, d \in \mathbb{R}$
- Output: Giá trị nhỏ nhất của 4 số  $a, b, c, d$



**Bài 3: (Tìm số dương nhỏ nhất)** Viết chương trình nhập vào 4 số nguyên a, b, c, d. Hãy xuất lên màn hình giá trị dương nhỏ nhất trong 4 số trên (nếu có)?

**Bài 4: (Tìm số âm lớn nhất)** Viết chương trình cho phép người dùng nhập vào 4 số nguyên a, b, c, d. Hãy xuất lên màn hình giá trị âm lớn nhất trong 4 số vừa nhập (nếu có)?

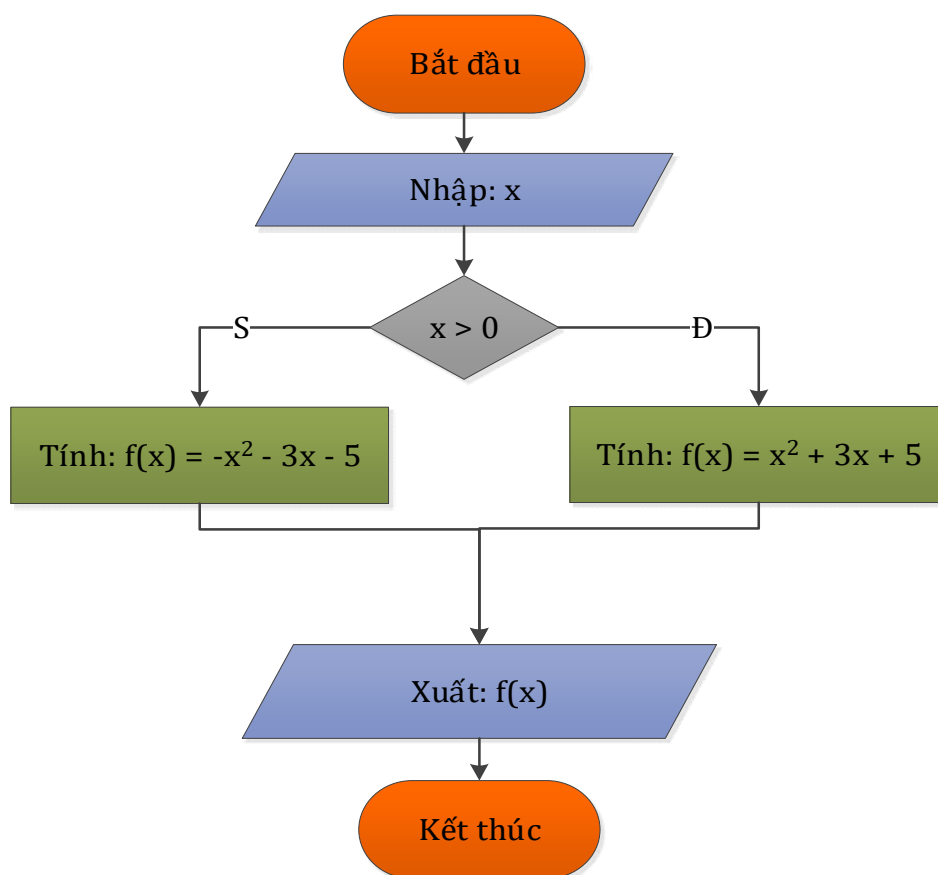
❖ **Dạng bài toán tính biểu thức:**

**Bài 5: (Tính biểu thức)** Viết chương trình cho phép người dùng nhập vào số thực x. Sử dụng câu lệnh điều kiện *if* để tính  $f(x)$  theo giá trị x với công thức sau:

$$f(x) = \begin{cases} x^2 + 3x + 5, & x > 0 \\ -x^2 - 3x - 5, & x \leq 0 \end{cases}$$

Lưu đồ:

- Input:  $x \in \mathbb{R}$
- Output:  $f(x)$



**Bài 6: (Số học, giá trị lớn nhất và giá trị nhỏ nhất)** Viết chương trình nhập vào ba số nguyên từ bàn phím, sau đó xuất lên màn hình tổng, trung bình, tích và giá trị lớn nhất, nhỏ nhất của những số trên.

❖ **Dạng bài toán đếm:**

**Bài 7: (Đếm số lượng chẵn, lẻ)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương N (với  $N \in [10000, 99999]$ ). Hãy cho biết trong số N vừa nhập có bao nhiêu chữ số chẵn, bao nhiêu chữ số lẻ?



**Bài 8: (Đếm số lượng số âm, dương)** Viết chương trình cho phép người dùng nhập vào 5 số a, b, c, d, e. Hãy cho biết trong 5 số vừa nhập có bao nhiêu số âm, bao nhiêu số dương?

❖ **Dạng bài toán kiểm tra:**

**Bài 9: (Kiểm tra năm hợp lệ)** Viết chương trình cho phép người dùng nhập vào một giá trị năm. Hãy cho biết giá trị năm vừa nhập có hợp lệ hay không? (Với qui ước: năm hợp lệ là năm không âm).

**Bài 10: (Kiểm tra tháng hợp lệ)** Viết chương trình cho phép người dùng nhập vào một giá trị tháng. Hãy cho biết giá trị tháng vừa nhập có hợp lệ hay không? (Với qui ước: tháng hợp lệ là tháng  $\in [1, 12]$ ).

**Bài 11: (Kiểm tra bội số)** Viết chương trình cho phép người dùng nhập vào hai số nguyên a, b. Hãy cho biết số a có phải là bội của số b hay không?

**Bài 12: (Kiểm tra ước số)** Viết chương trình cho phép người dùng nhập vào hai số nguyên a, b. Hãy cho biết số a có phải là ước số của số b hay không?

## BÀI THỰC HÀNH TUẦN 04 – BUỔI 01

### CÂU LỆNH ĐIỀU KHIỂN *if* LỒNG NHAU

#### ❖ Mục đích:

1. Làm việc với câu lệnh *if* lồng nhau
2. Sửa một số lỗi liên quan câu lệnh *if* lồng nhau

#### I. Câu lệnh *if* lồng

Chúng ta cũng có thể sử dụng câu lệnh *if* bên trong một câu lệnh *if* khác. Điều này được gọi là *if* lồng nhau.

Ví dụ:

**Nếu** thời tiết đẹp

Tôi sẽ ra ngoài sân

**Và nếu** trời đủ mát

Tôi sẽ ngồi dưới ánh mặt trời

**Ngược lại**

Tôi sẽ ngồi trong bóng râm

**Ngược lại**

Tôi sẽ ở trong nhà

Tôi sẽ uống một chút nước chanh

Trong lập trình, điều này tương ứng như sau:

```

if(<biểu thức 1>)           //thời tiết đẹp
{
    <Câu lệnh A>;           //Đúng – Ra ngoài sân
    if(<biểu thức 2>)       //đủ mát
        <Câu lệnh B>;      //Đúng – Ngồi dưới ánh mặt trời
    else
        <Câu lệnh C>;      //Sai – Ngồi trong bóng râm
}
else
    <Câu lệnh D>;          //Thời tiết không đẹp – Ngồi trong nhà

```

<Câu lệnh E>; //Uống nước chanh trong bất kỳ sự kiện nào

**Ví dụ 4.1:** Viết chương trình cho phép người dùng nhập vào một số nguyên. Hãy cho biết số vừa nhập có phải là số chẵn hay không và một nửa của số vừa nhập có phải là số chẵn hay không?

---

```
/* Thông tin chương trình và tác giả
Đây là chương trình minh họa if long
Họ tên: Nhập tên bạn ở đây
MSSV:
Lop:
Ngày:
*/

//Khai báo thu vien
#include<iostream>
using namespace std;

//Ham chinh
void main()
{
    //Khai báo biến
    long lNum;

    //Nhập dữ liệu
    cout<<"Hãy nhập vào một số nguyên: "<<endl;
    cin>>lNum;

    //Kiểm tra chẵn hay lẻ
    if(lNum % 2 == 0)
    {
        cout<<"Số "<<lNum<<" là số chẵn."<<endl;
        if((lNum/2)%2==0)
            cout<<"Một nửa của "<<lNum<<" cũng là số
                chẵn."<<endl;
        else
            cout<<"Một nửa của "<<lNum<<" là số lẻ."<<endl;
    }
    else
        cout<<"Số "<<lNum<<" là số lẻ."<<endl;
}
```

---

## II. Sửa một số lỗi liên quan câu lệnh if lồng nhau

### 1. Lỗi cú pháp

Trong quá trình lập trình với câu lệnh **if** và **if/else**, bạn có thể gặp phải một số lỗi cú pháp sau:

- Sau biểu thức điều kiện có dấu chấm phẩy ;

---

```
if (cKyTu >= 'A' && cKyTu <= 'Z') ;
{
    cKyTu = cKyTu + 32; //Chuyen hoa sang thuong
    cout<<"Ky tu thuong tuong ung: "<<cKyTu<<endl;
}
else
    cout<<"Ban da khong nhap vao mot ky tu hoa."
<<endl;
```

---

- Bạn mong muốn thực hiện một khối lệnh nhưng không đặt khối lệnh này trong cặp dấu ngoặc {}

---

```
if (cKyTu >= 'A' && cKyTu <= 'Z')
    cKyTu = cKyTu + 32; //Chuyen hoa sang thuong
    cout<<"Ky tu thuong tuong ung: "<<cKyTu<<endl;
else
    cout<<"Ban da khong nhap vao mot ky tu hoa"
<<endl;
```

---

### 2. Lỗi logic

Bạn cũng có thể gặp phải các lỗi logic trong một số trường hợp:

- Bạn mong muốn thực hiện một khối lệnh nhưng không đặt khối lệnh này trong cặp dấu ngoặc {}

---

```
if (delta > 0)
    x1 = (-b + sqrt(delta)) / (2*a);
    x2 = (-b - sqrt(delta)) / (2*a);
```

---

Lúc này, chỉ có câu lệnh  $x1 = (-b + \sqrt{\text{delta}})/(2*a)$ ; nằm trong if. Nghĩa là điều kiện  $\text{delta} > 0$  là đúng hay sai thì câu lệnh  $x2 = (-b - \sqrt{\text{delta}})/(2*a)$ ; luôn được thực hiện.

- Lỗi ở trên cũng thường xuất hiện trong phần else

---

```
if (a > 0)
    a = b;
else
    a = a + b;
    b = 2*a;
```

---

**Ví dụ 4.2:** Viết chương trình cho phép người dùng nhập vào một ký tự hoa. Hãy chuyển ký tự vừa nhập sang chữ thường.

---

```
/* Thông tin chương trình và tác giả
   Đây là chương trình chuyển chữ hoa sang chữ thường
   Họ tên: Nhập tên bạn ở đây
   MSSV:
   Lớp:
   Ngày:
   */

//Khai báo thư viện
#include<iostream>
using namespace std;

//Hàm chính
void main()
{
    //Khai báo biến
    char cKyTu;

    //Nhập liệu
    cout<<"Hãy nhập vào một ký tự hoa: "<<endl;
    cin>>cKyTu;

    //Kiểm tra chữ hoa
    if(cKyTu >= 'A')
    {
        if(cKyTu <= 'Z')
        {
            cKyTu = cKyTu + 32; //Chuyển hoa sang thường
        }
    }
}
```

---

---

```
        cout<<"Ky tu thuong tuong ung: "<<cKyTu
            <<endl;
    }
    else
    {
        if(cKyTu >= 'a')
        {
            if(cKyTu <= 'z')
                cout<<" Hay dung phim shift! Toi
                    muon mot ky tu hoa."<<endl;
            }
            else
                cout<<"Ban da khong nhap vao mot chu cai."
                    <<endl;
        }
    }
    else
        cout<<"Ban da khong nhap vao mot chu cai."<<endl;
}
```

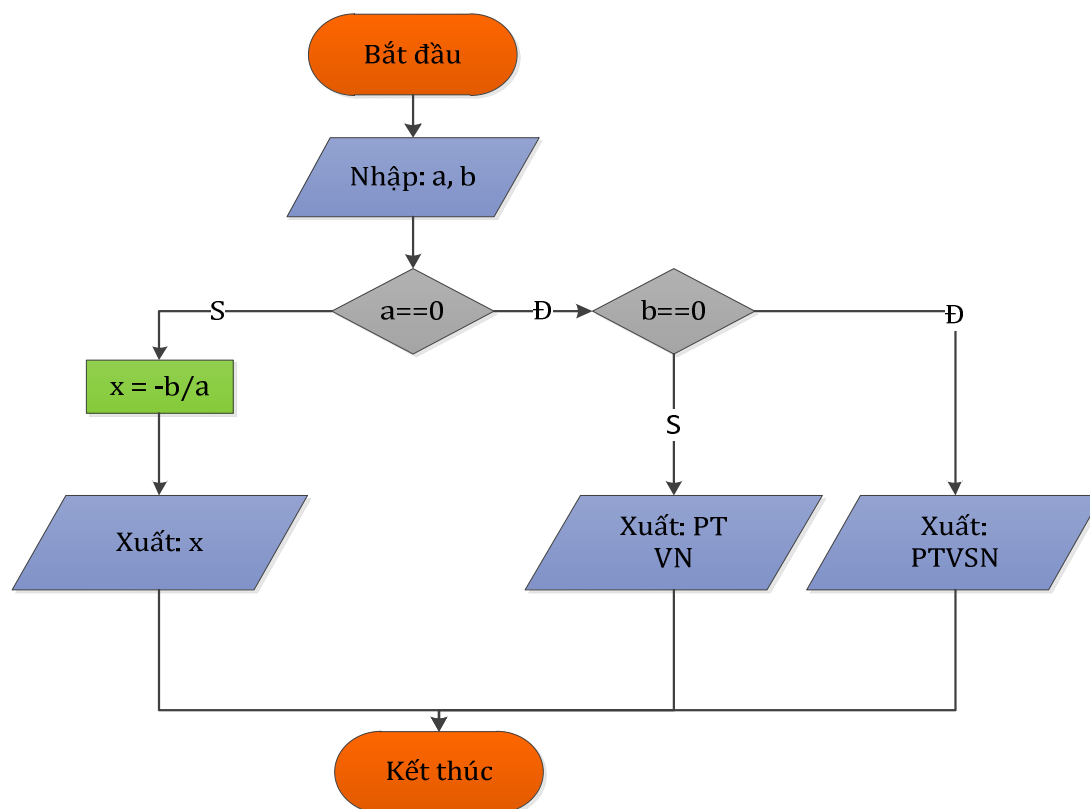
---

### III. Bài tập:

**Bài 1:** (*Phương trình bậc nhất*) Viết chương trình giải phương trình bậc 1:  $ax + b = 0$ , với  $a, b$  là các số thực nhập vào từ bàn phím.

Lưu đồ:

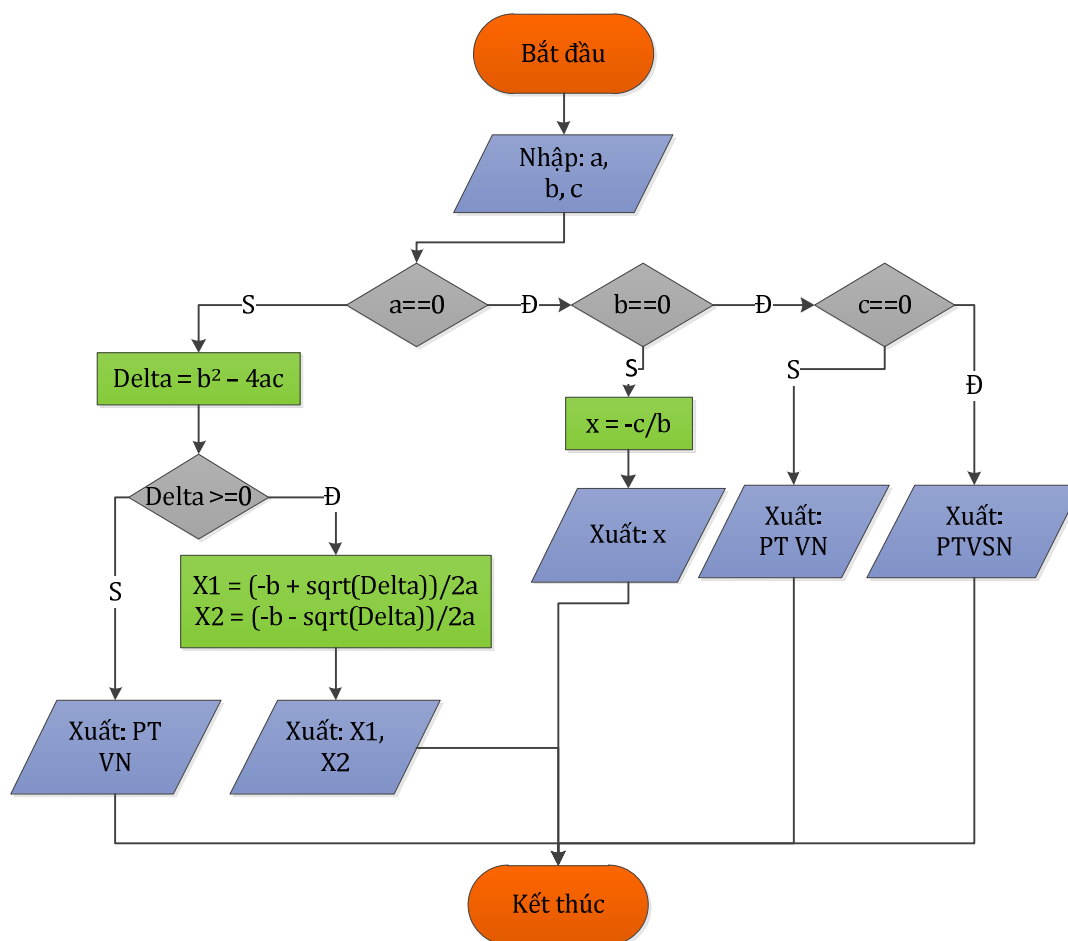
- Input:  $a, b \in \mathbb{R}$
- Output: Nghiệm của phương trình  $ax + b = 0$



**Bài 2: (Phương trình bậc hai)** Viết chương trình giải phương trình bậc 2:  $ax^2 + bx + c = 0$ , với  $a, b, c$  là các số thực nhập vào từ bàn phím.

Lưu đồ:

- Input:  $a, b, c \in \mathbb{R}$
- Output: Nghiệm của phương trình bậc hai  $ax^2 + bx + c = 0$

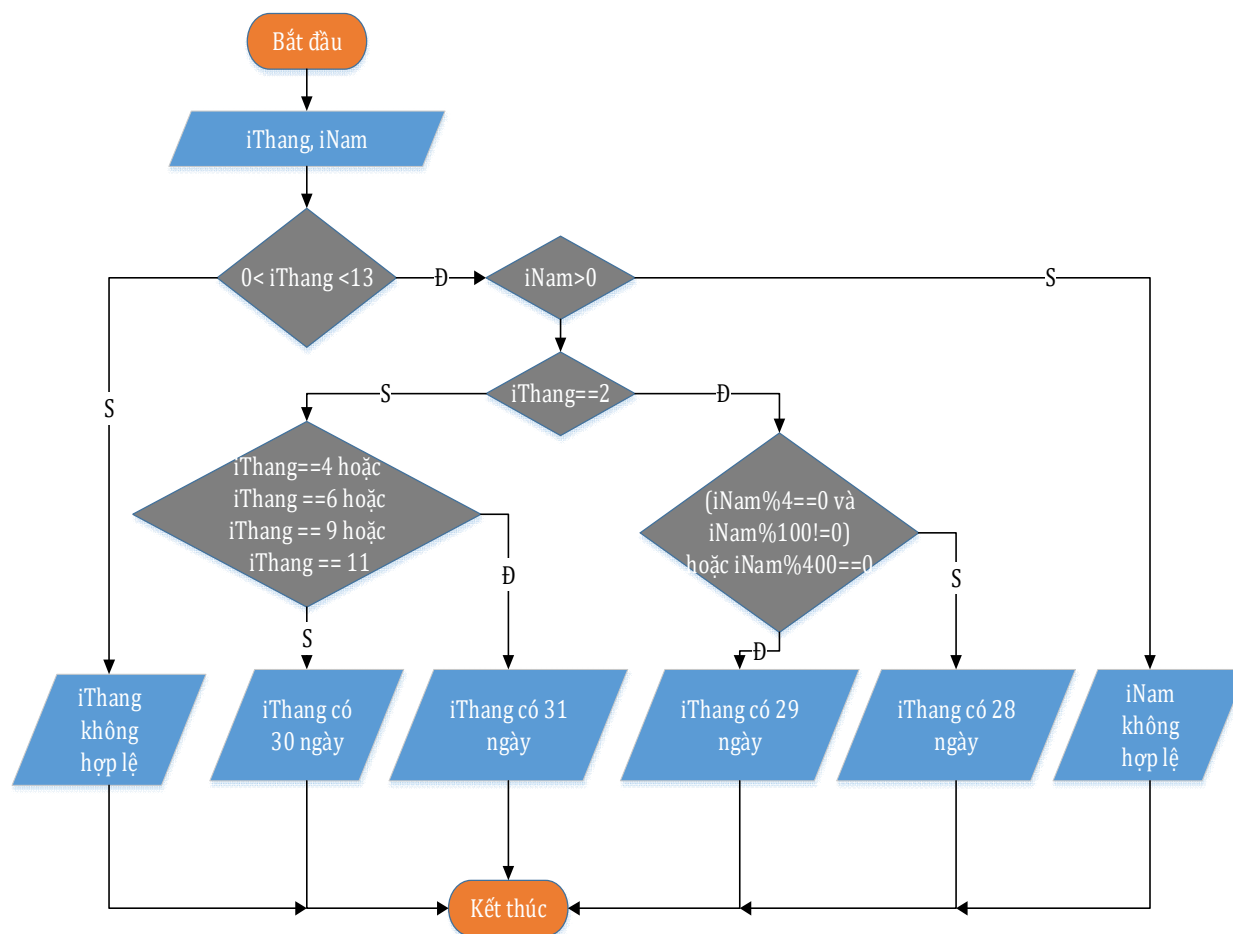


**Bài 3: (Số ngày trong tháng)** Viết chương trình nhập vào Tháng và Năm (dương lịch). Hãy xuất lên màn hình số ngày của Tháng vừa nhập. (biết rằng: Năm nhuận là năm chia hết cho 4 và không chia hết cho 100 hoặc chia hết cho 400).

Lưu đồ:

- Input: Tháng, Năm  $\in \mathbb{N}$
- Output: Số ngày của tháng trên.

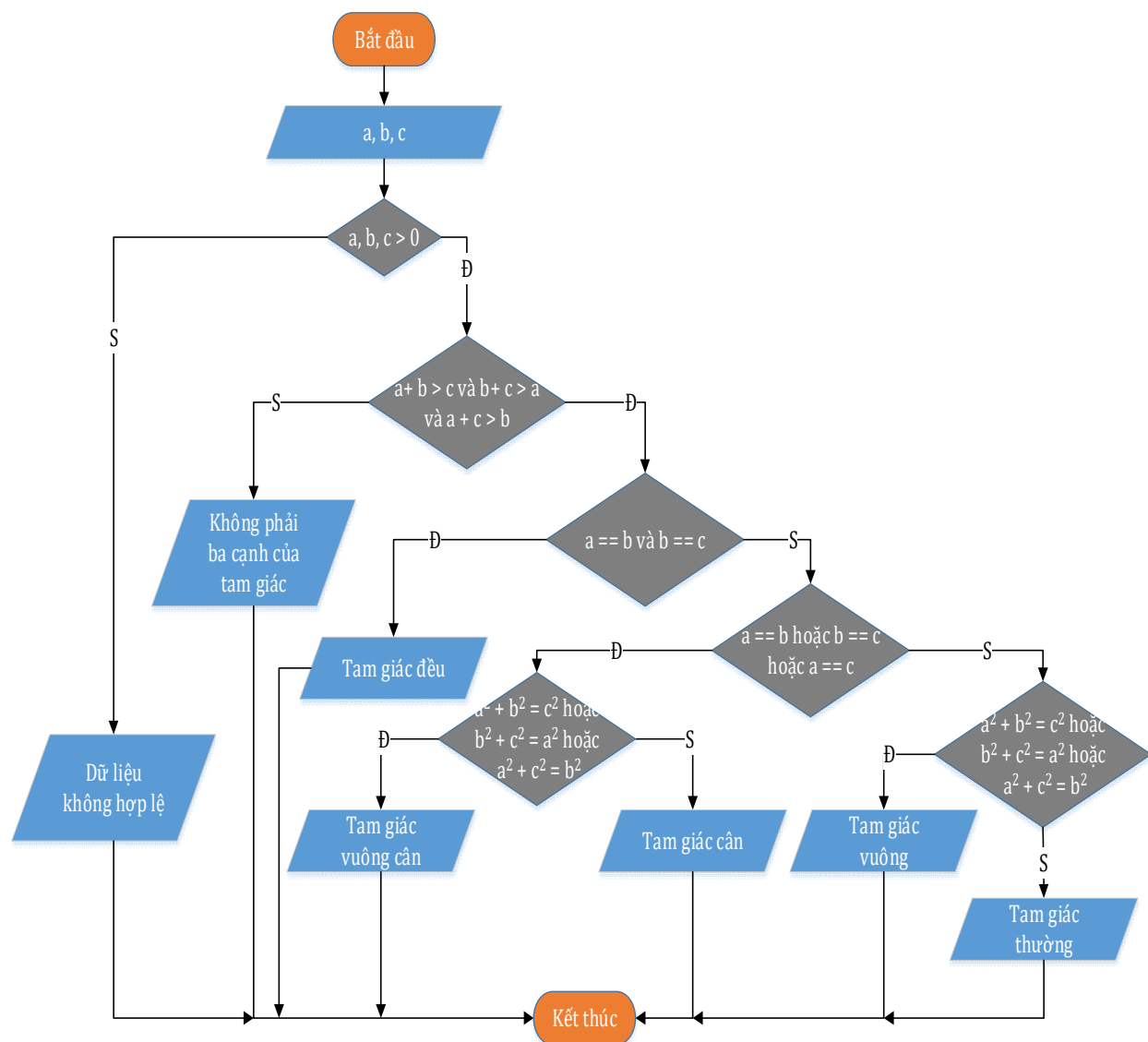




**Bài 4: (Loại tam giác)** Viết chương trình cho phép người dùng nhập vào ba số a, b, c. Hãy cho biết 3 số trên có thể là độ dài 3 cạnh của một tam giác? Nếu là một tam giác thì đó là tam giác gì: vuông, đều, cân, vuông cân hay tam giác thường?

Lưu đồ:

- Input:  $a, b, c \in \mathbb{R}$
- Output: Nếu là tam giác cho biết loại gì?



## BÀI THỰC HÀNH TUẦN 04 – BUỔI 02

### CÂU LỆNH ĐIỀU KHIỂN *switch*

#### ❖ Mục đích:

1. Làm việc với câu lệnh *switch-case*
2. Làm việc với câu lệnh *switch-case* lồng nhau

#### I. Câu lệnh *switch-case*

Câu lệnh *switch-case* là một cấu trúc điều kiện khác mà có thể hoặc không thể thực thi những câu lệnh. Tuy nhiên, *switch-case* có cú pháp và hành vi đặc biệt:

---

```
switch(<biểu thức>)  
{  
    case <hằng số 1>:  
        <Câu lệnh A1>;  
        <Câu lệnh A2>;  
        ...  
        break;  
    case <hằng số 2>:  
        <Câu lệnh B1>;  
        <Câu lệnh B2>;  
        ...  
        break;  
    ...  
    default:  
        <Câu lệnh Z1>;  
        <Câu lệnh Z2>;  
        ...  
        break;  
}
```

---

**Ví dụ 4.3:** Viết chương trình cho phép người dùng nhập vào một ký tự. Hãy xuất lên màn hoặc hình chữ “Yes” nếu ký tự vừa nhập là y/Y hoặc chữ “No” nếu ký tự vừa nhập là n/N hoặc chữ “Error” nếu không phải các trường hợp trên.

---

```
/* Thông tin chương trình và tác giả  
Đây là chương trình minh họa câu lệnh switch  
Họ tên: Nhập tên bạn ở đây  
MSSV:  
Lớp:  
Ngày:
```

---

---

```
*/  
  
//Khai bao thu vien  
#include<iostream>  
using namespace std;  
  
//Ham chinh  
void main()  
{  
    //Khai bao bien  
    char cKyTu = 'y';  
  
    //Kiem tra gia tri ky tu  
    switch(cKyTu)  
    {  
        case 'y':  
        case 'Y':  
            cout<<"Yes"<<endl;  
            break;  
        case 'n':  
        case 'N':  
            cout<<"No"<<endl;  
            break;  
        default:  
            cout<<"Error"<<endl;  
            break;  
    }  
}
```

---

- Hãy thực thi chương trình trên và quan sát kết quả trên màn hình.
- Hãy bỏ tất cả các lệnh break có trong chương trình trên. Sau đó, thực thi lại chương trình và cho nhận xét về kết quả trên màn hình?
- Sửa lại chương trình trên cho phép người dùng nhập vào một ký tự. Sau đó, thực thi chương trình trên với các mẫu sau:

**Hãy nhập vào một ký tự:**

y

Yes

**Hãy nhập vào một ký tự:**

Y

Yes

**Hay nhập vào một ký tự:**

**n**

**No**

**Hay nhập vào một ký tự:**

**N**

**No**

**Hay nhập vào một ký tự:**

**h**

**Error**

## II. Câu lệnh *switch-case* lồng nhau

Bạn cũng có thể sử dụng các câu lệnh *switch-case* lồng vào nhau như ví dụ sau:

**Ví dụ 4.4:** Viết chương trình xuất lên màn hình danh sách các chức năng của chương trình gồm: 1. Tính tuổi, 2. Cho biết tháng thuộc mùa nào trong năm. Chương trình cho phép người dùng chọn chức năng muốn thực hiện và chương trình thực hiện chức năng tương ứng.

---

```
/*Thông tin chương trình và tác giả
Chương trình tạo menu cho phép người dùng chọn chức năng thực
hiện
Họ tên:
MSSV:
Lớp:
Ngày:
*/

//Khai báo thư viện
#include<iostream>
using namespace std;

//Hàm chính
void main()
{
    //Khai báo biến
    int iChoose;
```

---

---

```
cout<<"HAY NHAP MOT SO NGUYEN TUONG UNG VOI CHUC NANG
                                BEN DUOI "<<endl;

cout<<"1. Tinh tuoi"<<endl;
cout<<"2. Cho biet thang trong nam thuoc mua nao"<<endl;

cout<<"Hay nhap chuc nang ban muon: "<<endl;
cin>>iChoose;

//Xet chuc nang nguoi dung chon
switch(iChoose)
{
case 1:
    {
        cout<<"Ban chon chuc nang tinh tuoi."<<endl;
        //Khai bao bien
        int iNamSinh, iNamHienTai, iTuoi;

        //Nhap du lieu
        cout<<"Hay nhap nam hien tai: "<<endl;
        cin>>iNamHienTai;
        cout<<"Hay nhap nam sinh: "<<endl;
        cin>>iNamSinh;

        //Kiem tra du lieu hop le
        if(iNamSinh < iNamHienTai)
        {
            //Tinh tuoi
            iTuoi = iNamHienTai - iNamSinh;
            //Xuat ket qua
            cout<<"So tuoi la: "<<iTuoi<<endl;
        }
        else //Du lieu nhap khong hop le
            cout<<"Nam sinh phai nho hon nam hien
                                tai!"<<endl;
    }
    break;
case 2:
    {
        cout<<"Ban chon chuc nang xac dinh mua cua mot
                                thang."<<endl;

        //Khai bao bien
        int iThang;
        //Nhap du lieu
        cout<<"Hay nhap thang ban muonkiem tra: " <<endl;
        cin>>iThang;

        //Xet thang tuong ung voi mua nao
```

---

---

```
        switch (iThang)
        {
            case 1:
            case 2:
            case 3:
                cout<<"Thang "<<iThang<<" thuoc mua
                    xuan."<<endl;

                break;
            case 4:
            case 5:
            case 6:
                cout<<"Thang "<<iThang<<" thuoc mua
                    he."<<endl;

                break;
            case 7:
            case 8:
            case 9:
                cout<<"Thang "<<iThang<<" thuoc mua
                    thu."<<endl;

                break;
            case 10:
            case 11:
            case 12:
                cout<<"Thang "<<iThang<<" thuoc mua
                    dong."<<endl;

                break;
            default:
                cout<<"Khong co thang nay trong
                    nam!"<<endl;
        }
    }
    break;
default:
    cout<<"Khong ton tai chuc nang nay!"<<endl;
}
}
```

---

### III. Bài tập:

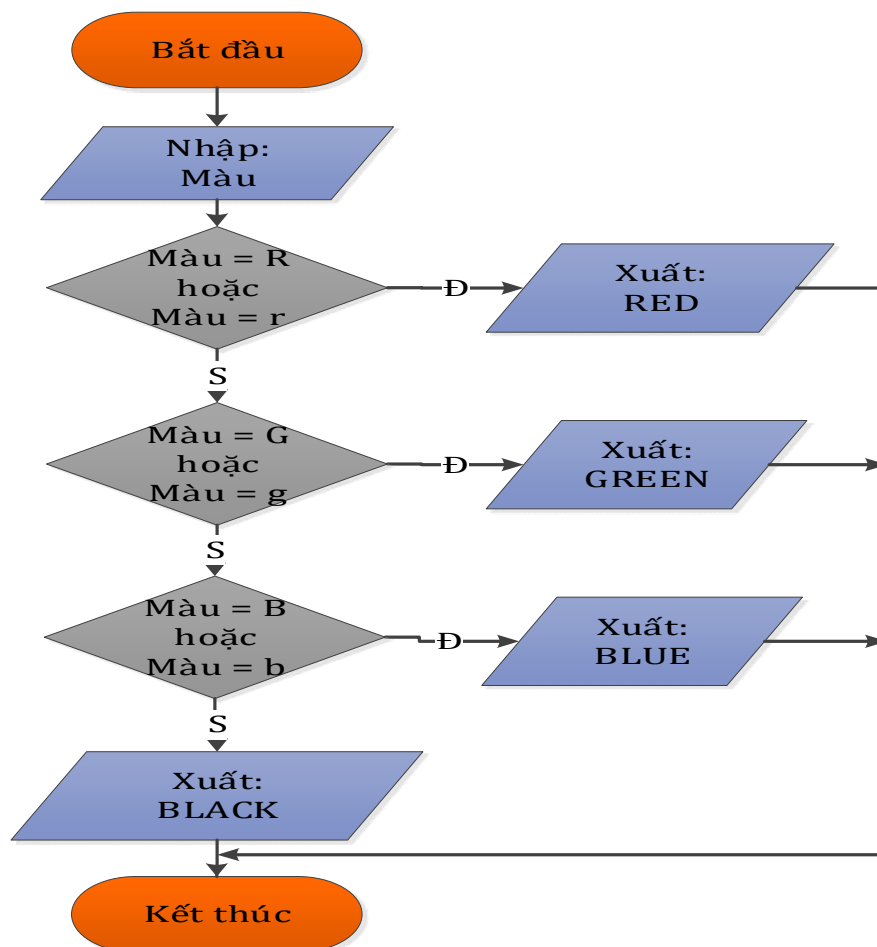
**Bài 1: (Đọc giá trị màu)** Viết chương trình thực hiện xác định giá trị màu do người dùng nhập vào từ bàn phím và xuất kết quả lên màn hình như sau:

- Nếu màu = 'R' hoặc màu = 'r' thì xuất lên RED.
- Nếu màu = 'G' hoặc màu = 'g' thì xuất lên GREEN.
- Nếu màu = 'B' hoặc màu = 'b' thì xuất lên BLUE.

- Nếu màu là các giá trị khác thì xuất BLACK.

Lưu đồ:

- *Input*: Ký tự  $\in \{R, G, B\}$
- *Output*: RED, GREEN, BLUE và BLACK



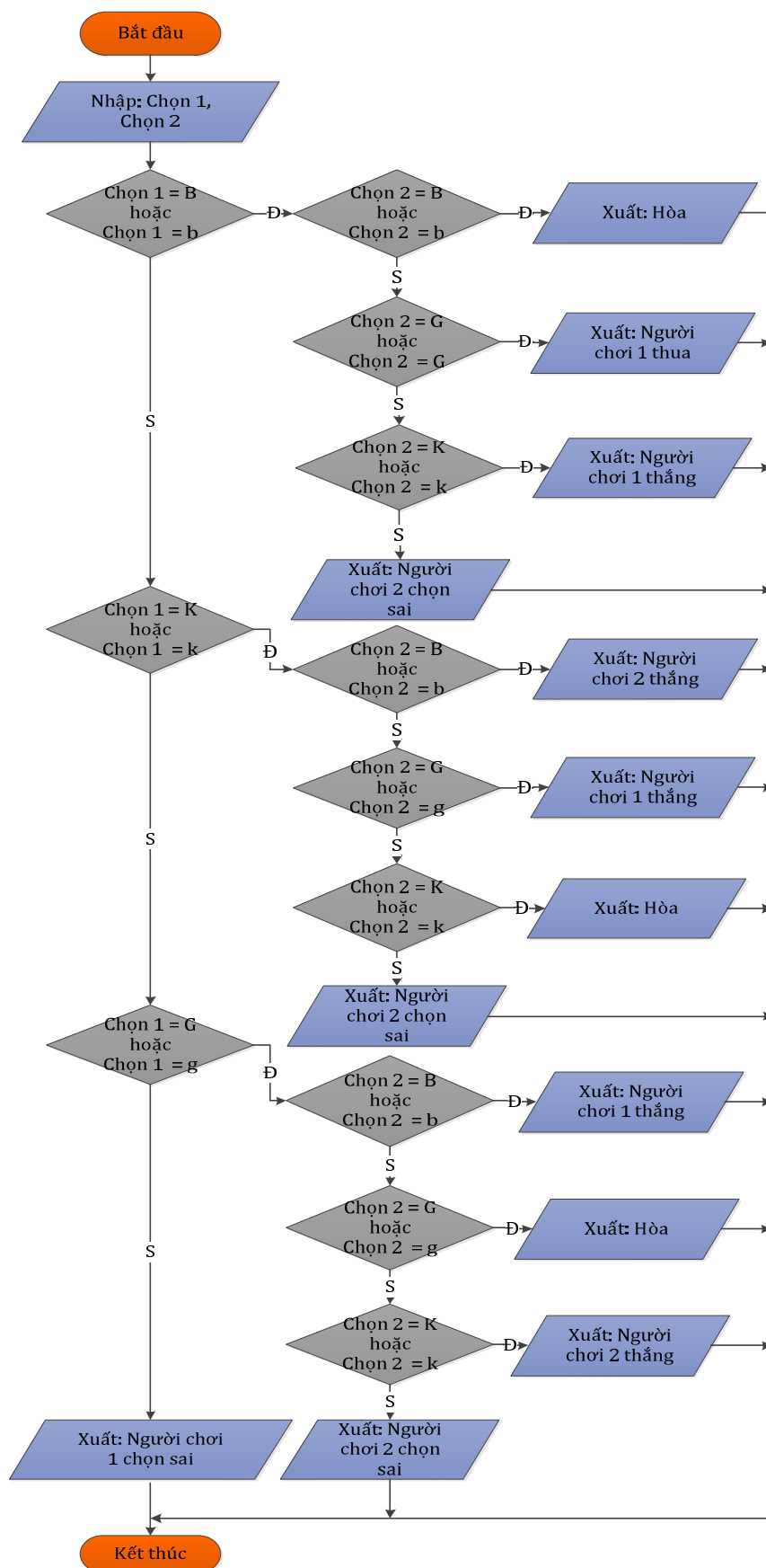
**Bài 2: (Trò chơi One-Two-Three)** Viết chương trình mô phỏng trò chơi One-Two-Three ra cái gì ra cái này theo điều kiện sau:

- Búa (B) thắng Kéo (K), thua Giấy (G).
- Kéo thắng Giấy, thua Búa.
- Giấy thắng Búa, thua Kéo.

Lưu đồ:

- *Input*: Chọn 1, Chọn 2 tương ứng với 2 người chơi chọn kéo, búa và giấy
- *Output*: Kết quả trò chơi.



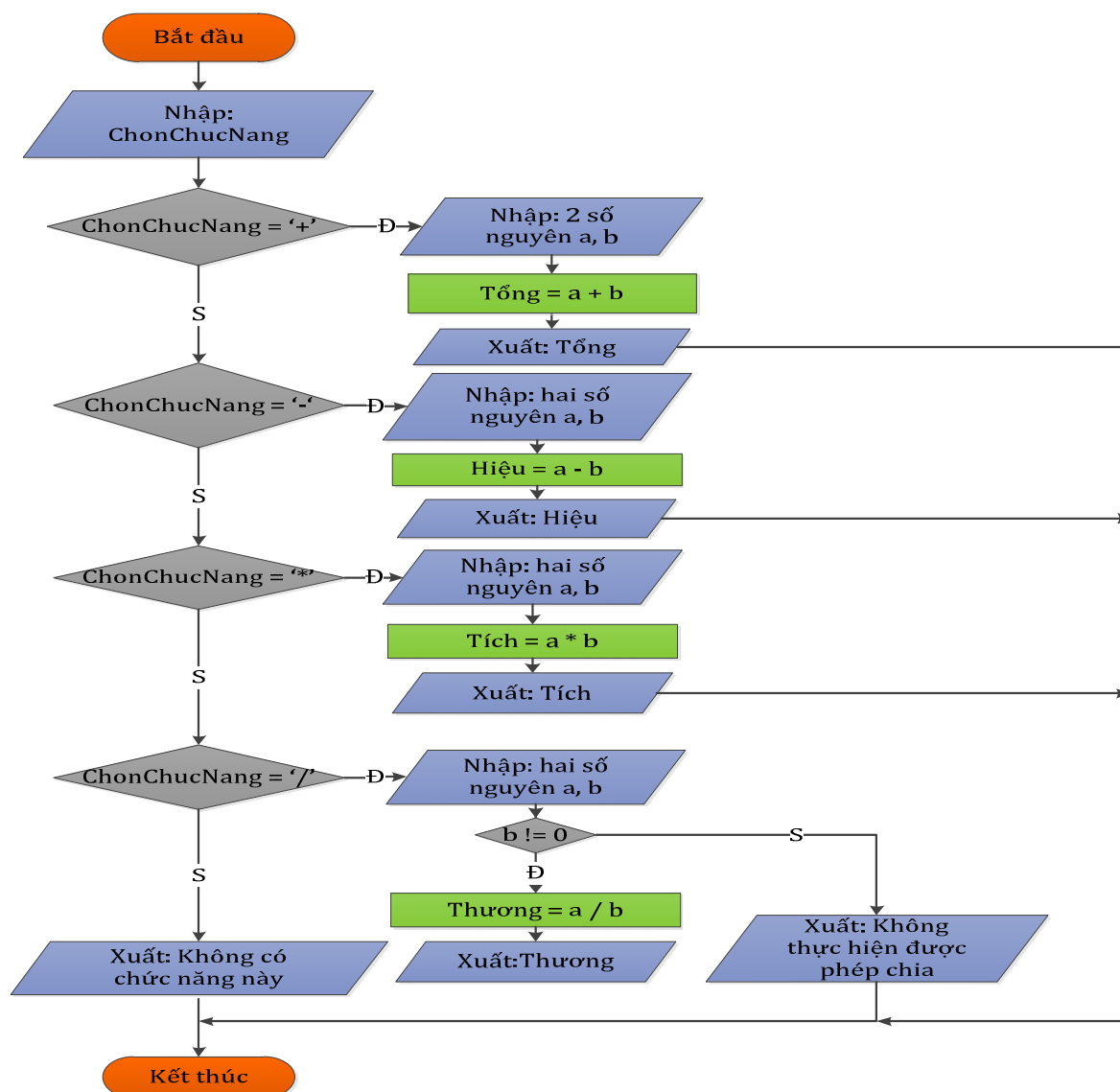


**Bài 3: (Lựa chọn chức năng)** Viết chương trình tạo menu các chức năng: + (phép tính cộng), - (phép tính trừ), \* (phép tính nhân), / (phép tính chia). Chương trình cho phép người dùng nhập vào chức năng muốn thực hiện. Ứng với chức năng người dùng chọn, chương trình thực hiện việc nhập 2 số nguyên a, b và thực hiện chức năng tương ứng.

**Lưu ý:** Đối với phép chia nếu  $b = 0$  thì thông báo không chia được.

**Lưu đồ:**

- Input: Các phép toán (+, -, \*, /)
- Output: Kết quả tương ứng với phép toán người dùng chọn



**Bài 4:** (*Kiểm tra ngày tháng năm hợp lệ*) Viết chương cho phép người dùng nhập vào ngày, tháng, năm. Hãy xác định và xuất lên màn hình cho biết ngày, tháng, năm vừa nhập có hợp lệ hay không?

**Bài 5:** (*Đọc giá trị*) Viết chương cho phép người dùng nhập vào một số nguyên dương N (với  $N \in [100, 999]$ ). Hãy xuất lên màn hình giá trị của N bằng chữ?

**Ví dụ:**

**Hay nhập số nguyên dương: 123**

**Một trăm hai mươi ba**

**Hay nhập số nguyên dương: 900**

**Chín trăm**

**Hay nhập số nguyên dương: 103**

**Một trăm lẻ ba**

## BÀI THỰC HÀNH TUẦN 05 – BUỔI 01

### BÀI THỰC HÀNH TỔNG HỢP 02

#### ❖ Mục đích:

1. Ôn tập cấu trúc rẽ nhánh với 2 câu lệnh *if/else* và *switch-case*
2. Vận dụng cài đặt các bài toán liên quan trên máy tính

#### I. Câu lệnh *if/else*

Trong quá trình lập trình, bạn cần một câu lệnh cho phép thực hiện một quyết định, bạn có thể sử dụng câu lệnh *if*. Câu lệnh *if* trong C++ có thể được viết đơn giản nhất với cú pháp như sau:

---

```
if(<Biểu thức logic>
    <Câu lệnh>;
Next_Statement;
```

---

Bạn cũng có thể mở rộng câu lệnh *if* để nó linh hoạt hơn bằng cách thêm vào *else*. Cú pháp của *if/else* sẽ là:

---

```
if(<Biểu thức logic>
    <Câu lệnh 1>;
else
    <Câu lệnh 2>;
Next_Statement;
```

---

Bạn cũng có thể viết các câu lệnh *if/else* lồng vào câu lệnh *if/else* khác nhiều cấp. Điều này có thể được minh họa như sau:

---

```
if(<Biểu thức logic 1>)
{
    if(<Biểu thức logic 2>)
        <Câu lệnh A1>;
    else
        <Câu lệnh A2>;
}
else
{
    if(<Biểu thức logic 3>)
        <Câu lệnh B1>;
    else
        <Câu lệnh B2>;
}
```

---

---

```
}
Next_Statement;
```

---

## II. Câu lệnh switch-case

Câu lệnh *switch-case* cho phép bạn lựa chọn thực thi một hành động từ một tập các hành động có khả năng, dựa trên kết quả của một biểu thức nguyên. Cú pháp tổng quát của câu lệnh *switch-case* như sau:

---

```
switch(<biểu thức>)
{
    case <Giá trị 1 của biểu thức>:
        <Câu lệnh 1>;
    break;
    .....
    case <Giá trị n của biểu thức>:
        <Câu lệnh n>;
    break;
    default:
        <Câu lệnh>;
    break;
}
```

---

## III. Bài tập:

**Bài 1: (Đổi ngày tháng)** Viết chương trình cho phép người dùng nhập vào 3 số nguyên tương ứng với ngày, tháng và năm. Chương trình sẽ xuất lên màn hình ngày tháng năm theo dạng nước Anh.

1 <sup>st</sup>	First	9 <sup>th</sup>	Ninth	17 <sup>th</sup>	Seventeenth	25 <sup>th</sup>	Twenty-fifth
2 <sup>nd</sup>	Second	10 <sup>th</sup>	Tenth	18 <sup>th</sup>	Eighteenth	26 <sup>th</sup>	Twenty-sixth
3 <sup>rd</sup>	Third	11 <sup>th</sup>	Eleventh	19 <sup>th</sup>	Nineteenth	27 <sup>th</sup>	Twenty-seventh
4 <sup>th</sup>	Fourth	12 <sup>th</sup>	Twelfth	20 <sup>th</sup>	Twentieth	28 <sup>th</sup>	Twenty-eighth
5 <sup>th</sup>	Fifth	13 <sup>th</sup>	Thirteenth	21 <sup>st</sup>	Twenty-first	29 <sup>th</sup>	Twenty-ninth
6 <sup>th</sup>	Sixth	14 <sup>th</sup>	Fourteenth	22 <sup>nd</sup>	Twenty-second	30 <sup>th</sup>	Thirtieth
7 <sup>th</sup>	Seventh	15 <sup>th</sup>	Fifteenth	23 <sup>rd</sup>	Twenty-third	31 <sup>st</sup>	Thirty-first
8 <sup>th</sup>	eighth	16 <sup>th</sup>	sixteenth	24 <sup>th</sup>	Twenty-fourth		

**Ví dụ:** Người dùng nhập vào ngày tháng năm là 31 12 2017. Chương trình của bạn sẽ xuất lên màn hình là 31ST December 2017

**Bài 2: (Tính tiền)** Viết chương trình cho phép người dùng nhập vào số lượng một sản phẩm. Chương trình sẽ tính thành tiền tương ứng với số lượng được nhập từ bàn phím với đơn giá một sản phẩm là 5\$ và có giảm giá 10% cho số lượng trên 30 và 15% cho số lượng trên 50.

**Bài 3: (Tính tiền nước)** Viết chương trình nhắc nhở người dùng về hóa đơn tiền nước hàng quý cho bốn quý gần nhất. Chương trình tìm và xuất hóa đơn tiền nước trung bình hàng tháng. Nếu hóa đơn trung bình vượt quá 75\$, xuất ra thông báo cho biết có quá nhiều nước đang được sử dụng. Nếu hóa đơn trung bình ít nhất là 25\$ nhưng không nhiều hơn 75\$, thông báo chỉ ra rằng một lượng nước hợp lý đang được sử dụng. Cuối cùng, nếu hóa đơn trung bình nhỏ hơn 25\$, xuất lên một thông điệp ca ngợi người dùng cho việc bảo tồn nước. Sử dụng mẫu chạy bên dưới như một mô hình cho đầu ra:

**Mẫu chạy 1:**

**Hay nhap vao hoa don nuoc quy I:**

**300**

**Hay nhap vao hoa don nuoc quy II:**

**200**

**Hay nhap vao hoa don nuoc quy III:**

**225**

**Hay nhap vao hoa don nuoc quy IV:**

**275**

**Hoa don trung binh hang thang cua ban la 83.33\$. Ban dang su dung qua nhieu nuoc.**

**Mẫu chạy 2:**

**Hay nhap vao hoa don nuoc quy I:**

100

**Hãy nhập vào hoa đơn nước quy II:**

150

**Hãy nhập vào hoa đơn nước quy III:**

75

**Hãy nhập vào hoa đơn nước quy IV:**

125

**Hoa đơn trung bình hàng tháng của bạn là 37.50\$. Bạn đang sử dụng một lương nước hợp lý.**

**Bài 4: (Tính tiền hóa đơn)** Một cửa hàng áo thun bán áo thun lẻ với giá 12\$. Số lượng lớn sẽ được giảm giá theo bảng dưới đây:

Số lượng	Giảm giá
5 – 10	10%
11 – 20	15%
21 – 30	20%
31 trở lên	25%

Viết chương trình cho phép người dùng nhập vào số lượng áo bán và sau đó tính tổng thành tiền. Hãy đảm bảo chương trình chỉ chấp nhận số nguyên dương.

Sử dụng các mẫu chạy sau làm hướng dẫn cho bạn:

**Mẫu chạy 1:**

**Bao nhiêu áo được bán?**

4

**Gia một áo là \$12 và tổng thanh tiền là \$48**

**Mẫu chạy 2:**

**Bao nhiêu áo được bán?**

0

**Gia một áo là \$12 và tổng thanh tiền là \$0**

**Mẫu chạy 3:**

**Bao nhiêu ao được bạn?**

**8**

**Gia một ao là \$10.80 và tổng thanh tiền là \$86.40**

**Mẫu chạy 4:**

**Bao nhiêu ao được bạn?**

**-2**

**Gia trị nhập không hợp lệ: Hay nhập một số nguyên dương**

**Bài 5: (Tính tiền học phí)** Giả sử Trường Cao Thắng thu học phí \$3000 một học kỳ đối với các sinh viên ở Tp.HCM và \$4500 với những sinh viên thuộc các tỉnh khác. Ngoài ra, Tiền ở ký túc xá và quản trị là \$2500 cho mỗi học kỳ đối với sinh viên Tp.HCM và \$3500 cho mỗi học kỳ đối với sinh viên ở tỉnh khác. Viết chương trình cho phép người dùng nhập vào nơi cư trú (nghĩa là, ở Tp.HCM hay ở tỉnh khác) và họ có ở ký túc xá hay không (Y hoặc N). Chương trình sẽ tính và xuất lên hóa đơn của sinh viên cho học kỳ đó.

Sử dụng các mẫu chạy sau:

**Mẫu chạy 1:**

**Hay nhập “I” nếu bạn ở Tp.HCM hoặc “O” nếu bạn ở ngoài Tp.HCM:**

**I**

**Hay nhập “Y” nếu bạn ở ký túc xá và “N” nếu không:**

**N**

**Tổng hóa đơn của bạn trong học kỳ này là \$3000**

**Mẫu chạy 2:**

**Hay nhập “I” nếu bạn ở Tp.HCM hoặc “O” nếu bạn ở ngoài Tp.HCM:**

**O**

**Hay nhập “Y” nếu bạn ở ký túc xá và “N” nếu không:**

**Y**

**Tổng hóa đơn của bạn trong học kỳ này là \$8000**



**Mẫu chạy 3:**

**Hay nhập “I” neu ban o Tp.HCM hoac “O” neu ban o ngoai Tp.HCM:**

**I**

**Hay nhập “Y” neu ban o ky tuc xa va “N” neu khong:**

**Y**

**Tong hoa don cua ban trong hoc ky nay la \$5500**

**Bài 6: (Đọc giá trị)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương N có 4 chữ số ( $N \in [1000, 9999]$ ). Hãy xuất lên màn hình cách đọc giá trị vừa nhập.

**Mẫu chạy 1:**

**Hay nhập mot so nguyen duong:**

**1234**

**Mot nghin hai tram ba muoi bon**

**Mẫu chạy 2:**

**Hay nhập mot so nguyen duong:**

**5680**

**Nam nghin sau tram tam muoi**

## BÀI THỰC HÀNH TUẦN 05 – BUỔI 02

### CÂU LỆNH LẶP *WHILE*

#### ❖ Mục đích:

1. Giới thiệu toán tử tăng giảm
2. Làm việc với câu lệnh lặp while
3. Giới thiệu bộ đếm và sự kiện kiểm soát vòng lặp

#### I. Toán tử tăng và giảm

Bạn có thể cộng hoặc trừ 1 từ một giá trị số nguyên cho trước:

---

```
count = count + 1;  
count += 1;
```

---

Có hai chế độ có thể được sử dụng:

---

```
count++; //toán tử tăng ở chế độ hậu tố  
count--; //toán tử giảm ở chế độ hậu tố  
  
++count; //toán tử tăng ở chế độ tiền tố  
--count; //toán tử giảm ở chế độ tiền tố
```

---

#### II. Câu lệnh *while*

Câu lệnh lặp *while* cho phép lặp lại một lệnh hoặc một khối lệnh trong suốt quá trình chương trình thực thi. Cú pháp như sau:

---

```
while (<biểu thức>)  
{  
    <Câu lệnh 1>;  
    <Câu lệnh 2>;  
    ...  
    <Câu lệnh n>;  
}
```

---

**Ví dụ 5.1:** Viết chương trình tính 5!.

---

```
/* Thông tin CT & TG  
Chương trình tính 5!  
Họ tên:  
MSSV:  
Lớp:
```

---

---

```

Ngay:
*/

//Khai bao thu vien
#include<iostream>
using namespace std;

//Ham chinh
void main()
{
    int iNum = 5;
    int iGiaiThua = 1;

    while(iNum > 0)
    {
        iGiaiThua = iGiaiThua * iNum;
        iNum--;    //Chu y su dung toan tu giam
    }

    cout<<" 5! = "<<iGiaiThua<<endl;
}

```

---

Chuyện gì sẽ xảy ra nếu ta bỏ đi câu lệnh `iNum--` trong đoạn lệnh trên? Giá trị của `iNum` sẽ luôn là 5. Điều này có nghĩa là biểu thức `iNum > 0` luôn đúng. Vì vậy, kết quả là một **vòng lặp vô hạn**, nghĩa là một khối lệnh sẽ lặp mãi mãi. Người ta phải rất cẩn thận khi sử dụng vòng lặp để đảm bảo rằng vòng lặp sẽ kết thúc.

**Ví dụ 5.2:** Viết chương trình cho phép người dùng nhập vào một ký tự khác x. Chương trình kết thúc khi người dùng nhập vào ký tự x.

---

```

/* Thông tin CT & TG
Chương trình nhập một ký tự khác x
Họ tên:
MSSV:
Lớp:
Ngay:
*/

//Khai bao thu vien
#include<iostream>
using namespace std;

//Ham chinh
void main()
{

```

---

---

```

char cLetter = 'a';

while(cLetter != 'x')
{
    cout<<"Hay nhap mot ky tu: "<<endl;
    cin>>cLetter;
    cout<<"Ky tu ban nhap la: "<<cLetter<<endl;
}

```

---

### III. Các bộ đếm

Thông thường lập trình viên cần kiểm soát số lần lặp lại cụ thể của một vòng lặp. Một cách chung nhất để thực hiện việc này là sử dụng **bộ đếm**. Giả sử bạn muốn tính trung bình của 5 cột điểm kiểm tra. Đầu tiên bạn phải nhập và cộng năm điểm. Điều này có thể được thực hiện với một **bộ đếm-kiểm soát** như trong ví dụ sau:

**Ví dụ 5.3:** Viết chương trình cho phép người dùng nhập vào điểm của 5 bài kiểm tra. Hãy tính và xuất lên màn hình điểm trung bình của 5 bài kiểm tra vừa nhập.

---

```

/* Thông tin CT & TG
Chương trình tính điểm trung bình 5 bài kiểm tra
Họ tên:
MSSV:
Lop:
Ngày:
*/

//Khai báo thư viện
#include<iostream>
using namespace std;

const int NUMBEROFTTEST = 5;

//Hàm chính
void main()
{
    int iScore;    //Lưu điểm được nhập
    float fpTotal = 0.0; //Tong cac diem so
    float fpAverage; //Trung binh
    int iTest = 1; //Bo dem kiểm soát vòng lặp

    //Ban đầu biểu thức được kiểm tra
    while(iTest <= NUMBEROFTTEST) //iTest là 1

```

---

---

```

    {
        cout<<"Hay nhap diemkiem tra "<<iTest<<" : "<<endl;
        cin>>iScore;

        fpTotal = fpTotal + iScore;
        iTest++; //Tang bien dem, iTest tang 1
    }
    fpAverage = fpTotal/NUMBEROFTTEST;

    cout<<"Diem trung binh tinh tren "<<NUMBEROFTTEST
        <<" diemkiem tra la: "<<fpAverage<<endl;
}

```

---

Chú ý rằng biến `iTest` được sử dụng như là bộ đếm kiểm soát vòng lặp. Cũng chú ý rằng hằng số `NUMBEROFTTEST` có thể dễ dàng thay đổi nếu muốn tính trung bình của số lượng bài kiểm tra khác.

#### IV. Giá trị lính canh

Chúng ta cũng có thể kiểm soát việc thực thi của vòng lặp bằng cách sử dụng một **giá trị lính canh**. Một giá trị đặc biệt được sử dụng để đánh dấu việc kết thúc một vòng lặp. Trong ví dụ 5.3, nếu chúng ta không biết chính xác có bao nhiêu điểm số cần nhập, chúng ta có thể thực hiện việc nhập điểm và cộng vào `fpTotal` cho đến khi nào nhập vào một giá trị lính canh. Hãy xem ví dụ dưới đây, giá trị lính canh là -1 vì nó là một giá trị điểm số không hợp lệ.

**Ví dụ 5.4:** Viết chương trình cho phép người dùng nhập vào điểm của 5 bài kiểm tra. Hãy tính và xuất lên màn hình điểm trung bình của 5 bài kiểm tra vừa nhập.

---

```

/* Thông tin CT & TG
Chương trình tính diem trung binh 5 baikiem tra
Ho ten:
MSSV:
Lop:
Ngay:
*/

//Khai bao thu vien
#include<iostream>
using namespace std;

```

---

---

```
//Ham chinh
void main()
{
    int iScore;    //Luu diem duoc nhap
    float fpTotal = 0.0; //Tong cac diem so
    float fpAverage; //Trung binh
    int iTest = 1; //Bo dem kiem soat vong lap

    cout<<"Hay nhap diem kiem tra "<<iTest<<" hoac -1 de
                                   ket thuc: "<<endl;
    cin>>iScore; //Doc ve diem lan 1

    //trong khi khong nhap vao gia tri linh
    //canh (ket thuc) thi thuc hien vong lap
    while(iScore != -1)
    {
        fpTotal = fpTotal + iScore;
        iTest++;
        cout<<"Hay nhap diem kiem tra "<<iTest<<" hoac -1
                                   de ket thuc: "<<endl;
        cin>>iScore; //Doc cac diem tiep theo
    }
    if(iTest > 1) //Neu iTest = 1 khong co diem duoc nhap
    {
        fpAverage = fpTotal/(iTest - 1);

        cout<<"Diem trung binh tinh tren "<< (iTest - 1)
        <<" diem kiem tra la: "<<fpAverage<<endl;
    }
}
```

---

Chú ý chương trình sẽ yêu cầu nhập ngay trước khi bắt đầu câu lệnh lặp while và lặp lại việc nhập này bên trong while. Việc này được thực hiện để câu lệnh while có thể kiểm tra giá trị lính canh.

## V. Bài tập:

**Bài 1: (Tìm giá trị lớn nhất)** Viết chương trình cho phép người dùng nhập vào 10 số nguyên. Tìm và xuất lên màn hình giá trị lớn nhất của 10 số vừa nhập. Chương trình chỉ sử dụng 3 biến sau:

*counter*: một bộ đếm từ 1 tới 10 (đếm số lượng giá trị đã nhập và xử lý)

*number*: giá trị được nhập hiện tại của chương trình

*largest*: số lớn nhất được tìm thấy

**Bài 2: (Tìm giá trị lớn nhất – cải tiến)** Cải tiến chương trình trên để tìm hai số lớn nhất của 10 số. [**Chú ý:** bạn chỉ được nhập mỗi giá trị một lần.]

**Bài 3: (Tính trung bình số Km/lít)** Các tài xế thường quan tâm đến số quãng đường đi được với chiếc xe của họ. Một lái xe đã theo dõi một vài bình xăng bằng cách ghi lại số km đã đi và số xăng đã đổ vào bình xăng. Hãy viết chương trình nhập vào số km đi được và số lít xăng đã đổ vào bình. Chương trình sẽ tính toán và hiển thị số km trên một lít xăng với mỗi bình xăng. Sau khi thực hiện tất cả thông tin nhập, chương trình tính và xuất lên số km trên mỗi lít đạt được cho tất cả các bình xăng. Dưới đây là một ví dụ:

**Hay nhập vào số lít xăng đã dùng (-1 để kết thúc): 12.8**

**Hay nhập vào số km đã chạy: 287**

**Số km/lít cho bình xăng này là: 22.421875**

**Hay nhập vào số lít xăng đã dùng (-1 để kết thúc): 10.8**

**Hay nhập vào số km đã chạy: 200**

**Số km/lít cho bình xăng này là: 18.518518**

**Hay nhập vào số lít xăng đã dùng (-1 để kết thúc): 5**

**Hay nhập vào số km đã chạy: 120**

**Số km/lít cho bình xăng này là: 24.000000**

**Hay nhập vào số lít xăng đã dùng (-1 để kết thúc): -1**

**Trung bình là: 21.646797 km/lít**

**Bài 4: (Tính thu nhập cho nhân viên)** Một công ty hóa chất lớn trả tiền cho nhân viên bán hàng của mình trên cơ sở hoa hồng. Các nhân viên bán hàng nhận được \$200 mỗi tuần cộng với 9% doanh thu của họ trong tuần đó. Ví dụ: nhân viên bán hàng có doanh thu \$5000 hóa chất trong một tuần sẽ nhận \$200 cộng 9% của \$5000, hoặc tổng cộng là \$650. Viết chương trình nhập vào tổng doanh thu của mỗi nhân viên bán

hàng cho tuần trước và tính và hiển thị lên thu nhập của nhân viên đó. Sau đây là một số ví dụ kết quả của chương trình:

**Nhap doanh thu cua nhan vien (-1 de ket thuc): 5000.00**

**Luong cua nhan vien la: \$650**

**Nhap doanh thu cua nhan vien (-1 de ket thuc): 1234.56**

**Luong cua nhan vien la: \$311.11**

**Nhap doanh thu cua nhan vien (-1 de ket thuc): 1088.89**

**Luong cua nhan vien la: \$298.00**

**Nhap doanh thu cua nhan vien (-1 de ket thuc): -1**

**Bài 5: (Tìm ước số chung)** Viết chương trình cho phép người dùng nhập vào hai số nguyên, sau đó tính và xuất lên màn hình ước số chung lớn nhất của hai số vừa nhập.

**Nhap hai so nguyen(-1 de ket thuc): 12 28**

**Uoc so chung lon nhat: 4**

**Nhap hai so nguyen(-1 de ket thuc): 6 9**

**Uoc so chung lon nhat: 3**

**Nhap hai so nguyen(-1 de ket thuc): -1**

**Bài 6: (Rút gọn phân số)** Viết chương trình yêu cầu người dùng nhập vào một phân số, sau đó rút gọn phân số và hiển thị kết quả lên màn hình. Chương trình kết thúc khi người dùng nhập mẫu số bằng 0.

**Nhap mot phan so(mau so 0 de ket thuc): 6/12**

**Phan so rut gon: 1/2**



## BÀI THỰC HÀNH TUẦN 06 – BUỔI 01

### CÂU LỆNH LẶP *DO-WHILE*

#### ❖ Mục đích:

1. Làm việc với câu lệnh lặp do-while
2. Chuyển đổi giữa câu lệnh while và do-while

#### I. Câu lệnh lặp do-while

Câu lệnh lặp while là một lệnh lặp mà việc kiểm tra điều kiện là **kiểm tra trước (pre-test) hay kiểm tra trên (top test)**. Vì chúng ta kiểm tra biểu thức trước khi vào vòng lặp, nếu biểu thức trong while ngay từ đầu là sai, thì các câu lệnh trong while sẽ không được thực thi. Nếu lập trình viên muốn lặp lại các câu lệnh ít nhất một lần, thì một vòng lặp mà việc **kiểm tra sau (post-test) hay kiểm tra dưới (bottom test)** nên được dùng. C++ cung cấp câu lệnh lặp do-while cho mục đích này. Một câu lệnh lặp do-while tương tự như câu lệnh lặp while ngoại trừ các câu lệnh bên trong do-while được thực thi trước khi biểu thức được kiểm tra. Cú pháp của do-while như sau:

---

```
do
{
    <Câu lệnh 1>;
    <Câu lệnh 2>;
    ...
    <Câu lệnh n>;
}
while(<biểu thức>;
```

---

Chú ý rằng các câu lệnh trong do-while phải được thực thi ít nhất một lần cho dù biểu thức là sai. Hãy so sánh kết quả của hai đoạn lệnh sau:

---

```
int iNum1 = 5;
int iNum2 = 7;
while (iNum2 < iNum1)
{
    iNum1 = iNum1 + 1;
    iNum2 = iNum2 - 1;
}
```

---

Ở đây cả hai câu lệnh  $iNum1 = iNum1 + 1$  và  $iNum2 = iNum2 - 1$  không bao giờ được thực thi vì kiểm tra biểu thức  $iNum2 < iNum1$  ngay từ đầu đã sai. Tuy nhiên kết quả sẽ khác khi sử dụng câu lệnh do-while:

---

```
int iNum1 = 5;
int iNum2 = 7;
do
{
    iNum1 = iNum1 + 1;
    iNum2 = iNum2 - 1;
}
while (iNum2 < iNum1);
```

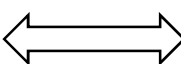
---

Trong đoạn lệnh này các câu lệnh  $iNum1 = iNum1 + 1$  và  $iNum2 = iNum2 - 1$  được thực thi chính xác một lần. Vào lúc này  $iNum1 = 6$  và  $iNum2 = 6$  vì vậy biểu thức  $iNum2 < iNum1$  là sai. Do đó, chương trình thoát khỏi vòng lặp và thực thi tiếp các lệnh tiếp theo.

## II. Chuyển đổi giữa câu lệnh *while* và *do-while*

Trong quá trình lập trình, lập trình viên có thể chuyển đổi qua lại giữa câu lệnh lặp while và do-while như sau:

---

<pre>do {     &lt;Câu lệnh&gt;; } while(&lt;biểu thức&gt;;</pre>		<pre>&lt;Câu lệnh&gt;; while(&lt;biểu thức&gt;) {     &lt;Câu lệnh&gt;; }</pre>
--	---	---

---

Ví dụ 6.1: Viết chương trình cho phép người dùng nhập vào một ký tự khác x. Chương trình kết thúc khi người dùng nhập vào giá trị là x.

---

```
/* Thông tin CT & TG
Chương trình nhập một ký tự khác x
Họ tên:
MSSV:
Lớp:
Ngày:
*/

//Khai báo thu vien
```

---

---

```
#include<iostream>
using namespace std;

//Ham chinh
void main()
{
    char cLetter;

    cout<<"Hay nhap mot ky tu: "<<endl;
    cin>>cLetter;
    cout<<"Ky tu ban nhap la: "<<cLetter<<endl;

    while(cLetter != 'x')
    {
        cout<<"Hay nhap mot ky tu: "<<endl;
        cin>>cLetter;
        cout<<"Ky tu ban nhap la: "<<cLetter<<endl;
    }
}
```

---

Chương trình trên có thể được chuyển sang câu lệnh do-while như sau:

---

```
/* Thông tin CT & TG
Chương trình nhập một ký tự khác x
Họ tên:
MSSV:
Lớp:
Ngày:
*/
//Khai báo thư viện
#include<iostream>
using namespace std;
//Ham chinh
void main()
{
    char cLetter;

    do
    {
        cout<<"Hay nhap mot ky tu: "<<endl;
        cin>>cLetter;
        cout<<"Ky tu ban nhap la: "<<cLetter<<endl;
    }
    while(cLetter != 'x');
}
```

---

### III. Bài tập:

Các bài tập sau đây phải được viết dưới hai câu lệnh lặp **while** và **do-while**

**Bài 1: (Đếm số ký số)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương. Hãy cho biết số vừa nhập có bao nhiêu ký số?

*Ví dụ:*

**Hay nhập số nguyên dương (-1 để kết thúc): 345**

**Số 345 có 3 ký số!**

**Hay nhập số nguyên dương (-1 để kết thúc): 12345**

**Số 12345 có 5 ký số!**

**Hay nhập số nguyên dương (-1 để kết thúc): -1**

**Bài 2: (Tính giá trị đảo ngược)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương. Chương trình tính và xuất lên màn hình giá trị đảo ngược của số vừa nhập.

*Ví dụ:*

**Hay nhập số nguyên dương (-1 để kết thúc): 345**

**Giá trị đảo ngược là: 543**

**Hay nhập số nguyên dương (-1 để kết thúc): 5378**

**Giá trị đảo ngược là: 8735**

**Hay nhập số nguyên dương (-1 để kết thúc): 937492**

**Giá trị đảo ngược là: 294739**

**Hay nhập số nguyên dương (-1 để kết thúc): 23**

**Giá trị đảo ngược là: 32**

**Hay nhập số nguyên dương (-1 để kết thúc): 9**

**Giá trị đảo ngược là: 9**

**Hay nhập số nguyên dương (-1 để kết thúc): -1**

**Bài 3: (Kiểm tra đối xứng)** Một số được gọi là số đối xứng khi bạn đọc xuôi cũng giống như đọc ngược. Ví dụ các số sau đây là số đối xứng: 34543, 4555554,... Viết

chương trình nhập vào một số nguyên dương, sau đó cho biết số vừa nhập có phải là số đối xứng hay không.

**Hướng dẫn:** Sử dụng phép chia và chia lấy phần dư để tính giá trị đảo ngược. Nếu giá trị đảo ngược bằng giá trị ban đầu thì đó là số đối xứng.

Ví dụ:

**Hãy nhập số nguyên dương (-1 để kết thúc): 345**

**345 không phải số đối xứng!**

**Hãy nhập số nguyên dương (-1 để kết thúc): 343**

**343 là số đối xứng!**

**Hãy nhập số nguyên dương (-1 để kết thúc): 23432**

**23432 là số đối xứng!**

**Hãy nhập số nguyên dương (-1 để kết thúc): -1**

**Bài 4: (Tìm chữ số lớn nhất và nhỏ nhất)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương. Sau đó, chương trình tìm ký số lớn nhất và nhỏ nhất xuất hiện trong số vừa nhập.

Ví dụ:

**Hãy nhập số nguyên dương (-1 để kết thúc): 23432**

**Ký số lớn nhất: 4**

**Ký số nhỏ nhất: 2**

**Hãy nhập số nguyên dương (-1 để kết thúc): 549**

**Ký số lớn nhất: 9**

**Ký số nhỏ nhất: 4**

**Hãy nhập số nguyên dương (-1 để kết thúc): -1**

**Bài 5: (Xuất lịch của tháng)** Viết chương trình xuất lên màn hình lịch của một tháng. Người dùng nhập vào số ngày của tháng và ngày bắt đầu của tháng đó.

**CHƯƠNG TRÌNH XUẤT LÊN MÀN HÌNH LỊCH CỦA MỘT THÁNG**  
**Nhập số ngày của tháng: 31**

Nhap ngay bat dau cua thang (1=Chu nhat, 7=Thu bay): 3

		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

**Hướng dẫn:** Sử dụng vòng lặp để lặp lệnh in lên màn hình các giá trị  $i$  từ 1 tới  $N$ , với  $N$  là số ngày của tháng. Trong vòng lặp kiểm tra  $i$  có phải là ngày cuối cùng của tuần hay không, nếu đúng thì xuất lên màn hình ký tự xuống dòng.

**Bài 6: (Liệt kê ước số)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương. Hãy xuất lên màn hình tất cả các ước số của số nguyên dương vừa nhập.

**Bài 7: (Đếm số ước số)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương. Hãy cho biết số nguyên dương vừa nhập có bao nhiêu ước số.

**Bài 8: (Tìm số thỏa điều kiện)** Tìm và in lên màn hình tất cả các số nguyên trong phạm vi từ 10 đến 99 sao cho tích của 2 chữ số bằng 2 lần tổng của 2 chữ số đó.

**Bài 9: (Tìm bội số chung nhỏ nhất)** Tìm bội số chung nhỏ nhất của 2 số nguyên dương  $a$  và  $b$  nhập từ bàn phím.

**Bài 10: (Xuất dãy số Fibonacci)** Nhập  $n$ . In  $n$  số đầu tiên trong dãy Fibonacci.

$$a_0 = a_1 = 1$$

$$a_n = a_{n-1} + a_{n-2}$$

## BÀI THỰC HÀNH TUẦN 06 – BUỔI 02

### CÂU LỆNH LẶP *FOR*

#### ❖ Mục đích:

1. Làm việc với câu lệnh lặp for
2. Chuyển đổi giữa câu lệnh while và for

#### I. Câu lệnh lặp for

Câu lệnh lặp for thường được sử dụng trong các chương trình có sử dụng bộ đếm. Cú pháp của for như sau:

---

```
for(<Khởi tạo>; <Điều kiện>; <Bước nhảy>)  
{  
    <Câu lệnh 1>;  
    <Câu lệnh 2>;  
    ...  
    <Câu lệnh n>;  
}
```

---

Chú ý rằng có ba biểu thức được đặt bên trong dấu ngoặc của câu lệnh for, được chia cách bởi dấu chấm phẩy.

- **Biểu thức khởi tạo:** thường được sử dụng để khởi tạo giá trị ban đầu cho một bộ đếm. Biểu thức này được thực hiện đầu tiên của vòng lặp và chỉ được thực hiện duy nhất một lần.
- **Biểu thức điều kiện:** giống như while và do-while, được sử dụng để kiểm soát việc thực thi của vòng lặp. Nếu biểu thức điều kiện là đúng, thân của vòng lặp sẽ được lặp lại. Câu lệnh for là câu lệnh lặp kiểm tra trước nghĩa là biểu thức điều kiện được kiểm tra trước mỗi lần lặp.
- **Biểu thức bước nhảy:** được thực thi tại cuối của mỗi lần lặp. Nó thường tăng hoặc giảm giá trị của bộ đếm.

Ví dụ 6.2: Viết chương trình tính trung bình cộng các giá trị từ 1 đến N, với N nhập vào từ bàn phím.

---

```
/* Thông tin CT & TG
Chương trình tính trung bình các giá trị từ 1 đến N
Họ tên:
MSSV:
Lớp:
Ngày:
*/

//Khai báo thư viện
#include <iostream>
using namespace std;
void main()
{
    //Khai báo biến
    int iN;
    int iTotal = 0;
    int iNumber;
    float fpMean;

    //Nhập liệu
    cout << "Hãy nhập một số nguyên dương: " << endl;
    cin >> iN;

    //Tính toán
    if (iN > 0)
    {
        for (iNumber = 1; iNumber <= iN; iNumber++)
        {
            iTotal = iTotal + iNumber;
        }
        fpMean = float(iTotal) / iN;
        cout << "Giá trị trung bình của " << iN << " số
        nguyên dương đầu tiên là: " << fpMean << endl;
    }
    else
        cout << "Giá trị nhập không hợp lệ." << endl;
}
```

---

Trong đoạn lệnh trên bộ đếm trong vòng lặp for là iNumber. Bộ đếm này tăng từ 1 đến iN trong suốt quá trình thực thi.

Trong lập trình, lập trình viên có thể không viết đầy đủ ba biểu thức trong ngoặc của câu lệnh for.



- Câu lệnh for có thể **không có biểu thức khởi đầu**, vì việc khởi tạo giá trị ban đầu cho bộ đếm có thể được thực hiện bên trên for.

---

```
if (iN > 0)
{
    iNumber = 1;
    for (; iNumber <= iN; iNumber++)
    {
        iTotal = iTotal + iNumber;
    }
    fpMean = float(iTotal) / iN;
    cout << "Gia tri trung binh cua " << iN
        << " so nguyen duong dau tien la: " << fpMean
    << endl;
}
```

---

Chú ý rằng dấu chấm phẩy trong câu lệnh lặp for không được bỏ đi.

- Trong câu lệnh for có thể **không có biểu thức điều kiện**. Lúc này, C++ mặc định điều kiện luôn luôn đúng. Do đó, để kết thúc việc lặp trong thân của for phải kiểm tra điều kiện và dùng câu lệnh break để thoát khỏi for.

---

```
if (iN > 0)
{
    iNumber = 1;
    for (; ; iNumber++)
    {
        iTotal = iTotal + iNumber;
        if(iNumber == iN)
            break;
    }
    fpMean = float(iTotal) / iN;
    cout << "Gia tri trung binh cua " << iN
        << " so nguyen duong dau tien la: " << fpMean
    << endl;
}
```

---

- Trong câu lệnh for có thể **không có bước nhảy**. Nghĩa là việc thay đổi giá trị của bộ đếm sẽ được thực hiện bên trong thân của câu lệnh for.

---

```
if (iN > 0)
{
```

---

---

```

    iNumber = 1;
    for (; ; )
    {
        iTotal = iTotal + iNumber;
        if(iNumber == iN)
            break;
        iNumber++;
    }
    fpMean = float(iTotal) / iN;
    cout << "Gia tri trung binh cua " << iN
         << " so nguyen duong dau tien la: " << fpMean << endl;
}

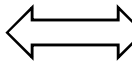
```

---

## II. Chuyển đổi giữa câu lệnh *while* và *for*

Trong quá trình lập trình, lập trình viên có thể chuyển đổi qua lại giữa câu lệnh lặp *while* và *do-while* như sau:

---

<pre> for(&lt;Khởi tạo&gt;;&lt;Điều kiện&gt;;&lt;Bước nhảy&gt;) {     &lt;Câu lệnh&gt;; } </pre>		<pre> &lt;Khởi tạo&gt;; while(&lt;Điều kiện&gt;) {     &lt;Câu lệnh&gt;;     &lt;Bước nhảy&gt;; } </pre>
--	--	--

---

Bạn có thể chuyển chương trình tính trung bình các giá trị từ 1 đến N trong mục trên sang câu lệnh *while* như sau:

### Ví dụ 6.3:

---

```

/* Thông tin CT & TG
Chương trình tính trung bình các giá trị từ 1 đến N
Họ tên:
MSSV:
Lop:
Ngày:
*/

//Khai báo thu viện
#include <iostream>
using namespace std;

//Hàm chính
void main()
{

```

---

---

```

//Khai bao bien
int iN;
int iTotat = 0;
int iNumber;
float fpMean;

//Nhap lieu
cout << "Hay nhap mot so nguyen duong: " << endl;
cin >> iN;

//Tinh toan
if (iN > 0)
{
    iNumber = 1;
    while(iNumber <= iN)
    {
        iTotat = iTotat + iNumber;
        iNumber++;
    }
    fpMean = float(iTotat) / iN;
    cout << "Gia tri trung binh cua " << iN
    << " so nguyen duong dau tien la: " << fpMean << endl;
}
else
    cout << "Gia tri nhap khong hop le." << endl;
}

```

---

### III. Bài tập:

Các bài tập sau đây phải được viết dưới hai câu lệnh lặp for và while

**Bài 1: (Tính các giá trị biểu thức)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương. Chương trình tính và xuất lên màn hình giá trị của biểu thức sau:

$$S_1 = \frac{1+2+3+\dots+N}{N} \qquad S_2 = \sqrt{1^2 + 2^2 + 3^2 + \dots + N^2}$$

$$S_3 = \sqrt{3 + \sqrt{3 + \sqrt{3 + \dots + \sqrt{3}}}} \text{ N dấu căn}$$

$$S_4 = \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \dots + \frac{1}{2}}}}} \text{ N dấu chia}$$

**Bài 2: (Đổi số thập phân sang nhị phân)** Viết chương trình nhập vào số nguyên dương N. Xác định chuỗi nhị phân biểu diễn cho N và xuất kết quả lên màn hình.

Ví dụ:

**Hay nhập số nguyên dương: 5**

**Chuỗi nhị phân tương ứng: 0101**

**Bài 3: (Đổi số nhị phân sang thập phân)** Viết chương trình nhập vào một số nguyên dương chỉ gồm các ký số 0 và 1 (nghĩa là số nhị phân) và xuất lên màn hình giá trị nguyên tương ứng.

**Hay nhập số nhị phân: 1101**

**Giá trị nguyên tương ứng: 13**

**Bài 4: (Xác định các giá trị căn chẵn)** Viết chương trình cho phép người dùng nhập một số nguyên dương N. Sau đó xuất lên màn hình tất cả các giá trị căn chẵn từ 1 tới N.

Ví dụ:

**Hay nhập số nguyên dương: 100**

**Các giá trị cơ bản chẵn là:**

**4**

**16**

**36**

**64**

**100**

**Bài 5: (Đếm số chữ số 7 có trong một số)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương. Sau đó, xác định xem có bao nhiêu ký số 7 trong số vừa nhập.

**Bài 6: (Liệt kê ước số lẻ)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương. Hãy xuất lên màn hình tất cả các ước số lẻ của số nguyên dương vừa nhập.

**Bài 7: (Đếm số ước số lẻ)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương. Hãy cho biết số nguyên dương vừa nhập có bao nhiêu ước số lẻ.

**Bài 8: (Tính tổng các ước số chẵn)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương. Hãy tính tổng tất cả các ước số chẵn của số nguyên dương vừa nhập.

**Bài 9: (Tính tổng các ước số)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương  $n$ . Hãy tính tổng các ước số của  $n$  mà nhỏ hơn chính  $n$ .

**Bài 10: (Kiểm tra số hoàn thiện)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương  $n$ . Hãy cho biết số  $n$  vừa nhập có phải là số hoàn thiện hay không? Biết số hoàn thiện là số có tổng các ước số nhỏ hơn nó (không tính số đó) bằng với chính nó.

Ví dụ: các số sau là số hoàn hảo: 6, 28, ...

# BÀI THỰC HÀNH TUẦN 07 – BUỔI 01

## CÁC CÂU LỆNH THOÁT KHỎI VÒNG LẶP

### ❖ Mục đích:

1. Làm việc với các câu lệnh `break` và `continue`
2. Kiểm tra dữ liệu

### I. Các câu lệnh `break` và `continue`

#### 1. Câu lệnh `break`

Câu lệnh `break` cũng có thể được sử dụng để nhảy ra ngoài vòng lặp `while`, `for` và `do-while`. Câu lệnh `break` đặc biệt được sử dụng cho việc viết các vòng lặp mà vị trí thoát là ở giữa thân vòng lặp hơn là ở vị trí bắt đầu hoặc ở cuối vòng lặp.

---

```
for(;;)
{
    cout<<"Nhập vào một số nguyên: "<<endl;
    cin>>iNhap;
    if(iNhap == 0) //Nhập vào số 0
        break; //Thoát khỏi vòng lặp
    iSum += iNhap; //Giá trị nhập là số dương
}
```

---

#### 2. Câu lệnh `continue`

Câu lệnh `break` chuyển điều khiển thẳng tới điểm kết thúc một vòng lặp, trong khi `continue` chuyển điều khiển tới điểm trước khi kết thúc thân vòng lặp. Với `break`, điều khiển sẽ ra khỏi vòng lặp, với `continue`, điều khiển vẫn còn ở trong vòng lặp. Có sự khác biệt giữa `break` và `continue`: `break` có thể sử dụng được trong `switch` và các câu lệnh lặp (`for`, `while`, `do-while`), còn `continue` chỉ giới hạn với các câu lệnh lặp.

---

```
for(;;)
{
    cout<<"Nhập vào một số nguyên: "<<endl;
    cin>>iNhap;
    if(iNhap < 0) //Nhập vào số âm
        continue; // quay lại nhập lại
}
```

---

---

```

        if(iNhap == 0) //Nhap vao so 0
            break; //Thoat khoi vong lap
        iSum += iNhap;
        //continue nhảy tới đây
    }
    //break nhảy tới đây (ngoài vòng lặp)

```

---

## II. Kiểm tra dữ liệu

Một trong những ứng dụng hiệu quả của vòng lặp đó là kiểm tra dữ liệu. Người dùng có thể nhập dữ liệu (từ bàn phím hoặc tập tin) và sau đó vòng lặp sẽ kiểm tra xem giá trị có hợp lệ hay không. Nếu giá trị là hợp lệ vòng lặp sẽ bỏ qua các lệnh trong vòng lặp nhưng với giá trị không hợp lệ chương trình sẽ yêu cầu người dùng nhập giá trị mới (hợp lệ).

### Ví dụ 7.1:

---

```

/* Thông tin CT & TG
Chương trình cho phép người dùng chọn đồ uống
Họ tên:
MSSV:
Lớp:
Ngày:
*/

//Khai báo thu vien
#include <iostream>
using namespace std;

//Ham chinh
void main()
{
    //Khai báo biến
    int iChon = 0;

    cout<<"CHUONG TRINH CHON DO UONG"<<endl;
    cout<<"Hay nhap do uong ban chon"<<
        "bang mot gia tri tu 1 toi 4"<<endl;
    cout<<"1 - Coffee"<<endl;
    cout<<"2 - Tea"<<endl;
    cout<<"3 - Coke"<<endl;

```

---

---

```
cout<<"4 - Orange juice"<<endl;

//Nhập liệu
do
{
    cout << "Hay nhập một số nguyên dương: " << endl;
    cin >> iChon;
    if(iChon < 1 || iChon > 4)
        cout<<"Giá trị hợp lệ là 1 -> 4! Hay nhập  
lại:"<<endl;
}while(iChon < 1 || iChon > 4);

//Tính toán
switch(iChon)
{
case 1:
    cout<<"Bạn vừa chọn Coffee"<<endl;
    break;
case 2:
    cout<<"Bạn vừa chọn Tea"<<endl;
    break;
case 3:
    cout<<"Bạn vừa chọn Coke"<<endl;
    break;
case 4:
    cout<<"Bạn vừa chọn Orange juice"<<endl;
    break;
}
}
```

---

### III. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức tương ứng với bài muốn thực hiện.

**Bài 1:** (*Tính tổng các số lẻ theo N*) Viết chương trình cho phép người dùng nhập vào một số nguyên dương N. Chương trình tính và xuất lên màn hình giá trị của biểu thức sau:

$$S = 1 + 3 + 5 + \dots + (2*N + 1)$$



**Bài 2: (Tính tổng các phân số theo N)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương N. Chương trình tính và xuất lên màn hình giá trị của biểu thức sau:

$$S_5 = \frac{1}{2} + \frac{3}{4} + \frac{5}{6} + \dots + \frac{2n+1}{2n+2}$$

**Bài 3: (Tính tổng các phân số âm dương xen kẽ theo N)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương N và một số nguyên x. Chương trình tính và xuất lên màn hình giá trị của biểu thức sau:

$$S = \frac{x}{1} - \frac{x^2}{2} + \frac{x^3}{3} - \dots + (-1)^{N-1} \frac{x^N}{N}$$

**Bài 4: (Đếm số lượng âm, dương)** Viết chương trình nhập vào các số nguyên hoặc nhập 0 để kết thúc. Chương trình cần đếm số lượng các số dương và số lượng các số âm có trong dãy số vừa nhập.

**Bài 5: (Đếm số lượng chẵn, lẻ)** Viết chương trình nhập vào một số nguyên dương N. Chương trình cần đếm số lượng các số chẵn và số lượng các số lẻ có trong N vừa nhập.

**Bài 6: (Đếm số lượng ước số)** Viết chương trình nhập vào một số nguyên dương N. Chương trình cần đếm số lượng các ước số của N vừa nhập.

**Bài 7: (Kiểm tra số nguyên tố)** Viết chương trình cho phép người dùng nhập một số nguyên dương N. Chương trình kiểm tra N có phải là số nguyên tố hay không? Biết số nguyên tố là số lớn hơn 1, chỉ chia hết cho 1 và cho chính nó (nghĩa là có đúng hai ước số). Bạn có thể sử dụng chương trình đếm số lượng ước số ở trên để kiểm tra số nguyên tố.

Ví dụ: các số sau là số nguyên tố: 2, 3, 5, 7, 11,...

**Bài 8: (Kiểm tra số hoàn hảo)** Cho số nguyên dương  $n$ . Hãy tìm giá trị nguyên dương  $k$  lớn nhất sao cho  $S(k) < n$ . Trong đó, chuỗi  $k$  được định nghĩa như sau:

$$S(k) = 1 + 2 + 3 + \dots + k$$

**Bài 9: (Tìm số âm lớn nhất)** Viết chương trình cho phép người dùng nhập vào các số nguyên hoặc nhập 0 để kết thúc. Tìm và xuất lên màn hình số âm lớn nhất trong dãy các số vừa nhập.

**Bài 10: (Tìm số dương nhỏ nhất)** Viết chương trình cho phép người dùng nhập vào các số nguyên hoặc nhập 0 để kết thúc. Tìm và xuất lên màn hình số dương nhỏ nhất trong dãy các số vừa nhập.

## BÀI THỰC HÀNH TUẦN 07 – BUỔI 02

### CÂU LỆNH LẶP LỒNG NHAU

#### ❖ Mục đích:

1. Làm việc với câu lệnh lặp lồng nhau
2. Nhận diện các bài toán có lặp lồng nhau

#### I. Câu lệnh lặp lồng nhau

Các lập trình viên có thể đặt một câu lệnh lặp bên trong một câu lệnh lặp khác, hay còn gọi là lặp lồng nhau. Để minh họa cho câu lệnh lặp lồng nhau, chúng ta hãy xem xét ví dụ sau:

##### Ví dụ 7.2:

---

```
/* Thông tin CT & TG
Chương trình xuất lên màn hình hình chu nhật các dấu *
Ho ten:
MSSV:
Lop:
Ngày:
*/
//Khai bao thu vien
#include <iostream>
using namespace std;

//Ham chinh
void main()
{
    //Khai bao bien
    int iWidth = 0;
    int iHeight = 0;
    cout<<"CHUONG TRINH VE HINH CHU NHAT"<<endl;
    //Nhap lieu
    do
    {
        cout << "Hay chieu rong: " << endl;
        cin >> iWidth;
        cout << "Hay chieu cao: " << endl;
        cin >> iHeight;
```

---

---

```

        if(iWidth < 0 || iHeight < 0)
            cout<<"Gia tri nhap phai nguyen duong! Hay
                                nhap lai."<<endl;
    }while(iWidth < 0 || iHeight < 0);
    //Tinh toan
    for(int i = 0; i< iHeight; ++i) //Voi moi hang
    {
        for(int j= 0; j < iWidth; ++j) //Voi mot cot
            cout<<"* ";
        cout<<endl;
    }
}

```

---

Ứng với mỗi hàng của hình chữ nhật, ta sẽ vẽ ra iWidth cột hình \*. Để duyệt qua mỗi hàng ta sử dụng vòng lặp bên ngoài với bộ đếm i. Để duyệt qua mỗi cột ta sử dụng vòng lặp bên trong với bộ đếm j.

Chúng ta cùng xem xét một ví dụ tính tổng theo công thức sau:

$S = 1 + (1 + 2) + (1 + 2 + 3) + \dots + (1 + 2 + \dots N)$  với N nhập từ bàn phím.

Ví dụ 7.3:

---

```

/* Thông tin CT & TG
Chương trình tính tổng s = 1 + (1+2) + ...+(1+2+...+ N)
Họ tên:
MSSV:
Lớp:
Ngày:
*/
//Khai báo thư viện
#include <iostream>
using namespace std;
//Hàm chính
void main()
{
    //Khai báo biến
    int iN = 0;
    int iSum = 0;

    cout<<"CHƯƠNG TRÌNH TÍNH S=1+(1+2)+...+(1+2+...+N) "
        <<endl;

    //Nhập liệu

```

---

```

do
{
    cout << "Hay nhap mot so nguyen duong: " << endl;
    cin >> iN;
    if(iN < 1)
        cout<<"Gia tri nhap phai nguyen duong! Hay
                                nhap lai."<<endl;

    }while(iN < 1);
    //Tinh toan
    for(int i = 1; i <= iN; ++i) //tong cua i gia tri
    {
        //khoei tao tong cho moi bieu thuc con
        int iSubSum = 0;
        //cac gia tri tu 1 toi i
        for(int j = 1; j <= i; ++j)
            iSubSum += j;
        iSum += iSubSum;
    }
    cout<<"Tong la: "<<iSum<<endl;
}

```

## II. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức tương ứng với bài muốn thực hiện.

**Bài 1: (Bảng nhân)** Viết một chương trình sẽ tạo một bảng nhân với kích thước được nhập bởi người dùng. Ví dụ, một bảng có kích thước 4 sẽ có bốn hàng và bốn cột. Các hàng và cột sẽ được dán nhãn từ 1 đến 4. Mỗi ô trong bảng sẽ là tích của số hàng và số cột tương ứng, vì vậy giá trị ở vị trí tương ứng với hàng thứ ba và cột thứ tư sẽ là 12.

Ví dụ:

```

Hay nhap vao kích thước bảng tính (-1 để kết thúc): 4
1  2  3  4
1  2  3  4
2  4  6  8
3  6  9  12
4  8  12 16

```

**Bài 2: (Vẽ hình vuông đặc)** Viết chương trình cho phép người dùng nhập vào một số nguyên dương là chiều dài một cạnh của hình vuông. Chương trình xuất lên màn hình một hình vuông với các hình sao (\*). Kích thước hình vuông thuộc đoạn từ 1 đến 20.

Ví dụ:

**Hãy nhập kích thước hình vuông (-1 để kết thúc): 6**

```
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
```

**Hãy nhập kích thước hình vuông (-1 để kết thúc): -1**

**Bài 3: (Vẽ hình vuông rỗng)** Sửa lại chương trình trên để nó xuất lên màn hình hình vuông rỗng với kích thước  $> 2$ .

Ví dụ:

**Hãy nhập kích thước hình vuông (-1 để kết thúc): 6**

```
* * * * *
*           *
*           *
*           *
*           *
*           *
* * * * *
```

**Hãy nhập kích thước hình vuông (-1 để kết thúc): -1**

**Bài 4: (Vẽ hình tam giác dưới-trái)** Viết chương trình cho phép người dùng nhập một số nguyên dương xác định kích thước một tam giác. Chương trình sẽ hiển thị hình tam giác tương ứng với kích thước người dùng nhập.

Ví dụ:

**Hay nhập chiều cao tam giác (-1 để kết thúc): 7**

```

*
* *
* * *
* * * *
* * * * *
* * * * *
* * * * *

```

**Hay nhập chiều cao tam giác (-1 để kết thúc): -1**

**Bài 5:** (Vẽ hình tam giác trên-trái) Cải tiến chương trình trên. Chương trình sẽ hiển thị hình tam giác tương ứng với kích thước người dùng nhập.

Ví dụ: người dùng nhập kích thước là 7

**Hay nhập chiều cao tam giác (-1 để kết thúc): 7**

```

* * * * *
* * * *
* * * *
* * *
* *
*

```

**Hay nhập chiều cao tam giác (-1 để kết thúc): -1**

**Bài 6:** (Vẽ hình tam giác trên-phải) Cải tiến chương trình bài 6. Chương trình sẽ hiển thị hình tam giác tương ứng với kích thước người dùng nhập.

Ví dụ:

**Hay nhập chiều cao tam giác (-1 để kết thúc): 7**

```

* * * * *
  * * * *
    * * * *
      * * *
        * *
          *
            *

```

**Hãy nhập chiều cao tam giác (-1 để kết thúc): -1**

**Bài 7: (Vẽ hình tam giác dưới-phải)** Cải tiến chương trình bài 6. Chương trình sẽ hiển thị hình tam giác tương ứng với kích thước người dùng nhập.

Ví dụ:

**Hãy nhập chiều cao tam giác (-1 để kết thúc): 7**

```

          *
        * *
      * * *
    * * * *
  * * * * *
* * * * * *
* * * * * *

```

**Hãy nhập chiều cao tam giác (-1 để kết thúc): -1**

**Bài 8: (Vẽ cây thông)** Viết chương trình cho phép người dùng nhập vào chiều cao một cây thông N, với  $N > 7$ . Chương trình sẽ hiển thị hình cây thông tương ứng với chiều cao vừa nhập.

Ví dụ:

**Hãy nhập chiều cao cây thông (-1 để kết thúc): 9**



```

      *
    * * *
  * * * * *
* * * * * * *
* * * * * * * *
      * * *
      * * *
      * * *
* * * * * * * *

```

**Hay nhập chiều cao tam giác (-1 để kết thúc): -1**

**Bài 9:** (*Vẽ hình kim cương kích thước 9*) Viết chương trình xuất lên màn hình hình kim cương sau:

```

      *
    * * *
  * * * * *
* * * * * * *
* * * * * * * *
      * * * * *
      * * * * *
      * * *
      *

```

**Bài 10:** (*Vẽ hình kim cương kích thước N*) Sửa lại chương trình của bài 4 để cho phép người dùng nhập vào một số lẻ thuộc đoạn từ 1 đến 19 xác định số dòng trong hình kim cương. Chương trình sẽ hiển thị hình kim cương tương ứng với kích thước người dùng nhập.

## BÀI THỰC HÀNH TUẦN 08 – BUỔI 01

### BÀI THỰC HÀNH TỔNG HỢP 03

#### ❖ Mục đích:

1. Ôn tập cấu trúc lặp
2. Vận dụng cấu trúc lặp để cài đặt các bài toán liên quan trên máy tính

#### I. Câu lệnh *while*

Câu lệnh *while* được sử dụng cho vòng lặp mà biểu thức kiểm soát được kiểm tra trước khi thân vòng lặp được thực thi. Câu lệnh *while* có dạng:

---

```
while (<Biểu thức>
    <Câu lệnh>;
```

---

<Biểu thức> là biểu thức kiểm soát và nó cho kết quả đúng (true) hoặc sai (false). Nếu <biểu thức> là đúng (true), <câu lệnh> trong thân *while* được thực thi. Nếu <biểu thức> là sai (false), <câu lệnh> trong thân *while* được bỏ qua.

#### II. Câu lệnh *do-while*

Câu lệnh *do-while* được sử dụng khi bạn muốn biểu thức kiểm soát được kiểm tra sau khi câu lệnh trong thân vòng lặp được thực thi. Câu lệnh *do-while* có dạng:

---

```
do
{
    <Câu lệnh>;
}
while (<biểu thức>;
```

---

#### III. Câu lệnh *for*

Câu lệnh *for* là một ý tưởng cho các vòng lặp có biến đếm, nhưng nó không đủ linh hoạt để sử dụng cho những loại khác. Câu lệnh *for* có dạng:

---

```
for (<Khởi tạo>; <điều kiện>; <bước nhảy>)
{
    <Câu lệnh>;
}
```

---

Ví dụ 8.1: Viết chương trình tính tổng N số ngẫu nhiên thuộc đoạn [0, 99], với N được định nghĩa trước.

---

```
/* Thông tin CT & TG
Chương trình tạo số ngẫu nhiên
Họ tên:
MSSV:
Lop:
Ngày:
*/

//Khai báo thư viện
#include <iostream>
#include<time.h> //các hàm thời gian
#include<stdlib.h> //rand() và srand()
using namespace std;

//Định nghĩa số lượng số ngẫu nhiên
#define NUMOFRANDOM 4 //Bạn có thể thay đổi giá trị này

//Hàm chính
void main()
{
    //Khai báo biến
    char cContinue = 'Y';
    cout<<"CHƯƠNG TRÌNH TÍNH TỔNG CÁC SỐ NGAU NHIENT" <<endl;
    //Nhập liệu: chương trình tạo ra một giá trị ngẫu nhiên
    do
    {
        int iNumber, iSum=0;
        //Khởi tạo bộ tạo số ngẫu nhiên
        srand(time(0));
        for(int i = 0; i < NUMOFRANDOM; ++i)
        {
            iNumber = rand()%100; //Giá trị thuộc [0, 99]
            cout<<"Số ngẫu nhiên vừa tạo: "<<iNumber<<endl;
            iSum += iNumber;
        }
        cout<<"Tổng các số ngẫu nhiên: "<<iSum<<endl;
        //Hỏi người dùng có muốn thực hiện tiếp
        cout<<"Bạn có muốn tiếp tục (y/n)?: "<<endl;
        cin>>cContinue;
    }
    while(cContinue == 'y' || cContinue == 'Y');
```

---

Ví dụ 8.2: Viết chương trình tạo dãy N số ngẫu nhiên. Chương trình cho phép tạo lại dãy N số ngẫu nhiên giống như dãy đã tạo trước đó.

---

```
/* Thông tin CT & TG
Chương trình tạo số ngẫu nhiên
Ho ten:
MSSV:
Lop:
Ngày:
*/

//Khai báo thư viện
#include <iostream>
#include<time.h> //các hàm thời gian
#include<stdlib.h> //rand() và srand()

using namespace std;

//Định nghĩa số lượng số ngẫu nhiên
#define NUMOFRANDOM 4 //Bạn có thể thay đổi giá trị này

//Hàm chính
void main()
{
    //Khai báo biến
    char cContinue = 'Y';
    time_t seed= 0;

    cout<<"CHƯƠNG TRÌNH TÍNH TỔNG CÁC SỐ NGAU NHIENT"<<endl;

    //Nhập liệu: chương trình tạo ra một giá trị ngẫu nhiên
    do
    {
        int iNumber, iSum=0;
        //Khởi tạo và tạo số ngẫu nhiên
        srand(time(&seed));
        cout<<"Các số ngẫu nhiên tạo lần 1: "<<endl;
        for(int i = 0; i < NUMOFRANDOM; ++i)
        {
            iNumber = rand()%100; //Giá trị thuộc [0, 99]
            cout<<"Số ngẫu nhiên vừa tạo: "<<iNumber<<endl;
        }
        cout<<"Các số ngẫu nhiên tạo lần 2: "<<endl;
        //Khởi tạo lại và tạo số ngẫu nhiên như trên
        //Hãy thực thi chương trình khi không có lệnh sau
        //cho nhận xét kết quả thực thi
    } while(cContinue == 'Y');
```

---

---

```

srand(time(&seed));
for(int i = 0; i < NUMOFRANDOM; ++i)
{
    iNumber = rand()%100; //Gia tri thuoc [0, 99]
    cout<<"So ngau nhien vua tao: "<<iNumber<<endl;
}
//Hoi nguoi dung co muon thuc hien tiep
cout<<"Ban co muon tiep tuc (y/n)?: "<<endl;
cin>>cContinue;
}
while(cContinue == 'y' || cContinue == 'Y');
}

```

---

#### IV. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức tương ứng với bài muốn thực hiện.

**Bài 1:** (*Ngày sớm nhất*) Viết chương trình cho phép người dùng nhập vào các bộ ngày, tháng, năm. Người dùng sẽ nhập vào 0/0/0 để kết thúc. Hãy xác định ngày, tháng, năm sớm nhất theo lịch.

Ví dụ:

**Hay nhap ngay, thang, nam (0,0, 0 de ket thuc): 6 3 18**

**Hay nhap ngay, thang, nam (0,0, 0 de ket thuc): 17 5 17**

**Hay nhap ngay, thang, nam (0,0, 0 de ket thuc): 3 6 17**

**Hay nhap ngay, thang, nam (0,0, 0 de ket thuc): 0 0 0**

**17 5 2017 la ngay som nhat.**

**Bài 2:** (*Tính giá trị e*) Giá trị của hằng số  $e$  trong toán học có thể được biểu diễn như một chuỗi vô hạn:

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \dots$$

Viết chương trình tính sắp xỉ  $e$  bằng cách tính giá trị của biểu thức:

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \dots + \frac{1}{n!}$$

Với  $n$  là một số nguyên được nhập từ bàn phím.

**Bài 3: (Xuất các số nguyên tố)** Viết chương trình xuất lên màn hình các số nguyên tố thuộc đoạn từ 1 đến 100.000 và cho biết có bao nhiêu số là số nguyên tố thuộc đoạn trên.

**Bài 4: (Số sinh đôi)** Các số sinh đôi là các số nguyên tố mà khoảng cách giữa chúng là 2. Hãy in tất cả các cặp số sinh đôi  $< 1000$ .

**Bài 5: (Xuất các số hoàn hảo)** Hãy viết chương trình xuất lên màn hình các số hoàn hảo trong đoạn từ 1 đến 100.000.000 và cho biết có bao nhiêu số là số hoàn hảo thuộc đoạn trên.

**Bài 6: (Bộ ba Pythagorean)** Một tam giác vuông có tất cả các cạnh là các số nguyên. Tập hợp ba giá trị số nguyên của ba cạnh một tam giác vuông được gọi là bộ ba Pythagorean khi và chỉ khi Những bộ ba số này phải thỏa mãn quan hệ là tổng bình phương hai cạnh bằng bình phương của cạnh huyền. Hãy viết chương trình tìm tất cả các bộ ba Pythagorean mà không lớn hơn 500.

## BÀI THỰC HÀNH TUẦN 08 – BUỔI 02

### HÀM (FUNCTION)

#### ❖ Mục đích:

1. Giới thiệu khái niệm hàm void (không có giá trị trả về - thủ tục)
2. Làm việc với hàm void (không có giá trị trả về - thủ tục) mà không có tham số
3. Giới thiệu và làm việc với hàm không có giá trị trả về (thủ tục) mà có tham số truyền theo giá trị và truyền tham chiếu

#### I. Định nghĩa hàm

Cú pháp:

---

```
<KDL> <Tên hàm>([Danh sách tham số])
{
    <Các câu lệnh>;
    [return <Giá trị>/Biến/<Biểu thức>;]
}
```

---

Ví dụ 8.3: Chương trình minh họa hàm cơ bản

---

```
/* Thông tin CT & TG
Chương trình minh họa hàm cơ bản
Họ tên:
MSSV:
Lớp:
Ngày:
*/

//Khai báo thu vien
#include <iostream>
using namespace std;

//Định nghĩa hàm
//*****
// Tên hàm: TinhTong
// Tác vụ: Hàm thực hiện việc tính tổng 2 số nguyên
// Đầu vào: Không có
// Đầu ra: Không có
//*****
void TinhTong()
{
    //Khai báo biến
```

---

```

    int iNum1, iNum2, iSum;
    //Nhập dữ liệu
    cout << "Hay nhap so nguyen: ";
    cin >> iNum1;
    cout << "Hay nhap so nguyen: ";
    cin >> iNum2;
    //Tính toán
    iSum = iNum1 + iNum2;
    //Xuất kết quả
    cout << iNum1 << " + " << iNum2 << " = " << iSum;
}
//Ham chinh
void main()
{
    TinhTong(); //Lời gọi hàm
    system("pause");
}

```

## II. Truyền tham trị (Pass by value)

Ví dụ 8.4: Chương trình minh họa cách truyền tham trị

```

/* Thông tin CT & TG
Chương trình minh họa hàm cơ bản
Họ tên:
MSSV:
Lop:
Ngày:
*/

//Khai báo thu vien
#include <iostream>
using namespace std;

// Định nghĩa hàm
//*****
// Tên hàm: TinhTong
// Tác vụ: Hàm thực hiện việc tính tổng 2 số nguyên
// Dấu vào: 2 số nguyên
// Dấu ra: không
//*****
void TinhTong(int a, int b)
{
    int iSum;
    //Tính toán
    iSum = a + b;
    // Xuất kết quả
}

```



---

```

        cout << a << " + " << b << " = " << iSum;
    }
    //Ham chinh
    void main()
    {
        //Khai bao bien
        int iNum1, iNum2;
        //Nhap du lieu
        cout << "Hay nhap so nguyen: ";
        cin >> iNum1;
        cout << "Hay nhap so nguyen: ";
        cin >> iNum2;
        TinhTong(iNum1, iNum2); //Loi goi ham
        system("pause");
    }

```

---

### III. Truyền tham chiếu (pass by reference)

Ví dụ 8.4: Chương trình minh họa cách truyền tham chiếu

---

```

/* Thông tin CT & TG
Chương trình minh họa cách truyền tham chiếu
Họ tên:
MSSV:
Lớp:
Ngày:
*/

//Khai bao thu vien
#include <iostream>
using namespace std;

//Định nghĩa hàm
//*****
// Tên hàm: TinhTong
// Tác vụ: Hàm thực hiện việc tính tổng 2 số nguyên
// Đầu vào: 2 số nguyên
// Đầu ra: tổng (thay đổi tham số thực tương ứng)
//*****
void TinhTong(int a, int b, int &sum)
{
    //Tính toán
    sum = a + b;
}
//Hàm chính
void main()
{

```

---

---

```
//Khai bao bien
int iNum1, iNum2, iSum;
//Nhap du lieu
cout << "Hay nhap so nguyen: ";
cin >> iNum1;
cout << "Hay nhap so nguyen: ";
cin >> iNum2;
TinhTong(iNum1, iNum2, iSum);
//Xuat ket qua
cout << iNum1 << " + " << iNum2 << " = " << iSum;
system("pause");
}
```

---

#### IV. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.

**Bài 1: (Hoán vị hai số)** Viết chương trình cho phép người dùng nhập vào hai số thực (số thứ nhất lưu vào biến *first* và số thứ hai lưu vào biến *second*) và sau đó gọi hàm HoanVi (swap) với các tham số thực là *first* và *second*. Hàm HoanVi (swap) có các tham số hình thức là *number1* và *number2* sẽ hoán vị giá trị của hai biến. Các giá trị sau khi hoán vị sẽ được xuất trong hàm *main*.

Ví dụ:

**Hay nhập số thứ nhất, sau đó enter: 80**

**Hay nhập số thứ hai, sau đó enter: 70**

**Bạn nhập các số là 80 và 70.**

**Sau khi hoán vị, số thứ nhất có giá trị 70.**

**Số thứ hai có giá trị là 80.**

**Bài 2: (Tính tốc độ Km/h)** Viết chương trình nhập vào số Km và số giờ đã đi du lịch. Chương trình xác định số tốc độ Km trên một giờ. Việc tính toán này được thực hiện trong một hàm khác hàm *main*, hàm *main* sẽ xuất lên kết quả tính toán. Hàm tính

toán sẽ có ba tham số: Km, Hours và KmPerHour. Tham số nào được truyền tham trị và truyền tham chiếu? Kết quả xuất lên màn hình lấy chính xác đến hai số thập phân.

Ví dụ:

**Hay nhập so km đã đi du lịch:**

**475**

**Hay nhập số giờ đã đi du lịch:**

**8**

**Tốc độ của bạn là 59.38 km trên giờ**

**Bài 3:** (*Chuyển điểm số sang điểm chữ*) Viết chương trình cho phép người dùng nhập vào số lượng điểm số. Sau đó, chương trình cho người dùng nhập vào từng điểm số. Chương trình sẽ tính tổng của các điểm được nhập và truyền nó cùng với số lượng điểm tới một hàm có một tham số “truyền tham chiếu”. Hàm này sẽ tính điểm trung bình của các điểm được nhập. Hàm main sẽ xác định điểm chữ tương ứng với điểm trung bình trên theo quy đổi:

90 – 100	A
80 – 89	B
70 – 79	C
60 – 69	D
0 – 59	F

Ví dụ:

**Hay nhập số lượng điểm số:**

**3**

**Hay nhập điểm số thu 1 (0 – 100):**

**90**

**Hay nhập điểm số thu 2 (0 – 100):**

**80**

**Hay nhập điểm số thu 3 (0 – 100):**

## 50

**Diem chu tuong ung la: C**

**Bài 4: (Tính số giây đồng hồ)** Viết một hàm tính số giây theo giờ (gồm ba thông tin giờ, phút và giây) tính từ thời điểm cuối cùng của đồng hồ theo cấu trúc 24h. Sử dụng hàm này để tính khoảng cách theo giây giữa hai giờ cho trước, cả hai giờ đều nằm trong chu kỳ 24h đồng hồ. Hàm được mô tả như sau:

---

```
//*****  
// Ten ham: Seconds(int, int, int, int &  
// Tac vu: Ham nay tinh tong so giay  
// Dau vao: hours, minutes, seconds  
// Dau ra: TotalSeconds (thay doi tham so thuc tuong ung)  
//*****
```

---

**Bài 5: (Tính cạnh huyền)** Định nghĩa một hàm có tên Hypotenuse để tính chiều dài cạnh huyền của một tam giác vuông khi biết chiều dài hai cạnh còn lại. Hàm này được mô tả như sau:

---

```
//*****  
// Ten ham: Hypotenuse(double, double, double &  
// Tac vu: Ham nay tinh va xuat ra chieu dai canh huyen  
// Dau vao: Side1 va Side2  
// Dau ra: Hypotenuse (thay doi tham so thuc tuong ung)  
//*****
```

---

## BÀI THỰC HÀNH TUẦN 09 – BUỔI 01

### HÀM (FUNCTION)

#### ❖ Mục đích:

1. Giới thiệu khái niệm phạm vi
2. Phân biệt được sự khác nhau giữa các biến tĩnh (static), cục bộ và toàn cục
3. Giới thiệu hàm có giá trị trả về

#### I. Phạm vi

Phạm vi toàn cục: Một đối tượng có phạm vi toàn cục là đối tượng được khai báo bên ngoài tất cả các hàm.

Phạm vi cục bộ: Một đối tượng được định nghĩa bên trong một hàm hoặc trong cặp ngoặc nhọn có phạm vi từ vị trí khai báo đến dấu ngoặc nhọn phải “}” chứa nó.

##### Ví dụ 9.1:

---

```
/*Thông tin CT & TG
Chương trình minh họa phạm vi của biến
Ho ten:
MSSV:
Lop:
Ngày:
*/
//Khai bao thu vien
#include<iostream>
using namespace std;
//Khai bao bien toan cuc
const float PI = 3.14;

//Khai bao nguyen mau ham
void XuatTieuDe();

//Ham chinh
void main()
{
    float fpCircle;
    cout<<"fpCircle co pham vi cuc bo trong ham main."<<endl;
    {
        float fpSquare;
```

---

---

```

        cout<<"fpSquare có phạm vi cục bộ hoạt động chỉ "
            <<"trong một phần của hàm main."<<endl;
        cout<<"Ca fpSquare và fpCircle có thể được "
            <<"truy xuất ở đây cũng như "
            <<"hàng số toán cục PI."<<endl;
    }
    cout<<"fpCircle hoạt động ở đây, "
        <<"nhưng fpSquare thì không."<<endl;

    XuatTieuDe();

}
void XuatTieuDe()
{
    int iTriangle;
    cout<<"Hàng số toán cục PI hoạt động ở đây "
        <<"cũng như biến cục bộ iTriangle."<<endl;
}

```

---

## II. Đối số mặc định

Các tham số thực (các tham số được sử dụng trong lời gọi một hàm) thường được gọi là **các đối số (arguments)**. Bình thường số lượng các tham số thực hoặc các đối số phải bằng số lượng các tham số hình thức. Tuy nhiên, cũng có thể gán một giá trị mặc định cho tất cả các tham số hình thức để lời gọi hàm không phải truyền giá trị cho tất cả các đối số. Mặc dù những giá trị mặc định này có thể được xác định trong phần tiêu đề hàm, chúng thường được định nghĩa trong nguyên mẫu hàm.

### Ví dụ 9.2:

---

```

/*Thông tin CT & TG
Chương trình minh họa đối số mặc định
Ho ten:
MSSV:
Lop:
Ngày:
*/
//Khai báo thu viện
#include<iostream>
#include<iomanip>
using namespace std;

```

---

---

```
//Khai bao nguyen mau ham
void TinhLuongThucLanh(float &ThucLanh, int SoGio = 40,
float TyLe = 6.00);
//Nguyen mau ham voi doi so mac dinh

//Ham chinh
void main()
{
    int iSoGioLamViec = 20;
    float fpTyLeChiTra = 5.00;
    float fpLuong;
    cout<<setprecision(2)<<fixed<<showpoint;
    //Goi ham chi voi 1 tham so
    TinhLuongThucLanh(fpLuong);
    cout<<"Luong thuc lanh la: "<<fpLuong<<endl;
}
/*
- Ten ham: TinhLuongThucLanh
- Cong viec: Ham nay nhan ty le chi tra va so gio
               va nhan chung de nhan duoc luong thuc lanh.
               No co hai tham so mac dinh.
- Du lieu vao: Ty le chi tra va thoi gian lam viec
- Du lieu ra: Luong thuc lanh (thay doi tham so thuc tuong
ung)
*/
void TinhLuongThucLanh(float &ThucLanh, int SoGio, float
TyLe)
{
    ThucLanh = SoGio * TyLe;
}
```

---

Dưới đây là những lời gọi hàm hợp lệ trong ví dụ trên. Hãy điền các giá trị vào chỗ trống cho những lời gọi hàm sau:

*TinhLuongThucLanh(fpLuong);*

*TinhLuongThucLanh(fpLuong);*

Lương thực lãnh là: \_\_ *TinhLuongThucLanh* nhận giá trị \_\_ cho Số giờ và \_\_ cho tỷ lệ.

*TinhLuongThucLanh(fpLuong, iSoGioLamViec);*

Lương thực lãnh là: \_\_ *TinhLuongThucLanh* nhận giá trị \_\_ cho Số giờ và \_\_ cho tỷ lệ.

*TinhLuongThucLanh(fpLuong, iSoGioLamViec, fpTyLeChiTra);*

Lương thực lãnh là: \_\_ *TinhLuongThucLanh* nhận giá trị \_\_ cho Số giờ và \_\_ cho tỷ lệ.

### III. Hàm có giá trị trả về

Những hàm được thảo luận trong những phần trước không là “hàm đúng” bởi vì chúng không trả về một giá trị cho lời gọi hàm. Chúng thường được gọi là các thủ tục trong lĩnh vực khoa học máy tính. Những hàm đúng, hoặc những hàm có giá trị trả về, là những mô đun mà trả về chính xác một giá trị cho lời gọi hàm. Trong C++ để trả về một giá trị ta dùng câu lệnh *return*.

#### Ví dụ 9.3:

---

```

/*Thông tin CT & TG
Chương trình minh họa hàm có giá trị trả về
Ho ten:
MSSV:
Lop:
Ngày:
*/
//Khai bao thu vien
#include<iostream>
#include<iomanip>
using namespace std;

//Khai bao nguyen mau ham
int TinhLapPhuong(int x);

//Ham chinh
void main()
{
    int iX = 2;
    int iLapPhuong;

    iLapPhuong = TinhLapPhuong(iX); //Loi gọi hàm
    cout<<"Lap phuong cua "<<iX<<" la"<<iLapPhuong<<endl;

```

---



---

```

}
/*
- Ten ham: TinhLapPhuong
- Cong viec: Ham nay nhan nhan mot gia tri va
               tra ve gia tri lap phuong cua no
- Du lieu vao: Mot so nguyen x
- Du lieu ra: Lap phuong cua x
*/
int TinhLapPhuong(int x) //Chu y kieu cua ham la int khong phai void
{
    int Num;
    Num = x*x*x;
    return Num;
}

```

---

Hàm TinhLapPhuong nhận giá trị của x, trong trường hợp này là 2, và tính lập phương của nó sau đó đưa vào biến cục bộ Num. Sau đó hàm trả về giá trị được lưu trong Num cho hàm gọi TinhLapPhuong(x). Đó là, iLapPhuong = TinhLapPhuong(iX); sẽ đặt iLapPhuong = 8. Không cần thiết phải lưu giá trị trả về vào biến cục bộ trước khi trả nó về cho hàm gọi. Hàm TinhLapPhuong có thể được viết như sau

---

```

int TinhLapPhuong(int x)
{
    return x*x*x;
}

```

---

Vì các hàm trả về giá trị chúng ta sẽ thay từ void bằng kiểu dữ liệu của giá trị mà hàm trả về. Ví những hàm này trả về một giá trị, nên không có sự ảnh hưởng trên bất kỳ tham số nào mà được truyền từ lời gọi hàm. Điều này có nghĩa rằng tất cả các tham số của các hàm trả về giá trị sẽ được truyền tham trị, không truyền tham chiếu. C++ không cấm lập trình viên sử dụng truyền tham chiếu trong các hàm trả về giá trị. Tuy nhiên, chúng ta không nên sử dụng.

#### Ví dụ 9.4:

---

```

/*Thông tin CT & TG
Chuong trình tinh luong thuc lanh voi ham co gia tri tra ve

```

---

---

```

Ho ten:
MSSV:
Lop:
Ngày:
*/
//Khai bao thu vien
#include<iostream>
#include<iomanip>

using namespace std;

//Khai bao nguyen mau ham
float TinhLuongThucLanh(int SoGio, float TyLe);

//Ham chinh
void main()
{
    int iSoGioLamViec = 20;
    float fpTyLeChiTra = 5.00;
    float fpLuong;

    cout<<setprecision(2)<<fixed<<showpoint;
    fpLuong = TinhLuongThucLanh(iSoGioLamViec, fpTyLeChiTra);
    cout<<"Luong thuc lanh la: "<<fpLuong<<endl;

}
/*
- Ten ham: TinhLuongThucLanh
- Cong viec: Ham nay nhan ty le chi tra va so gio
                va nhan chung de nhan duoc luong thuc lanh
                va tra gia tri ve cho ham goi.
- Du lieu vao: Ty le chi tra va thoi gian lam viec
- Du lieu ra: Luong thuc lanh
*/
float TinhLuongThucLanh(int SoGio, float TyLe)
{
    return SoGio * TyLe;
}

```

---

Chú ý lời gọi hàm được thực trong ví dụ trên.

```
fpLuong = TinhLuongThucLanh(iSoGioLamViec, fpTyLeChiTra);
```

Lời gọi hàm không phải là câu lệnh đứng một mình mà là một phần của phép gán. Lời gọi hàm được sử dụng trong một biểu thức. Thực tế, hàm trả về một giá trị thực thay cho phía bên phải của câu lệnh gán.

#### IV. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.

**Bài 1:** (*Liệt kê số nguyên tố*) Nhập số nguyên dương  $n$  ( $n > 0$ ). Liệt kê tất cả các số nguyên tố nhỏ hơn  $n$ .

*Ví dụ:*

**Hãy nhập số nguyên dương: 11**

**Các số nguyên tố nhỏ hơn 11 là: 2 3 5 7**

**Bài 2:** (*Liệt kê số chính phương*) Nhập số nguyên dương  $n$  ( $n > 0$ ). Liệt kê  $n$  số chính phương đầu tiên.

*Ví dụ:*

**Hãy nhập số nguyên dương: 5**

**5 số chính phương đầu tiên: 4 9 16 25 36**

**Bài 3:** (*Đếm số lượng số hoàn hảo*) Nhập số nguyên dương  $n$  ( $n > 0$ ). Cho biết có bao nhiêu số hoàn thiện nhỏ hơn  $n$ .

*Ví dụ:*

**Hãy nhập số nguyên dương:**

**3**

**Số lượng số hoàn hảo nhỏ hơn 3: 0**

**Hãy nhập số nguyên dương:**

**36**

**Số lượng số hoàn hảo nhỏ hơn 36: 2**

**Bài 4: (Tam giác)** Nhập vào tọa độ 3 điểm A, B, C và kiểm tra xem chúng có phải là 3 đỉnh của một tam giác hay không? Nếu có hãy tính diện tích, chiều dài mỗi đường cao của tam giác và in kết quả ra màn hình.

- Công thức tính diện tích  $s = \sqrt{p(p-a)(p-b)(p-c)}$

- Công thức tính các đường cao:  $h_a = \frac{2s}{a}, h_b = \frac{2s}{b}, h_c = \frac{2s}{c}$

(Với  $p = \frac{a+b+c}{2}$ : là nửa chu vi của tam giác,  $a, b, c$  là chiều dài 3 cạnh)

**Bài 5: (Tìm ước chung và bội chung)** Viết chương trình nhập vào 2 số nguyên dương a và b. Tìm ước số chung lớn nhất và bội số chung nhỏ nhất của 2 số vừa nhập.

## BÀI THỰC HÀNH TUẦN 09 – BUỔI 02

### ĐỆ QUY

#### ❖ Mục đích:

1. Tìm hiểu khái niệm đệ quy
2. Giới thiệu lớp các bài toán giải được bằng đệ quy

#### I. Một số ví dụ

Ví dụ 9.5: Tìm ước chung lớn nhất của 2 số nguyên dương  $a, b$

---

```

/*Thông tin CT & TG
Chương trình tìm ước chung lớn nhất theo đệ quy
Ho ten:
MSSV:
Lop:
Ngày:
*/
//Khởi tạo thư viện
#include<iostream>
using namespace std;
//Khởi tạo nguyên mẫu hàm
int UCLN(int , int);

//Hàm chính
void main()
{
    int iA, iB, iUSCLN;;

    cout<<"CHƯƠNG TRÌNH TÌM ƯỚC SỐ CHUNG LỚN NHẤT"<<endl;
    //Nhập liệu
    cout<<"Hay nhập số nguyên thứ nhất: "<<
        endl;
    cin>>iA;
    cout<<"Hay nhập số nguyên thứ hai: "<<endl;
    cin>>iB;

    iUSCLN = UCLN(iA, iB);

    cout<<"Ước số chung lớn nhất của "<<iA<<" và "<<iB<<" là:
        "<<iUSCLN<<endl;
}

```

---

---

```

/*
- Ten ham: UCLN
- Cong viec: Ham nay nhan 2 so nguyen
                va tim uoc so chung lon nhat cua 2 so
- Du lieu vao: so nguyen a, so nguyen b
- Du lieu ra: Uoc so chung lon nhat
*/
int UCLN(int a, int b)
{
    if(a < b)
        UCLN(a, b - a);
    else
        if(a == b)
            return a;
        else
            if(a > b)
                UCLN(a - b, b);
}

```

---

Ví dụ 9.6: Tính giá trị thứ  $n$  của dãy Fibonacci:

$$f(0) = f(1) = 1$$

$$f(n) = f(n-1) + f(n-2), \forall n \geq 2$$

---

```

/*Thông tin CT & TG
Chương trình tính giá trị thứ n của dãy Fibonacci
Họ tên:
MSSV:
Lop:
Ngày:
*/
//Khai báo thư viện
#include<iostream>
using namespace std;
//Khai báo nguyên mẫu hàm
int Fibo(int);

//Hàm chính
void main()
{
    int iN, iFn;

    cout<<"CHƯƠNG TRÌNH TÍNH GIÁ TRỊ FIBONACCI"<<endl;
    //Nhập liệu

```

---

---

```

    cout<<"Hay nhap so nguyen: "<<endl;
    cin>>iN;

    iFn = Fibo(iN);

    cout<<"Fibonacci("<<iN<<" ) = "<<iFn<<endl;
}
/*
- Ten ham: Fibo
- Cong viec: Ham nay nhan 1 so nguyen
                va tim gia tri Fibonacci tuong ung
- Du lieu vao: so nguyen n
- Du lieu ra: Gia tri Fibonacci tuong ung
*/
int Fibo(int n)
{
    if(n==0 || n == 1)
        return n;
    else
        return Fibo(n - 1) + Fibo(n - 2);
}

```

---

## II. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.

**Bài 1:** (*Phép tính đệ quy*) Viết một hàm đệ quy  $\text{power}(\text{CoSo}, \text{SoMu})$  mà khi được gọi sẽ trả về  $\text{CoSo}^{\text{SoMu}}$ . Giả sử  $\text{SoMu}$  là một số nguyên dương và công thức tính như sau:

$$\text{CoSo}^1 = \text{CoSo}$$

$$\text{CoSo}^{\text{SoMu}} = \text{CoSo} * \text{CoSo}^{\text{SoMu}-1}$$

Ví dụ:

$$\text{power}(3,4) = 3*3*3*3$$

**Bài 2:** (*Tính S theo đệ quy*) Nhập số nguyên dương  $n$  ( $n > 0$ ). Viết các hàm đệ quy tính  $S$  theo công thức như sau:

$$S_1 = \frac{1+2+3+\dots+N}{N}$$

$$S_2 = \sqrt{1^2 + 2^2 + 3^2 + \dots + N^2}$$

$$S_3 = \sqrt{3 + \sqrt{3 + \sqrt{3 + \dots + \sqrt{3}}}} \text{ N dấu căn}$$

$$S_4 = \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \dots + \frac{1}{2}}}} \text{ N dấu chia}$$

**Bài 3: (Tính  $C(n, k)$ )** Viết hàm đệ quy tính  $n!$ . Sau đó, sử dụng hàm trên để tính tổ hợp chập  $k$  theo công thức:

$$C(n, k) = \frac{n!}{k!(n-k)!}$$

**Bài 4: (Tam giác Pascal)** Hiển thị  $n$  dòng của tam giác Pascal.

- $a[i][0] = a[i][i] = 1$
- $a[i][k] = a[i-1][k-1] + a[i-1][k]$

Dòng 0: 1

Dòng 1: 1 1

Dòng 2: 1 2 1

Dòng 3: 1 3 3 1

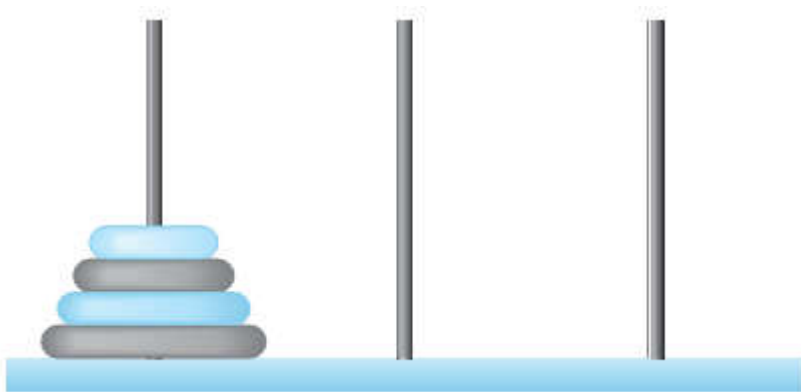
Dòng 4: 1 4 6 4 1

**Bài 5: (Tháp Hà Nội)** Bài toán tháp Hà Nội là một bài toán nổi tiếng có nội dung như sau: chuyển một tháp  $n$  đĩa đang xếp tại cột 1 sang cột 2 (cho phép sử dụng cột trung gian 3) theo điều kiện như sau:

- Mỗi lần chỉ được chuyển đĩa trên cùng của tháp



- Tại mọi thời điểm luôn đảm bảo rằng đĩa lớn hơn phải nằm dưới các đĩa nhỏ hơn.



Bài toán có thể được tổng quát như sau: chuyển tháp từ vị trí di đến vị trí den, trong đó di, den là các tham số có thể lấy giá trị là 1, 2, 3 đại diện cho 3 cột. Đối với 2 vị trí di và den, ta có thể tính ra vị trí trung gian (vị trí còn lại) là  $6 - di - den$  (vì  $di + den + tg = 1 + 2 + 3 = 6$ ). Từ đó để chuyển đĩa từ vị trí di đến vị trí den, ta có thể xây dựng một cách chuyển đệ quy như sau:

- Chuyển  $n - 1$  đĩa từ 1 sang 2, sử dụng 3 làm trung gian
- Chuyển đĩa cuối cùng (lớn nhất) từ 1 sang 3
- Chuyển  $n - 1$  đĩa từ 2 sang 3, sử dụng 1 làm trung gian

# BÀI THỰC HÀNH TUẦN 10 – BUỔI 01

## BÀI THỰC HÀNH TỔNG HỢP 04

### ❖ Mục đích:

1. Ôn tập các khái niệm về hàm, phạm vi của biến và các cách truyền tham số
2. Sử dụng hàm để cài đặt các chương trình trên máy tính

### I. Khái niệm hàm

Hàm là một chương trình con trong chương trình lớn. Hàm nhận (hoặc không) các đối số và trả lại (hoặc không) một giá trị cho chương trình gọi nó. Một chương trình là tập các hàm, trong đó có một hàm chính với tên gọi `main()`, khi chạy chương trình, hàm `main()` sẽ được chạy đầu tiên và gọi đến hàm khác. Kết thúc hàm `main()` cũng là kết thúc chương trình.

Hàm giúp cho việc phân đoạn chương trình thành những mô đun riêng rẽ, hoạt động độc lập với ngữ nghĩa của chương trình lớn, có nghĩa là một hàm có thể được sử dụng trong chương trình này mà cũng có thể được sử dụng trong chương trình khác, để cho việc kiểm tra và bảo trì chương trình. Hàm có một số đặc trưng:

- Nằm trong hoặc ngoài chương trình gọi đến hàm. Trong một chương trình có thể chứa nhiều hàm.
- Được gọi từ chương trình chính (`main`), từ hàm khác hoặc từ chính nó (đệ quy)
- Không lồng nhau.
- Có 2 cách truyền tham số: Truyền tham trị, tham chiếu.

*Cú pháp viết hàm:*

---

```
<kiểu dữ liệu> <Tên hàm>([Danh sách tham số])  
{  
    <câu lệnh>;  
    [return <giá trị>;]  
}
```

---

## II. Phạm vi của biến

Phạm vi toàn cục: Một đối tượng có phạm vi toàn cục là đối tượng được khai báo bên ngoài tất cả các hàm.

Phạm vi cục bộ: Một đối tượng được định nghĩa bên trong một hàm hoặc trong cặp ngoặc nhọn có phạm vi từ vị trí khai báo đến dấu ngoặc nhọn phải “}” chứa nó.

## III. Các cách truyền tham số

Truyền tham trị (pass by value): Theo cơ chế truyền tham số này, khi thực hiện lời gọi hàm giá trị của các tham số thực sẽ được *sao chép* để truyền cho các tham số hình thức tương ứng. Các lệnh trong hàm được gọi có thể thay đổi giá trị của các tham số hình thức nhưng không ảnh hưởng tới giá trị của các tham số thực tương ứng.

Truyền tham chiếu (pass by reference): nếu muốn hàm được gọi thay đổi giá trị của biến nơi gọi hàm, ta phải sử dụng cách truyền tham chiếu.

## IV. Bài tập:

**Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.**

**Bài 1: (Liệt kê số nguyên tố)** Viết hàm kiểm tra một số nguyên dương có phải là số nguyên tố hay không. Liệt kê tất cả các cặp số sinh đôi nhỏ hơn 1.000.000.

**Bài 2: (Tìm ngày sớm nhất)** Viết hàm kiểm tra ngày tháng năm có hợp lệ hay không và hàm tìm số nguyên nhỏ nhất của 2 số cho trước. Áp dụng: Chương trình cho phép người dùng nhập vào ngày tháng năm. Chương trình kết thúc nhập khi người dùng nhập ngày tháng năm không hợp lệ. Hãy cho biết ngày tháng năm sớm nhất trong những ngày tháng năm hợp lệ đã nhập.

**Bài 3: (Đếm số lượng số hoàn hảo)** Viết hàm kiểm tra số hoàn hảo. Áp dụng: hãy cho biết trong đoạn từ 1 đến 1.000.000 có bao nhiêu số hoàn hảo?

**Bài 4: (*Tam giác*)** Viết hàm kiểm tra một bộ ba số nguyên dương có tạo thành một tam giác hay không? Áp dụng: hãy cho biết trong đoạn từ 1 đến 1.000.000 có bao nhiêu bộ ba tạo thành một tam giác.

**Bài 5: (*Tìm số tự nhiên*)** Viết chương trình liệt kê tất cả các số nguyên dương gồm 4 chữ số mà trong mỗi số không có hai chữ số nào giống nhau.

## BÀI THỰC HÀNH TUẦN 10 – BUỔI 02

### MẢNG MỘT CHIỀU

#### ❖ Mục đích:

1. Giới thiệu mảng một chiều
2. Thực hành các thao tác cơ bản trên mảng một chiều

#### I. Mảng một chiều

Trước giờ chúng ta đã thảo luận về một biến như là một vị trí đơn trong bộ nhớ máy tính. Có thể có một tập các vị trí vùng nhớ, tất cả có cùng kiểu dữ liệu, được nhóm lại cùng nhau dưới cùng một tên. Như là một tập hợp được gọi là một mảng. Giống như mọi biến khác, một mảng phải được định nghĩa để máy tính có thể cung cấp số lượng vùng nhớ tương ứng. Số lượng này dựa trên kiểu dữ liệu lưu trữ và số vùng nhớ.

Ví dụ 10.1:

---

```
/*Thông tin CT & TG
Ho ten:
MSSV:
Lop:
Ngày:
*/
//Khai bao thu vien
#include<iostream>
#include<cmath>
using namespace std;

//Khai bao kích thước mảng
const int SIZE = 100;

//Ham chính
void main()
{
    int arA[SIZE]; //Khai bao mảng với 100 phần tử
    ...
}
```

---

Mỗi phần tử của mảng, được truy xuất bởi tên mảng và vị trí trong mảng (subscript). Trong C++ vị trí trong mảng, đôi khi được đề cập như là chỉ mục (index), được đặt trong cặp ngoặc vuông. Số của vị trí trong mảng luôn bắt đầu từ 0 đến tổng số vùng nhớ trừ 1.

0	1	2	3	4	5	...	97	98	99

## II. Khởi tạo mảng

Chúng ta có thể khởi tạo giá trị của mảng theo nhiều cách:

Khởi tạo mảng với tất cả các giá trị bằng 0:

---

```
int arA[SIZE] = {0};
```

---

Khởi tạo mảng với các giá trị được liệt kê, các phần tử còn lại bằng 0:

---

```
int arA[SIZE] = {1, 2, 4, 6, 9};
```

---

Mảng tự xác định kích thước:

---

```
int arA[] = {1, 2, 4, 6, 9}; //Kích thước mảng bằng 5
```

---

Khởi tạo mảng dùng vòng lặp:

---

```
int arA[SIZE];
for(int i= 0; i< SIZE; i++)
    arA[i] = 0;
```

---

Khởi tạo mảng với giá trị do người dùng nhập:

---

```
int arA[SIZE];
for(int i= 0; i< SIZE; i++)
    cin>>arA[i];
```

---

## III. Truyền tham số mảng cho hàm

Mảng có thể được truyền như là các đối số (tham số) tới các hàm. Mặc dù các biến có thể được truyền tham trị hoặc tham chiếu, các mảng luôn được **truyền con trỏ (pass by pointer)**, nó tương tự truyền tham chiếu. Điều này có nghĩa là các mảng

có thể thay đổi giá trị bởi lời gọi hàm. Tuy nhiên, không bao giờ có dấu & đứng giữa kiểu dữ liệu và tên, như là truyền tham chiếu các tham số.

### Ví dụ 10.2:

---

```

/*Thông tin CT & TG
Chương trình minh họa truyền mảng cho hàm
Ho ten:
MSSV:
Lop:
Ngày:
*/
//Khai bao thu vien
#include<iostream>
using namespace std;

//Khai bao kích thước mảng
const int MAXSIZE = 100;
//Khai bao nguyên mẫu hàm
void NhapDuLieu(int ar[], int &Size);
float TinhTrungBinh(const int ar[], int Size);

//Hàm chính
void main()
{
    int arA[MAXSIZE]; //Khai bao mảng với 100 phần tử
    int Size = 0; //Số lượng thực dụng
    float TrungBinh;

    NhapDuLieu(arA, Size);
    TrungBinh = TinhTrungBinh(arA, Size);

    cout<<"Trung bình của "<<Size<<" phần tử là: " <<TrungBinh
                                     <<endl;
}
/*****
- Tên hàm: NhapDuLieu
- Công việc: Hàm này nhập và lưu giá trị dữ liệu vào mảng
- Dữ liệu vào: Không
- Dữ liệu ra: mảng chứa các giá trị và số lượng phần tử
*****/
void NhapDuLieu(int ar[], int &Size)
{
    int pos = 0;

```

---

---

```

int Value;

cout<<"Hay nhap gia tri hoac -99 de ket thuc: "<<endl;
cin>>Value;
while(Value != -99)
{
    ar[pos] = Value; //luu du lieu vao mang
    pos++;           //tang chi muc cua mang

    cout<<"Hay nhap gia tri hoac -99 de ket thuc: "<<endl;
    cin>>Value;
}
Size = pos; //den khi vong lap ket thuc
            // pos se giu so luong phan tu da nhap
}
/*****
- Ten ham: TinhTrungBinh
- Cong viec: Ham nay tinh va tra ve trung binh cac gia tri
- Du lieu vaonhT: Mang cac so nguyen va kích thước
- Du lieu ra: trung binh cac gia tri trong mang
*****/
float TinhTrungBinh(const int a[], int Size)
{
    int sum = 0;
    for(int pos = 0; pos < Size; pos++)
        sum += a[pos]; //Cong cac gia tri vao tong
    return float(sum)/Size;
}

```

---

#### IV. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.

##### ❖ Thao tác nhập xuất

**Bài 1:** (*Nhập xuất mảng*) Nhập xuất mảng một chiều các số nguyên/ thực.

**Bài 2:** (*Tạo số ngẫu nhiên*) Phát sinh ngẫu nhiên mảng một chiều các số nguyên.

**Bài 3:** (*Xuất các số lẻ*) Nhập mảng các số nguyên và xuất lên màn hình các số lẻ có trong mảng.



**Bài 4:** (*Xuất các phần tử âm*) Nhập mảng một chiều các số thực và xuất các phần tử âm có trong mảng.

**Bài 5:** (*Xuất số nguyên tố*) Nhập mảng một chiều các số nguyên và xuất ra màn hình các phần tử là số nguyên tố có trong mảng.

❖ **Thao tác tìm kiếm**

**Bài 6:** (*Tìm vị trí cuối cùng*) Cho biết vị trí của phần tử bằng x xuất hiện cuối cùng trong mảng. Với x nhập vào từ bàn phím.

**Bài 7:** (*Liệt kê các vị trí có giá trị nhỏ nhất*) Cho biết các vị trí của phần tử nhỏ nhất trong mảng các số nguyên.

**Bài 8:** (*Liệt kê các vị trí của số nguyên tố*) Cho biết các vị trí của phần tử là số nguyên tố trong mảng các số nguyên.

**Bài 9:** (*Vị trí phần tử âm đầu tiên*) Cho biết vị trí phần tử âm đầu tiên trong mảng. Nếu không có phần tử âm trong mảng trả về -1.

**Bài 10:** (*Vị trí phần tử âm lớn nhất*) Cho biết vị trí phần tử âm lớn nhất trong mảng.

❖ **Thao tác tính toán**

**Bài 11:** (*Đếm số lượng âm, dương*) Đếm các phần tử âm, dương trong mảng.

**Bài 12:** (*Đếm số lượng chẵn, lẻ*) Đếm các phần tử chẵn, lẻ trong mảng.

**Bài 13:** (*Đếm số lượng bằng x*) Đếm số lần xuất hiện của phần tử x trong mảng. Với x nhập vào từ bàn phím.

**Bài 14:** (*Đếm số lượng số nguyên tố*) Đếm các phần tử là số nguyên tố có trong mảng.

**Bài 15:** (*Đếm số lượng số hoàn hảo*) Đếm các phần tử là số hoàn hảo có trong mảng.

**Bài 16:** (*Tổng các phần tử chẵn*) Tính tổng các phần tử chẵn trong mảng.

**Bài 17:** (*Tổng các số nguyên tố*) Tính tổng các phần tử là số nguyên tố trong mảng.

**Bài 18:** (*Tổng các vị trí chẵn*) Tính tổng các phần tử nằm ở vị trí chẵn trong mảng.

**Bài 19:** (*Tổng các phần tử cực đại*) Tính tổng các phần tử cực đại trong mảng các số nguyên (phần tử cực đại là phần tử lớn hơn các phần tử xung quanh nó).

***Ví dụ:*** 1 5 2 6 3 5 1 8 6

**Bài 20:** (*Trung bình các số hoàn hảo*) Tính giá trị trung bình của các số hoàn hảo trong mảng.

# BÀI THỰC HÀNH TUẦN 11 – BUỔI 01

## CHUỖI – MẢNG KÝ TỰ

### ❖ Mục đích:

1. Hiểu cơ bản mảng các ký tự
2. Làm quen các thao tác xử lý cơ bản trên mảng ký tự
3. Hướng dẫn cách thức nhập và xuất chuỗi
4. Làm quen với các hàm xử lý chuỗi của C++

### I. Hằng chuỗi

Một hằng chuỗi là một chuỗi được đặt trong dấu ngoặc kép. Ví dụ, “chao ban”, “hoc C++” là các hằng chuỗi. Khi chúng được lưu vào bộ nhớ máy tính, ký tự rỗng được tự động thêm vào cuối. Chuỗi “hay nhap mot ky tu” được lưu trữ như sau:

H	a	y		n	h	a	p		m	o	t		k	y		t	u	\0
---	---	---	--	---	---	---	---	--	---	---	---	--	---	---	--	---	---	----

Khi một hằng chuỗi được sử dụng trong C++, địa chỉ vùng nhớ thực của nó được truy xuất. Trong câu lệnh:

---

```
cout<<"hay nhap mot ky tu";
```

---

Địa chỉ vùng nhớ được truyền tới đối tượng cout. cout sau đó hiển thị các ký tự liên tiếp cho đến khi gặp ký tự rỗng.

### II. Lưu trữ chuỗi trong mảng

Thông thường chúng ta cần truy xuất đến những phần của chuỗi hơn là toàn bộ chuỗi. Ví dụ, chúng ta có thể muốn thay đổi các ký tự trong một chuỗi hoặc thậm chí so sánh hai chuỗi. Trong trường hợp này, một hằng chuỗi không phải là cái chúng ta cần. Thay vào đó, một mảng ký tự là sự lựa chọn thích hợp. Khi sử dụng mảng ký tự, không gian vùng nhớ phải đủ để lưu trữ chuỗi và cả ký tự rỗng. Ví dụ:

---

```
char cChuoi[10];
```

---

Câu lệnh này định nghĩa một mảng gồm 10 ký tự có tên là `cChuoi`. Tuy nhiên, mảng này chỉ có thể lưu trữ không nhiều hơn 9 ký tự khác rỗng vì một ô nhớ được dành riêng cho ký tự rỗng. Cùng xem xét đoạn lệnh sau:

---

```
char cChuoi[10];
cout<<"hay nhap mot chuoi khong qua 9 ky tu:";
cin>>cChuoi;
```

---

Hãy nhớ rằng máy tính thực sự xem `cChuoi` như là địa chỉ bắt đầu của mảng. Có một vấn đề có thể nảy sinh khi sử dụng `cin` trên một mảng ký tự. `cin` không biết `cChuoi` chỉ có 10 phần tử. Nếu người dùng nhập một chuỗi dài hơn 9 ký tự, thì `cin` sẽ ghi qua phần cuối của mảng. Chúng ta có thể giải quyết vấn đề này bằng cách sử dụng hàm `getline`. Nếu bạn sử dụng:

---

```
cin.getline(cChuoi, 10);
```

---

Thì máy tính biết rằng chiều dài tối đa của chuỗi, bao gồm cả ký tự rỗng, là 10. Do đó, `cin` sẽ đọc cho đến khi người dùng gõ ENTER hoặc cho đến khi đọc được 9 ký tự, tùy theo cái nào đến trước. Khi chuỗi là một mảng, nó có thể được xử lý theo từng ký tự.

### III. Các hàm thư viện xử lý chuỗi

C++ cung cấp nhiều hàm cho việc kiểm tra và thao tác với chuỗi. Dưới đây là một số hàm thông dụng:

Hàm	Ý nghĩa	Ví dụ
<code>strlen(char s[])</code>	Nhận về chiều dài chuỗi <code>s</code>	<pre>char cLine[40] = "Have a nice day!"; int iLength; iLength = strlen(cLine);</pre>
<code>strcat(char s1[], char s2[])</code>	Nối chuỗi <code>s2</code> vào cuối chuỗi <code>s1</code>	<pre>char str1[25] = "Hello "; char str2[11] = "World"; strcat(str1, str2);</pre>
<code>cin.get(char ch)</code>	Đọc một ký tự và lưu vào biến <code>ch</code>	<pre>char firstchar, ch, secondchar; cin.get(firstChar); cin.get(ch);</pre>

		<code>cin.get(secondChar);</code>
<code>cin.get(char s[], int n)</code>	Đọc n ký tự và lưu vào chuỗi s	<code>char strName[21];</code> <code>cin.get(strName, 21)</code>
<code>cin.ignore(int n, char ch)</code>	Bỏ qua n ký tự hoặc gặp ký tự ch	<code>cin.ignore(200, '\n');</code>
<code>cin.getline(char s[], int n)</code>	Đọc từ 1 tới n ký tự và lưu vào s	<code>char cLine[50];</code> <code>cin.getline(cLine, 50);</code>
<code>isalpha(char ch)</code>	Kiểm tra giá trị trong ch có phải là chữ cái hay không	
<code>isdigit(char ch)</code>	Kiểm tra giá trị trong ch có phải chữ số hay không	
<code>isspace(char ch)</code>	Kiểm tra giá trị trong ch có phải khoảng trắng hay không	
<code>strlwr(char s[]);</code>	Đổi tất cả ký tự của s sang chữ thường	
<code>strupr(char s[])</code>	Đổi tất cả ký tự của s sang chữ hoa	
<code>strcmp(char s[], char t[])</code>	So sánh 2 chuỗi s và t, giá trị âm nếu s < t, 0 nếu s bằng t, giá trị dương nếu s > t	
<code>strcpy(char s[], char t[])</code>	Gán nội dung của chuỗi t cho chuỗi s	

Ví dụ 11.1:


---

```

/*Thông tin CT & TG
Chương trình minh họa xử lý chuỗi là mảng các ký tự
Họ tên:
MSSV:
Lớp:
Ngày:
*/
//Khai báo thư viện
#include<iostream>
#include<string>
using namespace std;

//Khai báo kích thước mảng
const int MAXSIZE = 100;
//Khai báo nguyên mẫu hàm
int DemSoKyTuSo(char c[]);

//Hàm chính
void main()
{
    char cChuoi[MAXSIZE]; //Khai báo mảng ký tự với 100 phần tử
    int iLength = 0; //Chiều dài chuỗi
    int iNumDigits = 0;

    cout<<"Hãy nhập một chuỗi không quá 99 ký tự: "<<endl;
    cin.getline(cChuoi, 99);

    iNumDigits = DemSoKyTuSo(cChuoi);

    cout<<"Số ký tự số trong chuỗi: "<<iNumDigits<<endl;

}
/*****
- Tên hàm: DemSoKyTuSo
- Công việc: Hàm này đếm số lượng chữ số trong chuỗi
- Dữ liệu vào: Mảng ký tự
- Dữ liệu ra: Số lượng chữ số trong mảng ký tự
*****/
int DemSoKyTuSo(char c[])
{
    int KQ = 0;
    int pos = 0;

```

---

---

```
while(pos < strlen(c))
{
    if(isdigit( c[pos]))
        KQ++;
    pos++;
}
return KQ;
}
```

---

#### IV. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.

**Bài 1: (Chuỗi đối xứng)** Một chuỗi đối xứng là một chuỗi các ký tự mà đọc xuôi cũng giống như đọc ngược. Ví dụ, hai chuỗi sau là chuỗi đối xứng:

*1457887541 madam*

Viết một chương trình cho phép người dùng nhập vào một chuỗi các ký tự có kích thước không quá 50. Chương trình sẽ xác định chuỗi vừa nhập có phải là chuỗi đối xứng hay không?

**Bài 2: (So sánh hai chuỗi)** Hàm strcmp(char s1[], char s2[]) so sánh chuỗi s1 với chuỗi s2. Giá trị trả về của hàm là một số âm nếu s1 < s2, bằng 0 nếu s1==s2, và một số nguyên dương nếu s1>s2. Viết chương trình nhập vào hai tên (bao gồm cả họ và tên: tên nhập trước, họ nhập sau và cách nhau bằng dấu phẩy) và sau đó xuất lên màn hình theo thứ tự giảm dần. Hai tên được lưu trữ trong mảng các ký tự có kích thước không lớn hơn 25. Sử dụng hàm strcmp để so sánh hai tên. Lưu ý: 'a' < 'b' và 'b' < 'c', ...

**Bài 3: (Đếm số phụ âm)** Viết một chương trình xác định có bao nhiêu phụ âm trong một chuỗi được nhập bởi người dùng có kích thước không lớn hơn 50. Xuất chuỗi đã nhập và số lượng phụ âm có trong chuỗi.

**Bài 4: (Đếm số ký tự)** Đếm số lượng các khoảng trắng và số lượng các chữ cái có trong chuỗi.

**Bài 5: (Xóa khoảng trắng thừa)** Một khoảng trắng xuất hiện trong chuỗi được xem là thừa nếu nó xuất hiện ở đầu, cuối cũng như nhiều hơn một khoảng trắng liên tiếp ở giữa chuỗi. Hãy viết chương trình cho phép người dùng nhập vào một chuỗi không quá 50 ký tự. Sau đó, chương trình sẽ xóa tất cả các khoảng trắng thừa có trong chuỗi.

**Bài 6: (Đổi sang chữ in hoa)** Giả sử từ trong một chuỗi là một dãy các ký tự liên tiếp nhau và ngăn cách nhau bởi các khoảng trắng. Hãy viết chương trình đổi tất cả các ký tự đầu tiên của mỗi từ thành chữ in hoa.

**Bài 7: (Tổng kê số lần xuất hiện các ký tự)** Hãy viết chương trình cho phép người dùng nhập vào một chuỗi không quá 50 ký tự. Sau đó, chương trình sẽ tổng kê số lần xuất hiện của từng ký tự trong chuỗi vừa nhập.

**Bài 8: (Nối chuỗi)** Hãy viết chương trình cho phép người dùng nhập vào hai chuỗi s1, s2 không quá 50 ký tự. Sau đó, chương trình sẽ nối chuỗi s2 vào cuối chuỗi s1.

**Bài 9: (Chèn ký tự)** Hãy viết chương trình cho phép người dùng nhập vào một chuỗi không quá 50 ký tự và một ký tự muốn chèn vào chuỗi vừa nhập tại vị trí ViTri, với ViTri cũng được nhập từ bàn phím. Sau đó, chương trình sẽ thực hiện việc chèn ký tự muốn chèn vào chuỗi tại vị trí ViTri.

**Bài 10: (Mã hóa Caesar)** Thế kỷ thứ 3 trước công nguyên, nhà quân sự người La Mã Julius Caesar đã nghĩ ra phương pháp mã hóa một bản tin như sau: thay thế mỗi chữ trong bản tin bằng chữ đứng sau nó k vị trí trong bảng chữ cái. Giả sử chọn k = 3, ta có bảng chuyển đổi như sau:

Chữ ban đầu: **a b c d e f g h i j k l m n o p q r s t u v w x y z**

Chữ thay thế: **D E F G H I J K L M N O P Q R S T U V W X Y Z A B C**



(sau Z sẽ vòng lại là A, do đó  $x \rightarrow A$ ,  $y \rightarrow B$  và  $z \rightarrow C$ )

Giả sử có bản tin gốc (bản rõ): **meet me after the toga party**

Như vậy bản tin mã hóa (bản mã) sẽ là: PHHW PH DIWHU WKH WRJD SDUWB

Thay vì gửi trực tiếp bản rõ cho các cấp dưới, Ceasar gửi bản mã. Khi cấp dưới nhận được bản mã, tiến hành giải mã theo quy trình ngược lại để có được bản rõ. Như vậy nếu đối thủ của Ceasar có lấy được bản mã, thì cũng không hiểu được ý nghĩa của bản mã.

Phương pháp Ceasar được biểu diễn như sau: với mỗi chữ cái p thay bằng chữ mã hóa C, trong đó:

$$C = (p + k)$$

Và quá trình giải mã đơn giản là:  $p = (C - k)$

k được gọi là khóa. Dĩ nhiên là Ceasar và cấp dưới phải cùng dùng chung một giá trị khóa k, nếu không bản tin giải mã sẽ không giống bản rõ ban đầu.

**Yêu cầu:** Bạn hãy viết một chương trình mô phỏng quá trình mã hóa và giải mã theo phương pháp Ceasar như sau:

- Giai đoạn mã hóa: Chương trình cho phép người dùng nhập vào một thông điệp có chiều dài không quá 50 và một khóa bí mật k là một số nguyên. Sau đó chương trình sẽ mã hóa thông điệp trên theo khóa k và gửi nội dung đã mã hóa cho giai đoạn giải mã.
- Giai đoạn giải mã: Người nhận sẽ nhận về một nội dung thông điệp đã được mã hóa. Sau đó người nhận sẽ giải mã thông điệp bằng theo khóa k đã biết.

Hãy xuất lên màn hình các nội dung: thông điệp ban đầu (bản rõ), thông điệp sau khi mã hóa (bản mã) và thông điệp sau khi giải mã.

## BÀI THỰC HÀNH TUẦN 11 – BUỔI 02

### CHUỖI – LỚP STRING TRONG C++

#### ❖ Mục đích:

1. Giới thiệu lớp string trong c++
2. Làm quen các thao tác xử lý cơ bản trên lớp string

#### I. Lớp string

C++ có một thư viện string cung cấp nhiều cách thức thuận lợi để xử lý văn bản. Để sử dụng string chúng ta cần khai báo thư viện: **#include<string>**

Chúng ta có thể sử dụng **một số toán tử** trên các biến có kiểu dữ liệu string:

+ Các toán tử đại số:

Toán tử	Ý nghĩa	Ví dụ
=	Gán giá trị cho biến string	<code>string str1 = "I love";</code> <code>string str2 = "You";</code>
+	Cộng giá trị 2 biến string	<code>string str3;</code> <code>str3 = str1 + str2;</code>

+ Các toán tử quan hệ:

Các ví dụ dưới đây sử dụng các giá trị trong các biến có kiểu string như sau:

```
string str1 = "Apple"; string str2 = "apple";
string str3 = "apples"; string str4 = "orange";
```

Toán tử	Ý nghĩa	Ví dụ
==	So sánh bằng	<code>bool t1 = (str1 == str2);</code> <code>// t1 = false</code>
<	So sánh nhỏ hơn	<code>bool t2 = (str1 &lt; str2);</code> <code>//t2 = true</code>
<=	So sánh nhỏ hơn hay bằng	<code>bool t3 = (str2 &lt;= str3);</code> <code>//t3 = true</code>

!=	So sánh khác	<code>bool t4 = (str3 != str4); //t4 = true</code>
>	So sánh lớn hơn	<code>bool t5 = (str4 &gt; str3); // t5 = true</code>
>=	So sánh lớn hơn hay bằng	<code>bool t6 = (str4 &gt;= str3); // t6 = true</code>

Trong thực tế, một string là một tổ hợp của một chuỗi các ký tự. Các ký tự trong một string có thể được truy xuất thông qua vị trí của chúng. Nghĩa là, một string với chiều dài n, những vị trí của các ký tự của nó có phạm vi từ 0 tới n - 1. Với một biến string, chúng ta có thể truy xuất đến một ký tự của nó thông qua toán tử [<vị trí>].

---

```
string myMsg = "How are you doing?";
cout<<myMsg[4]<<" "<<myMsg[12]
<<myMsg[13]<<myMsg[16]<<endl; // a dog
```

---

## II. Các hàm thành viên trong lớp string

Thư viện string rất linh hoạt cho việc cung cấp nhiều phép toán liên quan tới thao tác xử lý chuỗi/ văn bản. Ví dụ: tìm chiều dài chuỗi, so sánh hai chuỗi, tìm kiếm/ rút trích chuỗi con, nối chuỗi, ... Mỗi biến string được kết hợp với một số hàm thành viên, như s.length(), s.substr(...). Dưới đây là một số hàm thành viên thông dụng của string.

Hàm	Mô tả
s.length()	Trả về số lượng ký tự trong string s
s.substr(x, y)	Rút trích một chuỗi con với chiều dài y bắt đầu tại vị trí x. Nếu không có y, một chuỗi con từ vị trí x tới cuối chuỗi sẽ được rút trích
s.find(r)	Kiểm tra chuỗi (string) r có xuất hiện trong chuỗi s hay không. Nếu có hàm trả về vị trí bắt đầu xuất hiện
s.erase(x, n)	Xóa n ký tự bắt đầu tại vị trí x
s.replace(x, n, str)	Thay thế n ký tự tại vị trí bắt đầu là x bằng str. <b>Lưu ý: chiều dài của str có thể lớn hơn n.</b>

s1.compare(s2)	So sánh chuỗi s1 với s2. Giá trị trả về là âm nếu s1 < s2, bằng 0 nếu s1 giống s2, là dương nếu s1 > s2
s1.swap(s2)	Hoán đổi nội dung hai chuỗi
s1.insert(index, s2)	Thêm chuỗi s2 vào s1 sau vị trí index
s.begin()	Trả về vị trí bắt đầu chuỗi
s.end()	Trả về vị trí kết thúc chuỗi

Ví dụ 11.2:

```

/*Thông tin CT & TG
Chương trình minh họa xử lý chuỗi có kiểu string
Họ tên:
MSSV:
Lớp:
Ngày:
*/
//Khai báo thư viện
#include<iostream>
#include<string>
using namespace std;

//Khai báo nguyên mẫu hàm
int DemSoKyTuSo(string str);

//Hàm chính
void main()
{
    string str;
    int iNumDigits = 0;
    cout<<"Hay nhập một chuỗi: "<<endl;
    getline(cin, str);

    iNumDigits = DemSoKyTuSo(str);

    cout<<"Số ký tự số trong chuỗi: "<<iNumDigits<<endl;
}
/*****
- Tên hàm: DemSoKyTuSo
- Công việc: Hàm này đếm số lượng chữ số trong chuỗi
- Dữ liệu vào: Mảng ký tự
- Dữ liệu ra: Số lượng chỉ số trong mảng ký tự
*****/

```

---

```
int DemSoKyTuSo(string s)
{
    int KQ = 0;
    int pos = 0;

    while(pos < s.length())
    {
        if(isdigit(s[pos]))
            KQ++;
        pos++;
    }
    return KQ;
}
```

---

### III. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính. Mỗi bài phải sử dụng các biến lưu trữ chuỗi có kiểu string trong thư viện `#include<string>` của C++.

**Bài 1: (Chuỗi đối xứng)** Một chuỗi đối xứng là một chuỗi các ký tự mà đọc xuôi cũng giống như đọc ngược. Ví dụ, hai chuỗi sau là chuỗi đối xứng:

*1457887541 madam*

Viết một chương trình cho phép người dùng nhập vào một chuỗi các ký tự. Chương trình sẽ xác định chuỗi vừa nhập có phải là chuỗi đối xứng hay không?

**Bài 2: (So sánh hai chuỗi)** Viết chương trình nhập vào hai tên (bao gồm cả họ và tên: tên nhập trước, họ nhập sau và cách nhau bằng dấu phẩy) và sau đó xuất lên màn hình theo thứ tự tăng dần. Sử dụng hàm `s1.compare(s2)` để so sánh hai tên. Lưu ý: 'a' < 'b' và 'b' < 'c', ...

**Ví dụ:**

Huy, tran

Cuong, nguyen

Kết quả:

Cuong, nguyen

Huy, tran

Vì theo kết quả so sánh hai chuỗi trên ta có chữ 'H' > 'C' nên hàm trả về một số âm.

**Bài 3: (Đếm số phụ âm)** Viết một chương trình xác định có bao nhiêu phụ âm trong một chuỗi được nhập bởi người dùng. Xuất chuỗi đã nhập và số lượng phụ âm có trong chuỗi.

**Bài 4: (Tìm chuỗi con)** Viết một chương trình nhập vào một dòng văn bản không dấu và tìm một chuỗi con được nhập từ bàn phím. Sử dụng phương thức find để xác định vị trí xuất hiện đầu tiên của chuỗi cần tìm trong dòng văn bản, và gán vị trí này vào một biến ViTri. Nếu tìm thấy chuỗi, xuất phần còn lại của dòng văn bản lên màn hình bắt đầu với chuỗi tìm kiếm. Sau đó, sử dụng lại phương thức find tìm vị trí xuất hiện tiếp theo của chuỗi tìm kiếm trong dòng văn bản còn lại. Nếu có xuất hiện lần hai, xuất phần còn lại của dòng văn bản bắt đầu với lần xuất hiện thứ hai.

**Bài 5: (Đếm số lần xuất hiện của chuỗi con)** Viết một chương trình dựa trên chương trình bài 4 cho phép nhập vào một vài dòng văn bản và một chuỗi cần tìm và sử dụng phương thức find để xác định tổng số lần xuất hiện của chuỗi cần tìm. Xuất lên màn hình kết quả.

# BÀI THỰC HÀNH TUẦN 12 – BUỔI 01

## MẢNG HAI CHIỀU

### ❖ Mục đích:

1. Giới thiệu mảng hai chiều
2. Làm quen các thao tác xử lý cơ bản trên mảng hai chiều

### I. Mảng hai chiều

Mảng hai chiều lưu trữ dữ liệu như là tập hợp các dòng và cột. Mỗi phần tử của mảng hai chiều được xác định qua một cặp chỉ số dòng và chỉ số cột. Tất cả các phần tử có cùng một kiểu dữ liệu. Mỗi phần tử được truy xuất thông qua chỉ số dòng và chỉ số cột.

Khai báo mảng và khởi tạo mảng hai chiều:

Ví dụ:

---

```
int Mang2Chieu[3][6]; // khai bao mang 2 chieu
int temp[4][3] = {50, 70, 60, 48, 75, 62, 51,
                  69, 60, 52, 78, 63};
int temp[4][3] = {{50, 70, 60}, {48, 75, 62},
                  {51, 69, 60}, {52, 78, 63}};
int t2[7][4] = {{50, 70, 60}, {48, 75, 62},
                {51, 69, 60}, {52, 78, 63}};
int temp[][3] = {{50, 70, 60}, {48, 75, 62},
                 {51, 69, 60}, {52, 78, 63}};
int temp[][3] = {50, 70, 60, 48, 75, 62, 51,
                 69, 60, 52, 78, 63};
```

---

### II. Xử lý mảng hai chiều

Để xử lý các giá trị dữ liệu được lưu trong mảng ta có thể sử dụng cú pháp sau:

---

```
int Mang2Chieu[3][6]; // khai bao mang 2 chieu
for(int Dong = 0; Dong < SoDong; Dong++) //Voi moi dong
    for(int Cot = 0; Cot < SoCot; Cot++) //Voi moi cot
        if(Mang2Chieu[Dong][Cot] thỏa điều kiện)
            Xử lý Mang2Chieu[Dong][Cot];
```

---

Ví dụ 12.1:

---

```

/*Thông tin CT & TG
Chương trình minh họa xử lý mảng 2 chiều
Họ tên:
MSSV:
Lop:
Ngày:
*/
//Khai báo thư viện
#include<iostream>
using namespace std;
//Định nghĩa số dòng, số cột tối đa
#define MAX_ROWS 50
#define MAX_COLS 100

//Khai báo nguyên mẫu hàm
//Nhập mảng
void NhapMang2Chieu(int a[][MAX_COLS], int &m, int &n);
//Tính toán
int TinhTongCacPhanTu(int a[][MAX_COLS], int m, int n);
//Tìm kiếm
int TimPhanTuNhoNhat(int a[][MAX_COLS], int m, int n);
//Hàm chính
void main()
{
    //Khai báo biến
    int arMang2Chieu[MAX_ROWS][MAX_COLS];
    int iNumRow, iNumCol;

    //Nhập liệu
    NhapMang2Chieu(arMang2Chieu, iNumRow, iNumCol);

    //Tính toán
    int Sum = TinhTongCacPhanTu(arMang2Chieu, iNumRow, iNumCol);
    cout<<"Tong cac phan tu cua mang: "<<Sum<<endl;
    //Tìm kiếm
    int Min = TimPhanTuNhoNhat(arMang2Chieu, iNumRow, iNumCol);
    cout<<"Gia tri nho nhat trong mang: "<<Min<<endl;
}
/*****
- Tên hàm: NhapMang2Chieu
- Công việc: Hàm này nhập vào số dòng, số cột và giá trị
               các phần tử trong mảng
- Dữ liệu vào: Không
- Dữ liệu ra: mảng các giá trị và số dòng, số cột
*****/

```

---



---

```
//Nhập mảng
void NhapMang2Chieu(int a[][MAX_COLS], int &m, int &n)
{
    cout<<"Hay nhap so dong: "<<endl;
    cin>>m;
    cout<<"Hay nhap so cot: "<<endl;
    cin>>n;
    for(int row_pos = 0; row_pos < m; row_pos++)
        for(int col_pos = 0; col_pos < n; col_pos++)
        {
            cout<<"a["<<row_pos<<"]["<<col_pos<<"]="";
            cin>>a[row_pos][col_pos];
        }
}
/*****
- Ten ham: TinhTongCacPhanTu
- Cong viec: Ham nay nhan ve mot mang 2 chieu cac phan tu
              va tinh tong cac phan tu co trong mang
- Du lieu vao:mang 2 chieu cac so nguyen, so dong va so cot
- Du lieu ra: Tong cac phan tu trong mang
*****/
int TinhTongCacPhanTu(int a[][MAX_COLS], int m, int n)
{
    int Tong = 0;
    for(int row_pos = 0; row_pos < m; row_pos++)
        for(int col_pos = 0; col_pos < m; col_pos++)
            Tong = Tong + a[row_pos][col_pos];
    return Tong;
}
/*****
- Ten ham: TinhTongCacPhanTu
- Cong viec: Ham nay nhan ve mot mang 2 chieu cac phan tu
              va tim gia tri nho nhat co trong mang
- Du lieu vao:mang 2 chieu cac so nguyen, so dong va so cot
- Du lieu ra: Gia tri nho nhat trong mang
*****/
int TimPhanTuNhoNhat(int a[][MAX_COLS], int m, int n)
{
    int minValue = a[0][0];
    for(int row_pos = 0; row_pos < m; row_pos++)
        for(int col_pos = 0; col_pos < m; col_pos++)
            if(a[row_pos][col_pos] < minValue)
                minValue = a[row_pos][col_pos];
    return minValue;
}

```

---

### III. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.

#### ❖ Thao tác nhập/xuất

**Bài 1:** (*Nhập/xuất mảng hai chiều các số nguyên dương*) Viết chương trình nhập vào mảng 2 chiều m dòng, n cột các số nguyên dương (Nhập sai báo lỗi và kết thúc quá trình nhập).

**Bài 2:** (*Nhập/xuất mảng hai chiều các số thực*) Viết hàm nhập/ xuất mảng 2 chiều các số thực.

**Bài 3:** (*Xuất phần tử tận cùng là 5*) Viết hàm in ra những phần tử có ký tự tận cùng là 5.

**Bài 4:** (*Xuất các giá trị thuộc 2 đường chéo*) Viết hàm in ra các phần tử nằm trên 2 đường chéo của ma trận vuông NxN.

**Bài 5:** (*Xuất các phần tử phía trên đường chéo phụ*) Viết hàm in ra các phần tử nằm phía trên đường chéo phụ của ma trận vuông các số nguyên.

**Bài 6:** (*Xuất các phần tử phía trên đường chéo chính*) Viết hàm in ra các phần tử nằm phía trên đường chéo chính của ma trận vuông các số nguyên.

**Bài 7:** (*Tạo mảng các số nguyên*) Viết hàm tạo ma trận a các số nguyên gồm m dòng, n cột. Trong đó phần tử tại thứ a[i][j] = i\*j. Với m, n nhập từ bàn phím.

**Bài 8:** (*Tam giác Pascal*) Viết chương trình in tam giác Pascal với chiều cao h được nhập từ bàn phím.

Ví dụ:  $h = 5$

```

1
1 1
1 2 1
1 3 3 1
1 4 6 4 1

```

#### ❖ Thao tác tính toán

**Bài 9:** (*Tính tổng các phần tử trên cùng dòng*) Viết hàm tính tổng các phần tử trên cùng một dòng.

**Bài 10:** (*Tính tổng các phần tử trên cùng cột*) Viết hàm tính tổng các phần tử trên cùng một cột.

**Bài 11:** (*Tính tổng các phần tử chẵn*) Viết hàm tính tổng các phần tử chẵn có trong ma trận.

**Bài 12:** (*Tính tổng các phần tử thuộc đường chéo chính*) Viết hàm tính tổng các phần tử nằm trên đường chéo chính của ma trận vuông.

**Bài 13:** (*Tính tổng các số nguyên tố*) Viết hàm tính tổng các phần tử là số nguyên tố có trong ma trận các số nguyên dương.

**Bài 14:** (*Tính tổng các số hoàn thiện*) Viết hàm tính tổng các phần tử là số hoàn thiện có trong ma trận các số nguyên.

**Bài 15:** (*Tính tổng các giá trị lớn nhất mỗi dòng*) Viết hàm tính tổng các giá trị lớn nhất trên mỗi dòng.

**Bài 16:** (*Tính trung bình các phần tử nhỏ nhất mỗi cột*) Viết hàm tính giá trị trung bình của các phần tử nhỏ nhất trên mỗi cột.

#### ❖ Thao tác tìm kiếm

**Bài 17:** (*Tìm vị trí phần tử lớn nhất*) Viết hàm tìm vị trí phần tử lớn nhất trong ma trận các số nguyên.

**Bài 18:** (*Tìm vị trí phần tử chẵn ở cuối mảng*) Viết hàm tìm vị trí phần tử chẵn cuối cùng trong ma trận các số nguyên.

**Bài 19:** (*Tìm giá trị âm lẻ lớn nhất*) Viết hàm tìm phần tử âm lẻ lớn nhất trong ma trận.

**Bài 20:** (*Tìm giá trị dương chẵn nhỏ nhất*) Viết hàm tìm phần tử chẵn dương và nhỏ nhất trong ma trận.

## BÀI THỰC HÀNH TUẦN 12 – BUỔI 02

### BÀI THỰC HÀNH TỔNG HỢP 05

#### ❖ Mục đích:

1. Ôn tập các bài toán trên mảng 1 chiều, mảng 2 chiều và chuỗi ký tự
2. Áp dụng mảng để giải các bài toán liên quan trên máy tính

#### I. Mảng 1 chiều

Khai báo và khởi tạo mảng 1 chiều

Cú pháp:

**<Kiểu dữ liệu> <Tên biến>[<Số phần tử>];**

Ví dụ:

```
int Mang1Chieu[3]; // khai bao mang 1 chieu 3 phan tu
int temp[4] = {50, 70, 60, 48}; // khai bao va khoi tao
```

Truyền mảng 1 chiều cho hàm

Trong C++, mảng được quản lý thông qua địa chỉ của phần tử đầu tiên. Do đó, mảng được truyền cho hàm dưới dạng **truyền tham chiếu**. Số phần tử của mảng có thể để trống nhưng phải có cặp ngoặc vuông. Số phần tử thực sự sử dụng được truyền thông qua một biến khác.

```
void NhapMang1Chieu(int a[], int &n);
```

Xử lý mảng 1 chiều: Để xử lý các phần tử trong mảng 1 chiều chúng ta sử dụng một vòng lặp.

```
for(int pos = 0; pos < n; pos++)
    if(a[pos] thỏa điều kiện)
        Xử lý a[pos];
```

Ví dụ:

```
/******
- Ten ham: TinhTrungBinh
- Cong viec: Ham nay tinh va tra ve trung binh cac gia tri
- Du lieu vaonhT: Mang cac so nguyen va kích thước
- Du lieu ra: trung bình các giá trị trong mảng
*****/
```

---

```
float TinhTrungBinh(const int a[], int Size)
{
    int sum = 0;
    for(int pos = 0; pos < Size; pos++)
        sum += a[pos]; //Cong cac gia tri vao tong
    return float(sum)/Size;
}
```

---

## II. Mảng 2 chiều

Khai báo và khởi tạo

Cú pháp:

**<Kiểu dữ liệu> <Tên biến>[<Số dòng>][<Số cột>;**

Ví dụ:

```
int Mang2Chieu[3][5]; //khai bao mang 2 chieu 3 dong, 5 cot
int temp[4][3] = {50, 70, 60, 48 }; // khai bao va khoi tao
```

---

Truy xuất các phần tử của mảng

Một phần tử trong mảng hai chiều được truy xuất thông qua chỉ số dòng vào chỉ số cột.

Cú pháp:

**<Tên biến>[<chỉ số dòng>][<chỉ số cột>;**

Ví dụ:

```
int Mang2Chieu[3][5]; //khai bao mang 2 chieu 3 dong, 5 cot
Mang2Chieu[0][3]; //phan tu tai dong 0 cot 3
```

---

Truyền mảng cho hàm

Mảng hai chiều được truyền cho hàm theo cách truyền tham chiếu. Số dòng có thể để trống, nhưng **số cột phải được xác định** cụ thể một giá trị số nguyên. Số dòng và số cột thực dùng được truyền thông qua hai biến số nguyên.

---

```
//Nhap mang
void NhapMang2Chieu(int a[][MAX_COLS], int &m, int &n);

//Tinh toan
int TinhTongCacPhanTu(int a[][MAX_COLS], int m, int n);
```

---

Xử lý các phần tử trong mảng

Để xử lý tất cả các giá trị được lưu trong mảng ta có thể sử dụng cú pháp sau:

---

```
//Mảng hai chiều m dòng, n cột thực dụng
for(int row_pos = 0; row_pos < m; row_pos++)
    for(int col_pos = 0; col_pos < n; col_pos++)
        if(a[row_pos][col_pos] thỏa điều kiện)
            xử lý a[row_pos][col_pos];
```

---

### III. Chuỗi

Chuỗi là mảng một chiều các ký tự. Do đó, các thao tác xử lý trên mảng một chiều có thể được áp dụng để xử lý chuỗi. Ngoài ra, C++ cung cấp một số hàm tiện ích cho việc xử lý chuỗi. Bạn cũng có thể sử dụng lớp string trong thư viện string của C++ để xử lý chuỗi. Trong thư viện string C++ cung cấp sẵn các hàm thành viên cho mỗi đối tượng thuộc lớp string giúp cho việc xử lý chuỗi nhanh hơn và thuận lợi hơn.

### IV. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.

❖ **Mảng 1 chiều:**

❖ **Thao tác tách/ghép mảng**

**Bài 1:** (*Tách mảng chẵn, lẻ*) Tách 1 mảng các số nguyên thành 2 mảng a và b, sao cho mảng a chứa toàn số lẻ và mảng b chứa toàn số chẵn.

**Bài 2:** (*Ghép mảng chẵn đầu, lẻ cuối*) Cho 2 mảng số nguyên a và b kích thước lần lượt là n và m. Viết chương trình nối 2 mảng trên thành mảng c theo nguyên tắc chẵn ở đầu mảng và lẻ ở cuối mảng.

❖ **Thao tác sắp xếp**

**Bài 3:** (*Sắp xếp giảm dần*) Sắp xếp mảng theo thứ tự giảm dần.

**Bài 4:** (*Sắp xếp số nguyên tố tăng dần*) Sắp xếp các phần tử là số nguyên tố theo thứ tự tăng dần.

**Bài 5:** (*Sắp xếp số lẻ tăng dần*) Sắp xếp các phần tử lẻ tăng dần.

**Bài 6:** (*Sắp xếp số chẵn tăng, số lẻ giảm dần*) Sắp xếp các phần tử chẵn nằm bên trái theo thứ tự tăng dần, còn các phần tử lẻ bên phải theo thứ tự giảm dần.

**Bài 7:** (*Sắp xếp số âm giảm dần, dương tăng dần*) Sắp xếp các phần tử âm giảm dần từ trái sang phải, các phần tử dương tăng dần từ phải sang trái.

**Ví dụ:** Mảng ban đầu: -1 2 -4 6 -7 8 -3 4

Mảng kết quả: -1 -3 -4 -7 8 6 4 2

❖ **Thao tác thêm/xóa.**

**Bài 8:** (*Xóa các số lớn nhất*) Xóa phần tử có giá trị lớn nhất trong mảng.

**Bài 9:** (*Xóa các số nhỏ hơn X*) Xóa tất cả các phần tử có giá trị nhỏ hơn x trong mảng. Với x nhập vào từ bàn phím.

**Bài 10:** (*Xóa các số gần X nhất*) Xóa phần tử có giá trị gần x nhất. Với x nhập vào từ bàn phím.

**Bài 11:** (*Chèn X vào đầu mảng*) Chèn phần tử x vào vị trí đầu tiên trong mảng. Với x nhập vào từ bàn phím.

**Bài 12:** (*Chèn X vào sau số lớn nhất*) Chèn phần tử x vào phía sau vị trí có giá trị lớn nhất trong mảng. Với x nhập vào từ bàn phím.

**Bài 13:** (*Chèn X vào trước số nguyên tố đầu tiên*) Chèn phần tử x vào phía trước phần tử có giá trị là số nguyên tố đầu tiên trong mảng. Với x nhập vào từ bàn phím.

**Bài 14:** (*Chèn X sau các số chẵn*) Chèn phần tử x vào phía sau tất cả các phần tử có giá trị chẵn trong mảng. Với x nhập vào từ bàn phím.

❖ **Mảng 2 chiều**

❖ **Thao tác đếm**

**Bài 15:** (*Đếm số lượng âm, dương*) Viết hàm đếm các giá trị âm, dương trong ma trận các số thực.



**Bài 16:** (*Đếm số lượng chẵn, lẻ*) Viết hàm đếm các giá trị chẵn, lẻ trong ma trận các số nguyên.

**Bài 17:** (*Đếm số lượng giá trị bằng X*) Viết hàm đếm số lần xuất hiện phần tử x trong ma trận các số thực. x nhập từ bàn phím.

**Bài 18:** (*Đếm số lượng giá trị nhỏ hơn X*) Viết hàm đếm các giá trị nhỏ hơn x trong ma trận các số thực.

**Bài 19:** (*Đếm số lượng số nguyên tố thuộc 2 đường chéo*) Viết hàm đếm các giá trị nguyên tố trên 2 đường chéo (chính, phụ) của ma trận vuông các số nguyên.

**Bài 20:** (*Đếm số lượng giá trị lớn nhất*) Viết hàm đếm các giá trị cực đại trong ma trận các số nguyên.

❖ **Thao tác sắp xếp**

**Bài 21:** (*Sắp xếp ma trận tăng từ trên xuống và trái sang phải*) Viết hàm sắp xếp ma trận theo thứ tự tăng dần từ trên xuống dưới và từ trái sang phải theo phương pháp dùng mảng phụ.

**Hướng dẫn:** Đảo ma trận sang mảng một chiều, sắp xếp trên mảng một chiều theo thứ tự tăng dần, sau đó chuyển ngược mảng một chiều thành ma trận kết quả.

**Bài 22:** (*Sắp xếp giảm dần từ trên xuống, trái sang phải*) Viết hàm sắp xếp ma trận theo thứ tự giảm dần từ trên xuống dưới và từ trái sang phải.

**Bài 23:** (*Sắp xếp các dòng tăng dần*) Viết hàm sắp xếp các dòng trên ma trận theo thứ tự tăng dần.

**Bài 24:** (*Sắp xếp các cột giảm dần*) Viết hàm sắp xếp các cột trên ma trận theo thứ tự giảm dần.

**Bài 25:** (*Sắp xếp ziczac ngang*) Viết hàm sắp xếp ma trận theo đường ziczac ngang.

❖ **Thao tác thêm, xóa, thay thế.**

**Bài 26:** (*Xóa dòng i*) Viết hàm xóa một dòng i trên ma trận.

**Bài 27:** (*Xóa cột j*) Viết hàm xóa một cột j trên ma trận.

**Bài 28:** (*Xóa dòng có tổng lớn nhất*) Viết hàm xóa dòng có tổng lớn nhất trên ma trận.

**Bài 29:** (*Hoán vị dòng có tổng lớn nhất và nhỏ nhất*) Viết hàm hoán vị dòng có tổng lớn nhất với dòng có tổng nhỏ nhất.

**Bài 30:** (*Thay giá trị X bằng Y*) Viết hàm thay thế những phần tử có giá trị x thành phần tử có giá trị y trong ma trận. Với x, y nhập từ bàn phím.

## BÀI THỰC HÀNH TUẦN 13 – BUỔI 01

### Kiểu dữ liệu cấu trúc - STRUCT

#### ❖ Mục đích:

1. Giới thiệu khái niệm một cấu trúc - struct
2. Sử dụng cấu trúc như các tham số

#### I. Dữ liệu kiểu cấu trúc - struct

Định nghĩa kiểu dữ liệu cấu trúc:

Cú pháp:

---

```
struct <Tên kiểu cấu trúc>
{
    <Kiểu dữ liệu> <Tên thành phần 1>;
    <Kiểu dữ liệu> <Tên thành phần 2>;
    ...
    <Kiểu dữ liệu> <Tên thành phần n>;
};
<Tên kiểu cấu trúc> <Tên biến cấu trúc>;
```

---

Khai báo biến cấu trúc:

+ Cách 1: Khai báo biến cấu trúc ngay khi định nghĩa kiểu dữ liệu cấu trúc

---

```
struct <Tên kiểu cấu trúc>
{
    <Kiểu dữ liệu> <Tên thành phần 1>;
    <Kiểu dữ liệu> <Tên thành phần 2>;
    ...
    <Kiểu dữ liệu> <Tên thành phần n>;
}<Tên biến cấu trúc 1>, <Tên biến cấu trúc 2>, ...;
```

---

+ Cách 2:

<Tên kiểu cấu trúc> <Tên biến cấu trúc>;

## II. Truy xuất các thành viên cấu trúc

Cú pháp:

---

**<Tên biến cấu trúc>.<Tên thành phần>**

---

Ví dụ 13.1:

---

```

/*Thông tin CT & TG
Chương trình minh họa kiểu dữ liệu cấu trúc
Họ tên:
MSSV:
Lop:
Ngày:
*/
//Khai báo thu vien
#include<iostream>
#include<cmath>
#include<iomanip>
using namespace std;
//Định nghĩa cấu trúc
struct HINH_TRON
{
    float x;
    float y;
    float BanKinh;
    float DienTich;
    float ChuVi;
    float KhoangCachToiGocToaDo;
};

const float PI = 3.14159;

//Ham chinh
void main()
{
    //Khai báo biến
    HINH_TRON Hinh1, Hinh2; //Khai báo 2 cấu trúc hình tròn
    //Nhập liệu
    //Nhập liệu cho hình 1
    cout<<"Nhập thông tin hình tròn 1: "<<endl;
    cout<<"Hãy nhập vào bán kính: "<<endl;
    cin>>Hinh1.BanKinh;
    cout<<"Hãy nhập tọa độ tâm (x, y): "<<endl;

```

---

---

```

cin>>Hinh1.x>>Hinh1.y;

//Nhap lieu cho hinh 2
cout<<"Nhap thong tin hinh tron 2: "<<endl;
cout<<"Hay nhap vao ban kinh: "<<endl;
cin>>Hinh2.BanKinh;
cout<<"Hay nhap toa do tam (x, y): "<<endl;
cin>>Hinh2.x>>Hinh2.y;

//Xu ly tinh toan
//Tinh toan hinh tron 1
Hinh1.DienTich = PI *pow(Hinh1.BanKinh, 2);
Hinh1.ChuVi = 2*PI*Hinh1.BanKinh;
Hinh1.KhoangCachToiGocToaDo = sqrt(pow(Hinh1.x, 2) +
                                     pow(Hinh1.y, 2));

cout<<endl;

//Tinh toan hinh tron 2
Hinh2.DienTich = PI *pow(Hinh2.BanKinh, 2);
Hinh2.ChuVi = 2*PI*Hinh2.BanKinh;
Hinh2.KhoangCachToiGocToaDo = sqrt(pow(Hinh2.x, 2) +
                                     pow(Hinh2.y, 2));

//Xac dinh tam cua hinh tron gan goc toa do
if(Hinh1.KhoangCachToiGocToaDo >
    Hinh2.KhoangCachToiGocToaDo)
{
    cout<<"Hinh tron 1 xa goc toa do hon hinh tron
        2."<<endl;
}
else if(Hinh1.KhoangCachToiGocToaDo <
    Hinh2.KhoangCachToiGocToaDo)
{
    cout<<"Hinh tron 1 gan goc toa do hon hinh tron
        2."<<endl;
}
else
    cout<<"Hai hinh tron cach goc toa do bang
        nhau."<<endl;

cout<<endl<<endl;

cout<<setprecision(2)<<fixed<<showpoint;

cout<<"Dien tich hinh tron 1:"<<Hinh1.DienTich<<endl;
cout<<"Chu vi hinh tron 1:"<<Hinh1.ChuVi<<endl;

```

---

---

```

    cout<<"Dien tich hinh tron 2:"<<Hinh2.DienTich<<endl;
    cout<<"Chu vi hinh tron 2:"<<Hinh2.ChuVi<<endl;
}

```

---

### III. Khởi tạo cấu trúc

Cách 1: Khởi tạo giá trị ngay khi khai báo biến:

---

```

struct HINH_TRON
{
    float x;
    float y;
    float BanKinh;
}Hinh1 = {3.4, 5.5, 4};

```

---

Hoặc

---

```

struct HINH_TRON
{
    float x;
    float y;
    float BanKinh;
};
HINH_TRON Hinh1 = {3.4, 5.5, 4};

```

---

Cách 2: Khởi tạo giá trị cho các thành phần đầu tiên:

---

```

struct HINH_TRON
{
    float x;
    float y;
    float BanKinh;
};
HINH_TRON Hinh1 = {3.4}; //Chỉ khai tạo cho thành phần x
HINH_TRON Hinh2 = {3.4, 5.5}; //Khai tạo cho thành phần x và y

```

---

### IV. Bài tập:

**Dữ liệu cho các bài tập dưới đây phải được định nghĩa ở dạng cấu trúc.**

**Bài 1: (Rút gọn phân số)** Viết chương trình nhập vào một phân số. Rút gọn phân số vừa nhập (nếu được).

**Bài 2: (Tính tổng, hiệu, tích, thương hai phân số)** Viết chương trình nhập vào hai phân số. Tính tổng, hiệu, tích và thương của hai phân số vừa nhập (các phân số kết quả phải ở dạng tối giản).

**Bài 3: (So sánh phân số)** Viết chương trình nhập vào một phân số. So sánh 2 phân số, trả về 1 trong 3 giá trị: 0 – nếu 2 phân số bằng nhau, -1 – nếu phân số 1 nhỏ hơn phân số 2, 1 – nếu phân số 1 lớn hơn phân số 2.

**Bài 4: (Tính khoảng cách giữa 2 điểm)** Viết chương trình định nghĩa cấu trúc điểm trong mặt phẳng. Hãy viết chương trình nhập và tọa độ 2 điểm. Sau đó, tính khoảng cách giữa hai điểm vừa nhập.

**Bài 5: (Tính đạo hàm đơn thức)** Viết chương trình nhập vào một đơn thức. Tính đạo hàm của đơn thức vừa nhập.

**Bài 6: (Tính tích, thương 2 đơn thức)** Viết chương trình nhập vào hai đơn thức. Tính tích và thương của hai đơn thức vừa nhập.

## BÀI THỰC HÀNH TUẦN 13 – BUỔI 02

### Kiểu dữ liệu cấu trúc - STRUCT

#### ❖ Mục đích:

1. Phát triển và vận dụng một mảng cấu trúc
2. Sử dụng cấu trúc lồng nhau

#### I. Mảng cấu trúc

#### II. Cấu trúc lồng nhau

Thường thì rất hữu ích khi lồng một cấu trúc vào trong một cấu trúc khác. Cùng xem xét ví dụ sau:

Ví dụ 13.2:

---

```
/*Thông tin CT & TG
Chương trình minh họa kiểu dữ liệu cấu trúc lồng nhau
Ho ten:
MSSV:
Lop:
Ngày:
*/
//Khai báo thu vien
#include<iostream>
#include<cmath>
#include<iomanip>
using namespace std;
//Định nghĩa cấu trúc
struct TOA_DO_TAM
{
    float x;
    float y;
};
struct HINH_TRON
{
    float BanKinh;
    float DienTich;
    float ChuVi;
    float KhoangCachToiGocToaDo;
    TOA_DO_TAM Tam;
};
```

---



---

```
const float PI = 3.14159;

//Ham chinh
void main()
{
    //Khai bao bien
    HINH_TRON HìnhTron1, HìnhTron2; //Khai bao 2 cau truc hình
                                    tron

    //Nhap lieu
    //Nhap lieu cho hình 1
    cout<<"Nhap thong tin hình tron 1: "<<endl;
    cout<<"Hay nhap vao ban kính: "<<endl;
    cin>>HìnhTron1.BanKính;
    cout<<"Hay nhap toa do tam (x, y): "<<endl;
    cin>>HìnhTron1.Tam.x>>HìnhTron1.Tam.y;

    //Nhap lieu cho hình 2
    cout<<"Nhap thong tin hình tron 2: "<<endl;
    cout<<"Hay nhap vao ban kính: "<<endl;
    cin>>HìnhTron2.BanKính;
    cout<<"Hay nhap toa do tam (x, y): "<<endl;
    cin>>HìnhTron2.Tam.x>>HìnhTron2.Tam.y;

    //Xu ly tinh toan
    //Tinh toan hình tron 1
    HìnhTron1.DienTich = PI *pow(HìnhTron1.BanKính, 2);
    HìnhTron1.ChuVi = 2*PI*HìnhTron1.BanKính;
    HìnhTron1.KhoangCachToiGocToaDo = sqrt(pow(HìnhTron1.Tam.x,
                                                2) + pow(HìnhTron1.Tam.y, 2));

    cout<<endl;

    //Tinh toan hình tron 2
    HìnhTron2.DienTich = PI *pow(HìnhTron2.BanKính, 2);
    HìnhTron2.ChuVi = 2*PI*HìnhTron2.BanKính;
    HìnhTron2.KhoangCachToiGocToaDo = sqrt(pow(HìnhTron2.Tam.x,
                                                2) + pow(HìnhTron2.Tam.y, 2));

    //Xac dinh tam cua hình tron gan goc toa do
    if(HìnhTron1.KhoangCachToiGocToaDo >
    HìnhTron2.KhoangCachToiGocToaDo)
    {
        cout<<"Hình tron 1 xa goc toa do hon hình tron 2."<<endl;
    }
}
```

---

---

```

else if(HinhTron1.KhoangCachToiGocToaDo <
        HinhTron2.KhoangCachToiGocToaDo)
{
    cout<<"Hình tròn 1 gần góc tọa độ hơn hình tròn
        2."<<endl;
}
else
    cout<<"Hai hình tròn cách góc tọa độ bằng
        nhau."<<endl;

cout<<endl<<endl;

cout<<setprecision(2)<<fixed<<showpoint;

cout<<"Diện tích hình tròn 1:"<<HinhTron1.DienTich<<endl;
cout<<"Chu vi hình tròn 1:"<<HinhTron1.ChuVi<<endl;
cout<<"Hình tròn 1 có tâm tại (";
cout<<HinhTron1.Tam.x<<","<<HinhTron1.Tam.y<<")"<<endl;

cout<<"Diện tích hình tròn 2:"<<HinhTron2.DienTich<<endl;
cout<<"Chu vi hình tròn 2:"<<HinhTron2.ChuVi<<endl;
cout<<"Hình tròn 2 có tâm tại (";
cout<<HinhTron2.Tam.x<<","<<HinhTron2.Tam.y<<")"<<endl;
}

```

---

### III. Bài tập:

Dữ liệu cho các bài tập dưới đây phải được định nghĩa ở dạng cấu trúc.

**Bài 1:** (*Tính đạo hàm đa thức*) Viết chương trình nhập vào một đa thức. Tính đạo hàm của đa thức vừa nhập.

**Bài 2:** (*Tính diện tích, chu vi tam giác*) Viết chương trình nhập vào một tam giác. Tính diện tích, chu vi của tam giác vừa nhập.

**Bài 3:** (*Tính diện tích, chu vi hình tròn*) Viết chương trình định nghĩa tọa độ một điểm trong mặt phẳng Oxy. Sau đó, định nghĩa cấu trúc hình tròn gồm: tâm là cấu trúc điểm vừa định nghĩa ở trên và bán kính. Hãy nhập vào tọa độ tâm và bán kính của hình tròn. Tính diện tích và chu vi của nó.

# BÀI THỰC HÀNH TUẦN 14 – BUỔI 01

## CON TRỎ - POINTER

### ❖ Mục đích:

1. Giới thiệu các biến con trỏ
2. Con trỏ với dữ liệu kiểu mảng
3. Giới thiệu khái niệm cấp phát động

### I. Các biến con trỏ

Khai báo biến con trỏ

---

<Kiểu dữ liệu> \*<Tên biến con trỏ>;

Ví dụ:

`int *ptr1;`

---

Ví dụ 14.1:

---

```
/*Thông tin CT & TG
Chương trình minh họa con trỏ
Ho ten:
MSSV:
Lop:
Ngày:
*/
//Khai bao thu vien
#include<iostream>
using namespace std;

//Ham chinh
void main()
{
    int one = 10;
    int *ptr1; // ptr1 la mot bien co tro tro toi mot int
    ptr1 = &one; // &one cho biet dia chi, khong phai noi dung
    // cua one duoc dung de gan cho ptr1.
    // Hay nho rang ptr1 chi co the luu giu dia chi.
    // Vi ptr1 giu dia chi cua bien one, ta noi rang
    // ptr1 "tro toi" one.
    cout << "Gia tri cua one la " << one << endl << endl;
```

---

---

```

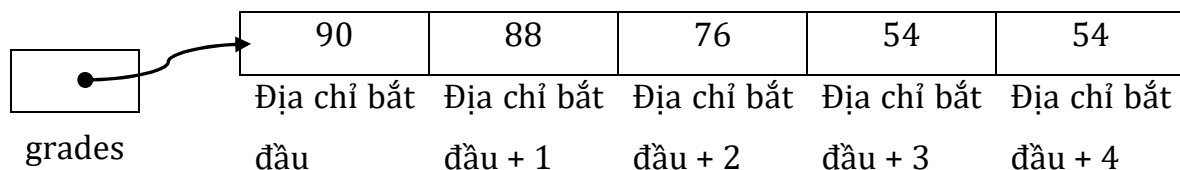
cout << "Gia tri cua &one la " << &one << endl << endl;
cout << "Gia tri cua ptr1 la " << ptr1 << endl << endl;
cout << "Gia tri cua *ptr1 la " << *ptr1 << endl << endl;
}

```

---

## II. Mảng và con trỏ

Khi các mảng được truyền tới các hàm chúng được truyền bằng con trỏ. Tên một mảng là một con trỏ trỏ tới điểm bắt đầu của mảng. Các biến có thể chỉ lưu giữ một giá trị và do đó ta có thể tham chiếu tới giá trị đó chỉ bằng tên biến. Tuy nhiên, mảng lưu giữ nhiều giá trị. Tất cả các giá trị này không thể được tham chiếu chỉ bởi tên mảng. Con trỏ cho phép chúng ta truy cập tất cả các phần tử của mảng. Nhớ rằng tên mảng thực sự là một con trỏ lưu giữ địa chỉ của phần tử đầu tiên trong mảng. Nếu grades là một mảng gồm 5 số nguyên, như hình bên dưới, grades thực chất là một con trỏ trỏ tới vị trí đầu tiên trong mảng, và grades[0] cho phép chúng ta truy xuất nội dung của vị trí đầu tiên đó.



Truy xuất một phần tử của mảng thông qua chỉ số được thực hiện bởi con trỏ số học. Chúng ta có thể truy xuất đến phần tử thứ hai của mảng với phát biểu grades[1], phần tử thứ ba với grades[2], vân vân. Cụm từ "Địa chỉ bắt đầu + 1" trong hình trên nghĩa là di chuyển tới một phần tử từ địa chỉ bắt đầu của mảng. Phần tử thứ ba được truy xuất bằng cách di chuyển qua hai phần tử và vân vân. Hai câu lệnh sau là tương đương:

---

```

cout<<grades[2];
cout<<*(grades + 2);

```

---

Ví dụ 14.2:

---

```

/*Thông tin CT & TG
Chuong trình minh hoa con tro so hoc
Ho ten:
MSSV:
Lop:
Ngay:
*/
//Khai bao thu vien
#include<iostream>
using namespace std;

//Ham chinh
void main()
{
    int grades[] = {90, 88, 76, 54, 34};
    // Khai bao va khoi tao mang cac so nguyen.
    // Vi grades la mot ten mang, nó thực su la mot con tro
    // giu dia chi dau tien cua mang.
    cout << "grade dau tien la: "<< *grades << endl;
    // Dau * truoc grades de lay noi dung cua phan tu dau tien
    cout << "grade thu hai la: "<< *(grades + 1) << endl;
    cout << "grade thu ba la: "<< *(grades + 2) << endl;
    cout << "grade thu tu la: "<< *(grades + 3) << endl;
    cout << "grade thu nam la: "<< *(grades + 4) << endl;
}

```

---

### III. Các biến động

Khai báo biến con trỏ và cấp phát vùng nhớ động:

---

```

<Kiểu dữ liệu> *<Tên biến con trỏ>;
<Tên biến con trỏ> = new <Kiểu dữ liệu>;

```

Ví dụ:

```

int *ptr1;
ptr1 = new int;
*ptr1 = 10;

```

---

Cấp phát số phần tử của mảng động thông qua biến con trỏ

---

```

<Kiểu dữ liệu> *<Tên biến con trỏ>;
<Tên biến con trỏ> = new <Kiểu dữ liệu> [<số phần tử>];

```

Ví dụ:

```

int *ptr1;
int n=10;
ptr1 = new int[n];

```

---

Hủy vùng nhớ do con trỏ trỏ tới

---

```
delete <Tên biến con trỏ>;
hoặc
delete []<Tên biến con trỏ>;
Ví dụ:
delete ptr1;
hoặc
delete []ptr1;
```

---

Ví dụ 14.3:

---

```
/*
Thông tin chương trình và tác giả
Chương trình tính trung bình các điểm số
Họ tên:
MSSV:
Lop:
Ngày:
*/
//Khai báo thu viên
#include <iostream>
#include <iomanip>
using namespace std;

// Nguyên mẫu hàm
void SapXep(float* grades, int numOfGrades);
void HienThiDiem(float* grades, int numOfGrades);

void main()
{
    float *grades; // Một con trỏ sẽ được sử dụng để trỏ tới
    // điểm bắt đầu của mảng các số thực
    float total = 0; // Tổng tất cả các điểm số
    float average; // trung bình của tất cả các điểm
    int numOfGrades; // Số lượng điểm được xử lý
    int count; // bộ đếm vòng lặp
    cout << fixed << showpoint << setprecision(2);
    cout << "Hay nhập số lượng điểm cần xử lý: " << endl;
    cin >> numOfGrades;
    while (numOfGrades <= 0) // kiểm tra giá trị hợp lệ
    {
        cout<<"Phải có ít nhất một điểm số.\n";
        cout << "Hay nhập số lượng điểm cần xử lý: " << endl;
        cin >> numOfGrades;
    }
}
```

---

---

```

    }
    grades = new float[numOfGrades];
    // cap phat vung nho cho mot mang
    // new la mot toan tu cap phat mot mang cac so thuc
    // voi so luong phan tu duoc xac dinh boi nguoi dung.
    // grades la con tro luu tru dia chi dau tien
    // cua mang
    if (grades == NULL)
    {
        cout << "Loi cap phat bo nho!\n";
    }
    cout << "Hay nhap cac diem so: \n";
    for (count = 0; count < numOfGrades; count++)
    {
        cout << "Diem thu " << (count + 1) << ": " << endl;
        cin >> grades[count];
        total = total + grades[count];
    }
    average = total / numOfGrades;
    cout << "Diem trung binh la: " << average << "%" << endl;
    SapXep(grades, numOfGrades);
    HienThiDiem(grades, numOfGrades);
    delete [] grades; // Huy cap phat tat cac bo nho mang
}
//*****
// - Ten Ham: SapXep
// - Cong viec: Sap xep cac diem so trong mang
// - Du lieu vao: mang cac so thuc va so phan tu
// - Du lieu ra: Mang da duoc sap xep
//*****
void SapXep(float* grades, int numOfGrades)
{
    for(int pos1 = 0; pos1 < numOfGrades - 1; pos1++)
    {
        for(int pos2 = pos1 + 1; pos2 < numOfGrades; pos2++)
            if(grades[pos1] > grades[pos2])
            {
                float t = grades[pos1];
                grades[pos1] = grades[pos2];
                grades[pos2] = t;
            }
    }
}
//*****
// - Ten ham: HienThiDiem

```

---

---

```
// - Công việc: hiển thị các giá trị trong mảng
// - Dữ liệu vào: một mảng các số thực và số phần tử của mảng
// - Dữ liệu ra: không có
//*****
void HienThiDiem(float* grades, int numOfGrades)
{
    cout<<"Cac diem co trong mang: "<<endl;
    for(int pos = 0; pos < numOfGrades; pos++)
        cout<<grades[pos]<<" ";
}
```

---

Chú ý rằng mảng động được truyền như một tham số cho các hàm Sắp Xếp và HienThiDiem. Trong trường hợp này, lời gọi hàm chỉ đơn giản truyền tên của mảng, cùng với kích thước của nó như một tham số. Tên của mảng giữ địa chỉ bắt đầu của mảng. Trong tiêu đề hàm, tham số chính thức nhận mảng được định nghĩa là kiểu dữ liệu con trỏ.

#### IV. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính. Dữ liệu của chương trình phải được quản lý bởi con trỏ.

**Bài 1:** (*Xuất giá trị lớn nhất*) Viết chương trình cho phép người dùng nhập vào một dãy các số nguyên. Hãy xuất lên màn hình giá trị lớn nhất có trong dãy vừa nhập.

**Bài 2:** (*Tìm vị trí có giá trị nhỏ nhất*) Viết chương trình cho phép người dùng nhập vào một dãy các số nguyên. Hãy xuất lên màn hình vị trí có giá trị nhỏ nhất có trong dãy vừa nhập.

**Bài 3:** (*Đếm số lượng số nguyên tố*) Viết chương trình cho phép người dùng nhập vào một dãy các số nguyên. Hãy xuất lên màn hình số lượng các số nguyên tố nhỏ hơn 100 có trong dãy vừa nhập.



**Bài 4: (Tính trung bình các số thực)** Viết chương trình cho một số lượng tùy ý các số thực được nhập từ bàn phím. Lưu trữ tất cả các giá trị trong vùng nhớ được cấp phát động trước khi tính toán và hiển thị giá trị trung bình.

**Bài 5: (Tìm giá trị dương đầu tiên)** Viết chương trình cho phép người dùng nhập vào một dãy các số thực. Tìm giá trị dương đầu tiên trong dãy vừa nhập. Nếu dãy không có giá trị dương thì trả về -1.

**Bài 6: (Sắp xếp mảng giảm dần)** Không dùng mảng, hãy nhập một dãy các số nguyên và sắp xếp các giá trị theo chiều giảm dần.

## BÀI THỰC HÀNH TUẦN 14 – BUỔI 02

### CON TRỎ - POINTER

#### ❖ Mục đích:

1. Con trỏ tới kiểu dữ liệu cấu trúc
2. Cấu trúc tự trỏ

#### I. Con trỏ tới kiểu dữ liệu cấu trúc

Ví dụ 14.4:

---

```
/*
Thông tin chương trình và tác giả
Chương trình minh họa con trỏ và cấu trúc
Giới thiệu toán tử mũi tên (->)
Họ tên:
MSSV:
Lop:
Ngày:
*/

//Khai báo thu viện
#include <iostream>
#include <string>
using namespace std;

//Định nghĩa kiểu dữ liệu cấu trúc
struct PHIM
{
    string TuaPhim;
    int NamSanXuat;
};

void main ()
{
    PHIM Phim1;
    PHIM *pPhim;
    pPhim = &Phim1;

    cout << "Hay nhap tua phim: ";
    getline (cin, pPhim->TuaPhim);
    cout << "Hay nhap nam san xuat: ";
    cin>>pPhim->NamSanXuat;
```

---

---

```

cout << "\nBan da nhap:\n";
cout << pPhim->TuaPhim;
cout << " (" << pPhim->NamSanXuat << ")\n";
}

```

---

Câu lệnh	Ý nghĩa	Câu lệnh tương đương
a.b	Thành viên b của biến a	
a->b	Thành viên b của biến được trỏ tới bởi con trỏ a	(*a).b
*a.b	Giá trị được trỏ tới của thành viên b của biến a.	*(a.b)

## II. Cấu trúc tự trỏ

Ví dụ 14.5:

---

```

/*
Thông tin chương trình và tác giả
Chương trình minh họa con trỏ và cấu trúc tự trỏ

Ho ten:
MSSV:
Lop:
Ngày:
*/

//Khai bao thu vien
#include <iostream>
#include <string>
using namespace std;

//Định nghĩa kiểu dữ liệu cấu trúc tự trỏ
struct PHIM
{
    string TuaPhim;
    int NamSanXuat;
    PHIM *pNextPhim; //con trỏ tới phim tiếp theo
};

```

---

---

```
void main ()
{
    PHIM *PhimDauTien = NULL; //Tao danh sach voi khong nut
    PHIM *PhimHienTai = NULL;
    PHIM *PhimTruoc = NULL;

    char cContinue = 'Y';

    while(cContinue == 'Y' || cContinue == 'y')
    {
        PhimHienTai = new PHIM();

        if(PhimTruoc != NULL)
            PhimTruoc->pNextPhim= PhimHienTai;

        cout<<"Hay nhap ten phim: "<<endl;
        cin>>PhimHienTai->TuaPhim;
        cout<<"Hay nhap nam san xuat: "<<endl;
        cin>>PhimHienTai->NamSanXuat;
        PhimHienTai->pNextPhim = NULL;

        PhimTruoc = PhimHienTai;

        if(PhimDauTien == NULL)
            PhimDauTien = PhimHienTai;

        cout<<"Ban co muon them phim moi (y/n)?: ";
        cin>>cContinue;
    }
    //Xuat thong tin cac phim co trong danh sach
    PhimHienTai = PhimDauTien;
    while(PhimHienTai != NULL)
    {
        cout<<PhimHienTai->TuaPhim<<" ("<<PhimHienTai->
                                     NamSanXuat<<" ) ";
        PhimHienTai = PhimHienTai->pNextPhim;
        if(PhimHienTai != NULL)
            cout<<"-> ";
    }
}
```

---

### III. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.

Dữ liệu của mỗi chương trình phải được định nghĩa dưới dạng cấu trúc. Các biến cấu trúc được quản lý bởi các con trỏ cấu trúc.

**Bài 1:** (*Tính trung bình các số thực*) Không dùng mảng, hãy nhập một dãy các số thực và tính trung bình các giá trị có trong dãy vừa nhập.

**Bài 2:** (*Sắp xếp danh sách giảm dần*) Không dùng mảng, hãy nhập một dãy các số nguyên và sắp xếp các giá trị theo chiều giảm dần.

**Bài 3:** (*Tìm node có giá trị bằng X*) Không dùng mảng, hãy nhập một dãy các số nguyên và một số nguyên X. Hãy xuất lên màn hình các giá trị bằng X có trong danh sách vừa nhập.

# BÀI THỰC HÀNH TUẦN 15 – BUỔI 01

## TẬP TIN - FILE

### ❖ Mục đích:

1. Giới thiệu khái niệm cơ bản về tập tin
2. Giới thiệu các thao tác xử lý tập tin văn bản

### I. Tập tin văn bản

Kiểu dữ liệu của một tập tin phụ thuộc vào mục đích nó được sử dụng như là một tập tin đầu vào (được dùng để đọc thông tin vào chương trình), tập tin đầu ra (ghi nội dung thông tin từ chương trình ra tập tin), hoặc cả hai. Các tập tin đầu ra có kiểu dữ liệu được gọi là *ofstream*, các tệp đầu vào có một kiểu dữ liệu *ifstream*, và các tập tin được sử dụng như cả hai có kiểu dữ liệu *fstream*. Chúng ta phải thêm chỉ thị **#include <fstream>** khi sử dụng tập tin.

---

```
ofstream outfile; //khai bao outfile nhu mot tap tin dau ra
ifstream infile; //Khai bao infile nhu mot tap tin dau vao
fstream datafile; //Khai bao datafile de thuc hien ca hai
```

---

### II. Các thao tác tập tin

#### ❖ Mở các tập tin

---

##### Cú pháp:

```
infile.open(<Tên tập tin>);
```

<Tên tập tin> có thể bao gồm cả đường dẫn đến thư mục chứa tập tin.

##### Ví dụ:

```
ifstream infile;
infile.open("payroll.dat");
hoặc:
infile.open("d:/payroll.dat");
if (!infile)
{
    cout<<"Loi mo file. No co the khong ton tai tai noi
ban chi dinh.\n";
}
```

---

#### ❖ Đọc dữ liệu từ tập tin

Cú pháp:

```
infile>><Tên biến>;
```

Ví dụ:

```
int hours; //So gio lam viec
ifstream infile;
infile.open("payroll.dat");
infile >> hours; //Doc gia tri trong tap tin va luu vao bien hours
```

Ta có thể đọc tất cả các nội dung tập tin thông qua cú pháp:

Cú pháp:

```
while (infile)
{
    infile >> <Tên biến>;
}
```

Hoặc:

```
while (infile.peek()!=EOF) //EOF: End Of File
{
    infile >> <Tên biến>;
}
```

Ví dụ 15.1:

```
/*Thong tin chuong trinh va tac gia
Chuong trinh minh hoa doc noi dung tap tin
Ho ten:
MSSV:
Lop:
Ngày:
*/
#include <fstream>
#include <iostream>
#include <iomanip>

using namespace std;
int main()
{
    ifstream infile; //Khai bao tap tin nhap
    ofstream outfile; //Khai bao tap tin xuat

    infile.open("datain.txt"); //Cau lenh nay mo tap tin doc
    outfile.open("dataout.txt");
    // Cau lenh nay se mo tap tin de ghi
```

---

```

int hours; //So giờ làm việc
float payRate; //Đơn giá một giờ làm việc
float net; //Thực trả

if (!infile)
{
    cout << "Loi mo tap tin.\n";
    cout << "Co the tap tin khong ton tai.\n";
}

outfile << fixed << setprecision(2);
infile >> hours; //Doc so giờ làm việc
while (infile)
{
    infile >> payRate; //Doc đơn giá một giờ
    net = hours * payRate; //Tinh thực trả
    outfile << hours << setw(10) << "$ " << setw(6)
    << payRate << setw(5) << "$ " << setw(7) << net << endl;
    infile >> hours;
}
infile.close();
outfile.close();
cout<<"Done!"<<endl;
}

```

---

Giả sử tập tin dữ liệu (datain.txt) có nội dung như sau:

---

```

40 10.00
30 6.70
50 20.00

```

---

Chương trình sẽ tạo ra một dataout.txt như sau:

---

```

40 $ 10.00 $ 400.00
30 $ 6.70 $ 201.00
50 $ 20.00 $1000.00

```

---

### ❖ Tập tin xuất

Tập tin xuất được mở tương tự như: **outfile.open("dataout.txt")**. Bất cứ khi nào chương trình ghi vào outfile, thông tin sẽ được ghi vào tập tin vật lý dataout.txt. Chú ý: chương trình sẽ tạo ra một tập tin được lưu vào cùng thư mục với tập tin nguồn.



Người dùng có thể chỉ định một thư mục khác cho tập tin được lưu trữ bằng cách chỉ ra đường dẫn đầy đủ.

### ❖ Tập tin được dùng cho cả đọc và ghi

Một tập tin có thể được sử dụng cho cả đọc và ghi. Kiểu dữ liệu `fstream` được sử dụng để quản lý cả đọc và ghi, phải có **một cờ trạng thái** truy xuất tập tin như là một đối số cho hàm `open` để chỉ ra kiểu truy xuất, đọc hoặc ghi, có thể được sử dụng. Dưới đây là danh sách các cờ trạng thái truy xuất thường được sử dụng:

Kiểu cờ	Ý nghĩa
<code>ios::in</code>	Kiểu để đọc. Tập tin được sử dụng cho việc đọc thông tin. Nếu tập tin không tồn tại nó sẽ không được tạo ra.
<code>ios::out</code>	Kiểu để ghi. Thông tin được ghi vào tập tin. Nếu tập tin đã tồn tại, nội dung sẽ bị xóa và ghi vào nội dung mới.
<code>ios::app</code>	Kiểu để thêm. Nếu tập tin đã tồn tại, nội dung của nó sẽ được giữ nguyên và tất cả thông tin mới được ghi vào cuối tập tin. Nếu tập tin không tồn tại thì nó sẽ được tạo ra. Chú ý sự khác biệt với <code>ios::out</code>
<code>ios::binary</code>	Kiểu nhị phân. Thông tin được ghi vào hoặc đọc từ một định dạng thuần nhị phân

### ❖ Đóng tập tin

Các tệp nên được đóng lại trước khi chương trình kết thúc để tránh làm hỏng tệp và / hoặc mất dữ liệu có giá trị.

---

```
infile.close();
outfile.close();
```

---

## III. Truyền các tập tin như là các tham số cho hàm

Tập tin có thể được truyền như các tham số giống như các biến. Các tập tin luôn được truyền bằng tham chiếu. Biểu tượng & phải được theo sau kiểu dữ liệu trong phần tiêu đề hàm và nguyên mẫu hàm.

---

```
void GetData(ifstream& infile, ofstream& outfile);
// Nguyen mau ham voi cac tap tin la cac tham so

void GetData(ifstream& infile, ofstream& outfile)
// Tieu de ham voi cac tap tin la cac tham so
```

---

#### IV. Tập tin nhị phân

Cho đến thời điểm này, tất cả các tập tin chúng ta đã nói đến đều là các tệp văn bản, các tập tin được định dạng dưới dạng văn bản ASCII. Ngay cả những con số được ghi vào một tập tin với toán tử << được thay đổi thành văn bản ASCII. ASCII là một mã lưu trữ mỗi số liệu (chữ cái của bảng chữ cái, chữ số, dấu chấm câu, v.v.) dưới dạng ký tự có một số duy nhất. Mặc dù văn bản ASCII là phương pháp mặc định để lưu trữ thông tin trong các tập tin, nhưng chúng ta có thể chỉ định rằng chúng ta muốn lưu trữ dữ liệu ở dạng nhị phân thuần túy bằng cách "mở" một tập tin ở chế độ nhị phân với cờ **ios :: binary**.

---

```
fstream test("grade.dat", ios::out | ios::binary);
// Cau lenh nay khai bao va mo tap tin test
// nhu la tap tin nhi phan de ghi du lieu

int grade[] = {98, 88, 78, 77, 67, 66, 56, 78, 98, 56};
// Khai bao va khoi tao mang cac so nguyen

test.write((char*)grade, sizeof(grade));
// Ghi tat ca cac gia tri cua mang vao tap tin
test.close(); //Dong tap tin
```

---

Chương trình mẫu sau đây khởi tạo một mảng và sau đó ghi các giá trị vào một tập tin như các số nhị phân. Chương trình sau đó cộng thêm 10 cho mỗi phần tử của mảng và in các giá trị đó lên màn hình. Cuối cùng chương trình đọc các giá trị từ cùng một tập tin và in chúng. Những giá trị này là những con số ban đầu. Nghiên cứu chương trình và nhận xét nó một cách cẩn thận.

#### Ví dụ 15.2:

---

```
/*Thong tin chuong trinh va tac gia
Chuong trinh minh hoa xu ly tap tin nhi phan
```

---

---

```
Ho ten:
MSSV:
Lop:
Ngày:
*/
#include <fstream>
#include <iostream>
using namespace std;
const int ARRAYSIZE = 10;
void main()
{
    fstream test("grade.dat", ios::out | ios::binary);
    //chu y su dung dau | de phan tach cac co truy xuat tap tin
    int grade[ARRAYSIZE] = {98,88,78,77,67,66,56,78,98,56};
    int count; // loop counter
    test.write((char*)grade, sizeof(grade));
    //Ghi cac gia tri cua mang vao tap tin
    test.close(); // Dong tap tin
    // Bay gio cong them 10 cho moi phan tu
    cout << "Cac gia tri cua mang duoc cong them 10:\n";
    for (count =0; count < ARRAYSIZE; count++)
    {
        grade[count] = grade[count] + 10;
        // Cau lenh nay cong 10 vao moi phan tu cua mang
        cout << grade[count] << endl;
        //Xuat cac gia tri moi len man hinh
    }
    test.open("grade.dat", ios::in);
    //Moi lai tap tin de doc du lieu
    test.read((char*) grade, sizeof(grade));
    /* Cau lenh trne doc du lieu tu tap tin test va dat cac gia tri
    doc duoc vao mang grade. Cung nhu ham write, doi so day tien la mot
    con tro ky tu mac du mang la cac so nguyen. No tro toi dia chi bat
    dau trong bo nho noi ma thong tin tap tin duoc chuyen len chuong
    trnh
    */
    cout <<"Cac diem duoc doc tu tap tin:" << endl;
    for (count =0; count < ARRAYSIZE; count++)
    {
        cout << grade[count] << endl;
        //Ghi cac gia tri cu len man hinh
    }
    test.close();
}
```

---

## V. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.

Dữ liệu đầu vào của mỗi chương trình phải được lưu trong các tập tin và kết quả đầu ra của chương trình phải được ghi vào tập tin.

**Bài 1:** (*Tạo tập tin*) Viết chương trình cho phép người dùng nhập vào các số bất kỳ. Sau đó, chương trình sẽ ghi các số người dùng nhập vào một tập tin và không xóa tập tin đã tồn tại vì nó sẽ được sử dụng cho các bài tập tiếp theo.

**Bài 2:** (*Đọc nội dung tập tin*) Viết một chương trình đọc tập tin được tạo ra bởi bài tập trước đó, và lấy lại các số một lần theo thứ tự ngược lại và sau đó ghi chúng vào một tập tin mới theo thứ tự chúng đã được lấy ra.

**Bài 3:** (*Tạo tập tin thông tin sinh viên*) Viết một chương trình cho phép người dùng nhập vào thông tin các sinh viên gồm: tên (bao gồm họ và tên), và số điện thoại liên quan từ bàn phím và ghi thông tin vào một tập tin mới nếu một tệp không tồn tại và thêm thông tin vào cuối tập tin nếu tập tin đã tồn tại.

**Bài 4:** (*Tìm thông tin sinh viên*) Viết một chương trình cho phép người dùng nhập vào tên sinh viên (chỉ nhập tên không nhập họ) và chương trình sẽ xuất tất cả các thông tin sinh viên có tên trùng với tên vừa nhập lên màn hình.

## BÀI THỰC HÀNH TUẦN 15 – BUỔI 02

### BÀI THỰC HÀNH TỔNG HỢP 06

#### ❖ Mục đích:

1. Ôn tập các bài toán trên kiểu dữ liệu cấu trúc và xử lý tập tin
2. Áp dụng kiểu dữ liệu cấu trúc và xử lý tập tin để giải các bài toán liên quan

#### I. Kiểu dữ liệu cấu trúc

Kiểu dữ liệu cấu trúc cho phép bạn gom nhóm các phần tử có kiểu dữ liệu khác nhau lại với nhau.

Định nghĩa cấu trúc: một kiểu dữ liệu cấu trúc được định nghĩa trong vùng toàn cục (global section) theo cú pháp sau:

---

```
struct <Tên kiểu cấu trúc>
{
    <Kiểu dữ liệu> <Tên thành phần 1>;
    <Kiểu dữ liệu> <Tên thành phần 2>;
    ...
    <Kiểu dữ liệu> <Tên thành phần n>;
};
```

---

Khai báo biến cấu trúc:

---

```
<Tên kiểu dữ liệu> <Tên biến 1>, <Tên biến 2>;
```

---

Truy xuất các thành phần của cấu trúc: Các thành phần dữ liệu có trong cấu trúc được truy xuất thông qua toán tử dấu chấm (.) theo cú pháp:

---

```
<Tên biến cấu trúc>.<Tên thành phần>;
```

---

#### II. Xử lý tập tin

Khai báo biến tập tin:

---

```
ofstream outfile; //khai bao outfile nhu mot tap tin dau ra
ifstream infile; //Khai bao infile nhu mot tap tin dau vao
fstream datafile; //Khai bao datafile de thuc hien ca hai
```

---

### III. Các thao tác tập tin

#### ❖ Mở các tập tin

##### Cú pháp:

```
infile.open(<Tên tập tin>);
```

<Tên tập tin> có thể bao gồm cả đường dẫn đến thư mục chứa tập tin.

##### Ví dụ:

```
ifstream infile;
infile.open("payroll.dat");
hoặc:
infile.open("d:\\payroll.dat");
if (!infile)
{
    cout<<"Loi mo file. No co the khong ton tai tai noi
ban chi dinh.\n";
}
```

#### ❖ Đọc dữ liệu từ tập tin

##### Cú pháp:

```
infile>><Tên biến>;
```

##### Ví dụ:

```
int hours; //So gio lam viec
ifstream infile;
infile.open("payroll.dat");
infile >> hours; //Doc gia tri trong tap tin va luu vao bien hours
```

##### Ví dụ 15.3:

```
/*Thong tin chuong trinh va tac gia
Chuong trinh minh hoa doc noi dung tap tin
Ho ten:
MSSV:
Lop:
Ngay:
*/
#include <fstream>
#include <iostream>
#include <iomanip>

using namespace std;
```

---

```

int main()
{
    ifstream infile; //Khai bao tap tin nhap
    ofstream outfile; //Khai bao tap tin xuất
    //Cau lenh sau mo tap tin doc
    infile.open("datain.txt");
    //Hoac: infile.open("d:\\datain.txt");
    // Cau lenh sau se mo tap tin de ghi
    outfile.open("dataout.txt");
    // hoac: outfile.open("d:\\dataout.txt");
    int hours; //So gio lam viec
    float payRate; //Don gia mot gio lam viec
    float net; //Thuc tra

    if (!infile)
    {
        cout << "Loi mo tap tin.\n";
        cout << "Co the tap tin khong ton tai.\n";
    }
    outfile << fixed << setprecision(2);
    infile >> hours; //Doc so gio lam viec
    while (infile)
    {
        infile >> payRate; //Doc don gia mot gio
        net = hours * payRate; //Tinh thuc tra
        //Ghi du lieu ra file dataout.txt
        outfile << hours << setw(10) << "$ " << setw(6)
        << payRate << setw(5) << "$ " << setw(7) << net << endl;
        infile >> hours;
    }
    infile.close();
    outfile.close();
    cout<<"Done!"<<endl;
}

```

---

Giả sử tập tin dữ liệu (datain.txt) có nội dung như sau:

---

```

40 10.00
30 6.70
50 20.00

```

---

Chương trình sẽ tạo ra một dataout.txt như sau:

---

```

40 $ 10.00 $ 400.00
30 $ 6.70 $ 201.00
50 $ 20.00 $1000.00

```

---

### ❖ Đóng tập tin

Các tập nên được đóng lại trước khi chương trình kết thúc để tránh làm hỏng tập và / hoặc mất dữ liệu có giá trị.

```
infile.close();  
outfile.close();
```

## IV. Truyền các tập tin như là các tham số cho hàm

Tập tin có thể được truyền như các tham số giống như các biến. Các tập tin luôn được **truyền bằng tham chiếu**. Biểu tượng & phải được theo sau kiểu dữ liệu trong phần tiêu đề hàm và nguyên mẫu hàm.

```
// Nguyen mau ham voi cac tap tin la cac tham so  
void GetData(ifstream& infile, ofstream& outfile);  
// Tieu de ham voi cac tap tin la cac tham so  
void GetData(ifstream& infile, ofstream& outfile)
```

## V. Tập tin nhị phân

```
fstream test("grade.dat", ios::out | ios::binary);  
// Cau lenh nay khai bao va mo tap tin test  
// nhu la tap tin nhi phan de ghi du lieu  
int grade[] = {98, 88, 78, 77, 67, 66, 56, 78, 98, 56};  
// Khai bao va khoi tao mang cac so nguyen  
test.write((char*)grade, sizeof(grade));  
// Ghi tat ca cac gia tri cua mang vao tap tin  
test.close(); //Dong tap tin
```

## VI. Bài tập:

Các bài tập dưới đây phải được viết vào cùng một chương trình. Chương trình cho phép người dùng chọn chức năng tương ứng với bài muốn thực hiện. Mỗi chức năng được viết bởi ít nhất một hàm mà không phải hàm chính.

Dữ liệu đầu vào của mỗi chương trình phải được lưu trong các tập tin và kết quả đầu ra của chương trình phải được ghi vào tập tin.



**Bài 1: (Tạo tập tin)** Viết chương trình cho phép người dùng nhập vào các số bất kỳ. Sau đó, chương trình sẽ ghi các số người dùng nhập vào một tập tin và không xóa tập tin đã tồn tại vì nó sẽ được sử dụng cho các bài tập tiếp theo.

**Bài 2: (Đọc nội dung tập tin)** Viết một chương trình đọc tập tin được tạo ra bởi bài tập trước đó, và lấy lại các số một lần theo thứ tự ngược lại và sau đó ghi chúng vào một tập tin mới theo thứ tự chúng đã được lấy ra.


**Bài 3: (Tạo tập tin thông tin sinh viên)** Viết một chương trình cho phép người dùng nhập vào thông tin các sinh viên gồm: tên (bao gồm họ và tên), và số điện thoại liên quan từ bàn phím và ghi thông tin vào một tập tin mới nếu một tệp không tồn tại và thêm thông tin vào cuối tập tin nếu tập tin đã tồn tại.

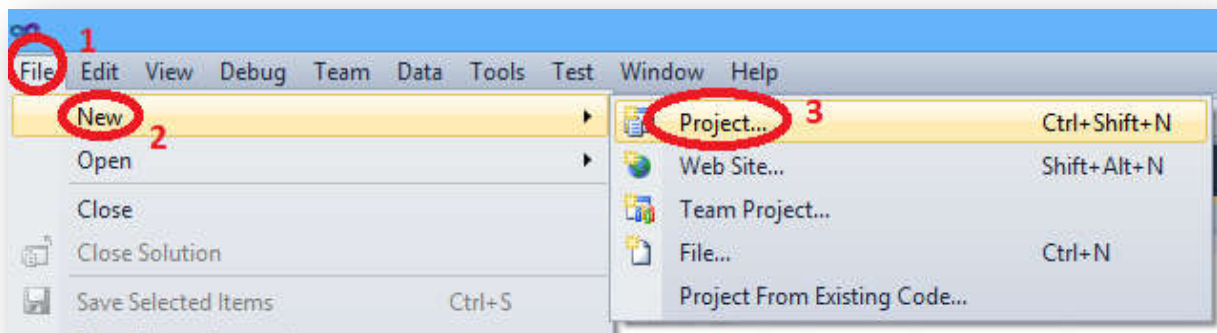
**Bài 4: (Tìm thông tin sinh viên)** Viết một chương trình cho phép người dùng nhập vào tên sinh viên (chỉ nhập tên không nhập họ) và chương trình sẽ xuất tất cả các thông tin sinh viên có tên trùng với tên vừa nhập lên màn hình.

# PHỤ LỤC

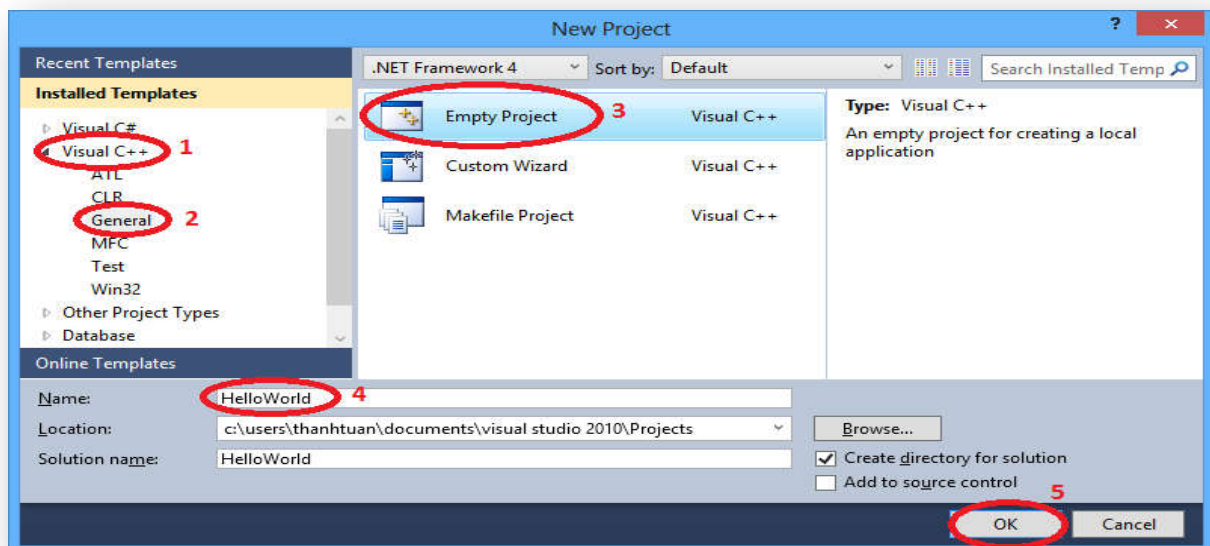
## HƯỚNG DẪN SỬ DỤNG VISUAL STUDIO 2010

### I. Tạo project Visual C++

- Mở Visual Studio 2010 
- Chọn menu *File* → *New* → *Project*

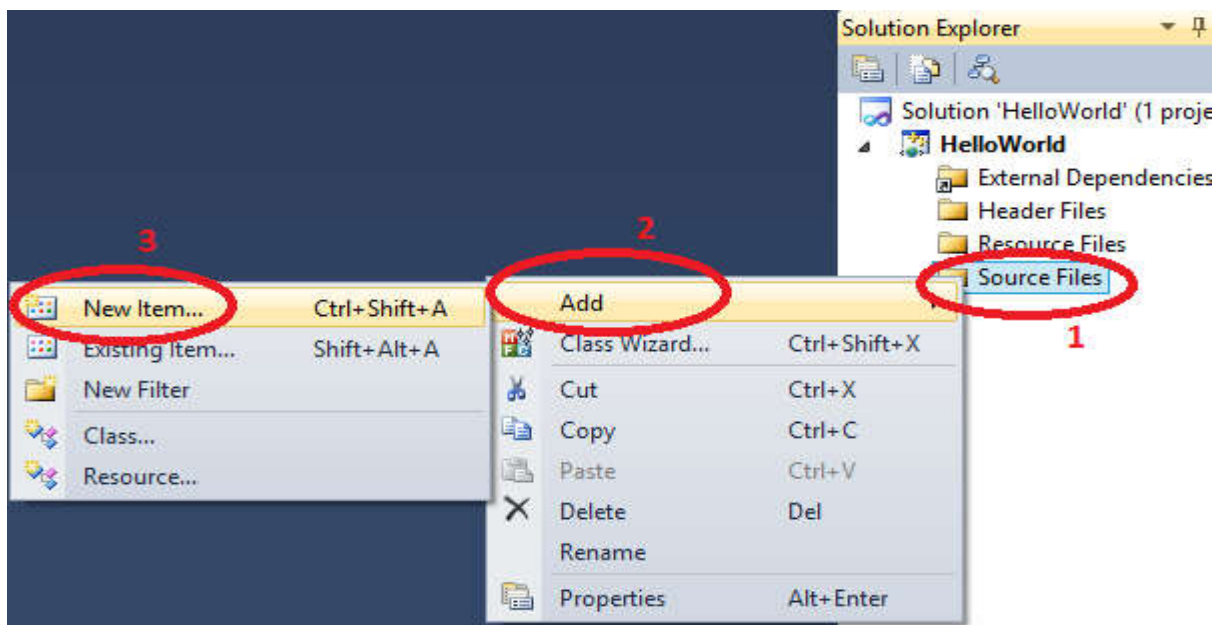


- Chọn loại project C++: **Empty Project**  
Chọn theo thứ tự sau: *Visual C++* / *General* / *Empty Project*

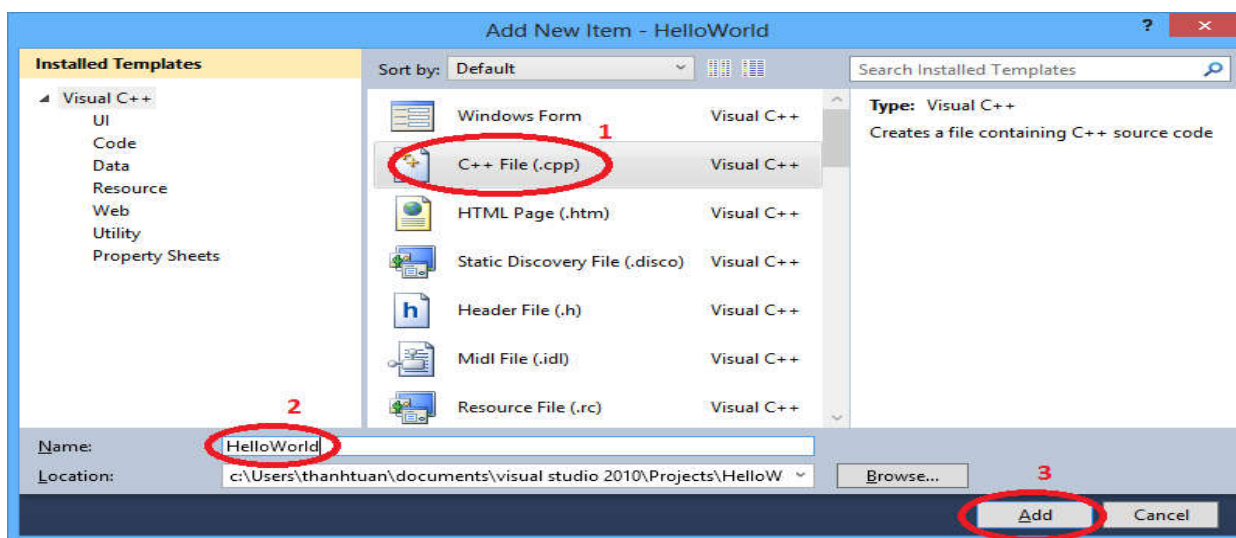


- + **Name**: tên project
- + **Location**: nơi lưu trữ project
- + **Solution name**: tên solution chứa project
- Thêm tập tin mã nguồn vào project

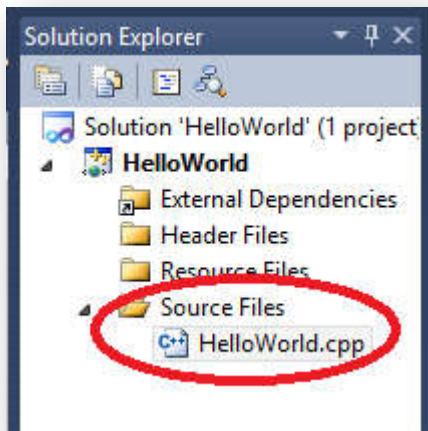
Nhấn chuột phải vào thư mục **Source Files** trong panel **Solution Explorer** → chọn **Add** → Chọn **New Item...**



- Chọn loại tập tin: **C++ File (.cpp)**



- Cấu trúc project C++: xem trong panel **Solution Explorer**



---

```
//Day la chương trình đầu tiên chỉ xuất một thông điệp đơn giản  
//Thêm tên của bạn ở đây
```

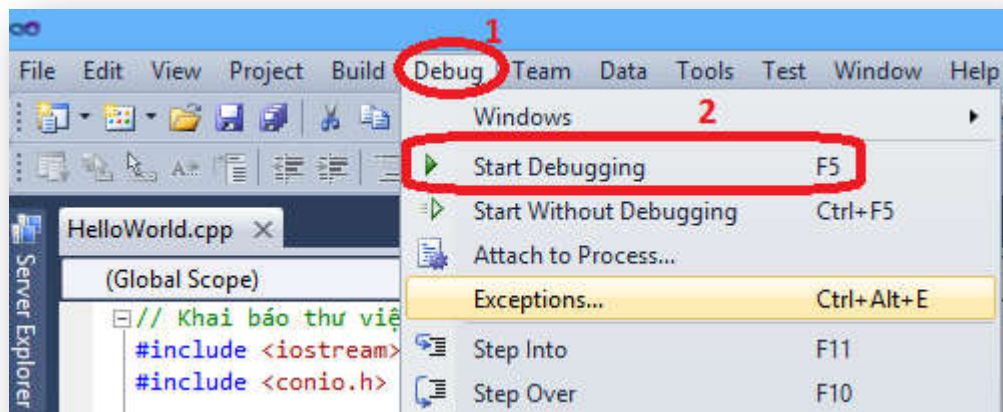
```
//Khai báo thư viện  
#include<iostream>  
using namespace std;  
  
//Hàm chính  
void main()  
{  
    cout<<"Hello World"<<endl;  
}
```

---

Mã nguồn chương trình C++ (cơ bản) được lưu trong thư mục **Source Files**.

## II. Biên dịch và thực thi chương trình

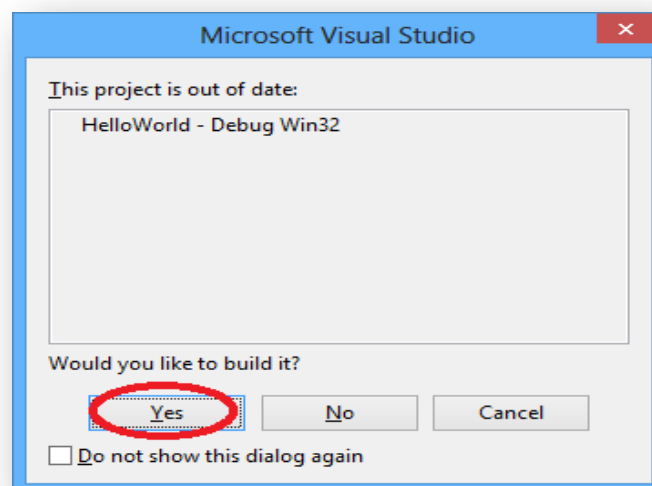
- Thực hiện **một** trong các thao tác sau:
  - + Nhấn phím F5 hoặc CTRL + F5
  - + Chọn menu Debug → Start Debugging



+ Nhấn vào nút  trên thanh công cụ



- Nếu mã nguồn chương trình có thay đổi sẽ có thông báo hỏi ***“Bạn có muốn biên dịch lại mã nguồn không?”***

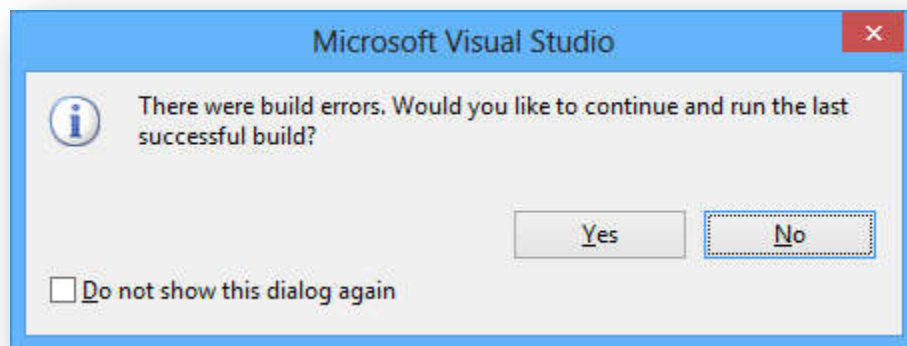


+ **Yes**: biên dịch mã nguồn sau khi thay đổi rồi mới thực thi chương trình → nên chọn

+ **No**: không biên dịch mã nguồn có thay đổi, thực thi chương trình đã được biên dịch trước khi mã nguồn có thay đổi

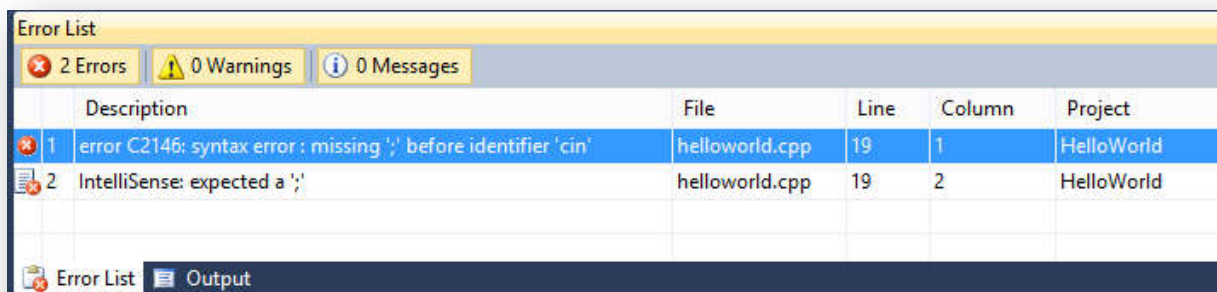
+ **Cancel**: hủy bỏ việc gọi lệnh biên dịch và thực thi chương trình

- Nếu chương trình có lỗi xảy ra thì hiện thông báo như hình bên dưới

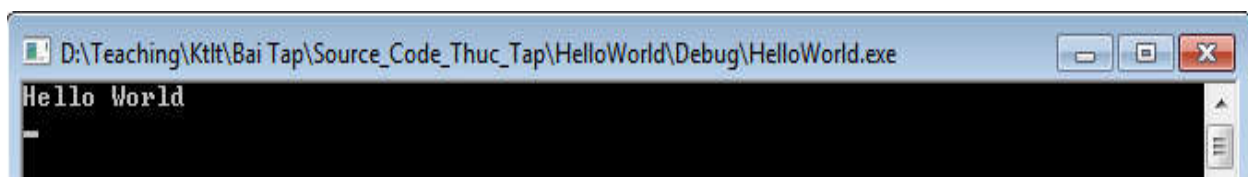


→ nên chọn **No** để tiến hành sửa lỗi chương trình rồi biên dịch lại.

Danh sách lỗi được hiển thị trong khung **Error list**



- Nếu chương trình không có lỗi sẽ thực thi như hình bên dưới



### III. Sửa một số lỗi lập trình

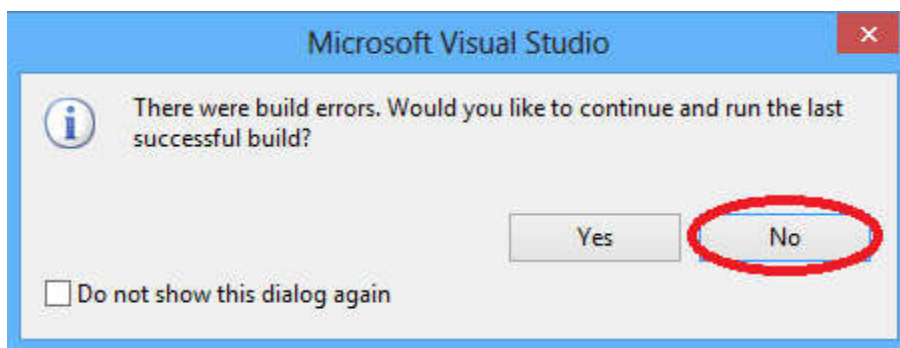
#### 1. Lỗi cú pháp

Khi viết code: những từ khóa, tên thư viện, cú pháp lệnh bị viết sai sẽ được gạch dưới bởi những **đường màu đỏ**.

```

// khởi báo thư viện
#include iostream>
#include <conio.h>
  
```

Sau khi thực hiện biên dịch, nếu chương trình có lỗi sẽ xuất hiện hộp thoại thông báo có lỗi biên dịch như hình bên dưới



+ Chọn **No** để tiến hành sửa lỗi

+ Xem danh sách lỗi trong panel **Error List**

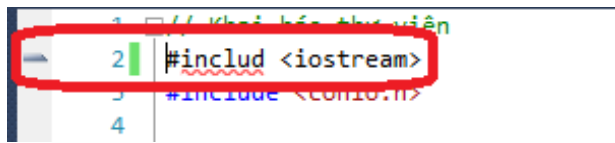
Error List					
<div> <div>5 Errors</div> <div>0 Warnings</div> <div>0 Messages</div> </div>					
	Description	File	Line	Column	Project
1	error C1021: invalid preprocessor command 'includ'	helloworld.cpp	2	1	HelloWorld
2	IntelliSense: unrecognized preprocessing directive	helloworld.cpp	2	2	HelloWorld
3	IntelliSense: name must be a namespace name	helloworld.cpp	5	17	HelloWorld
4	IntelliSense: identifier "cout" is undefined	helloworld.cpp	14	2	HelloWorld
5	IntelliSense: identifier "cin" is undefined	helloworld.cpp	15	2	HelloWorld

+ Nhấn đúp chuột vào từng lỗi và xem trên cửa sổ soạn thảo mã nguồn vị trí của con trỏ → **đó là nơi bị lỗi hoặc lỗi ở trên đó 1 dòng**

+ Hoặc xem ở cột **Line** trong panel **Error List** để xem vị trí dòng bị lỗi



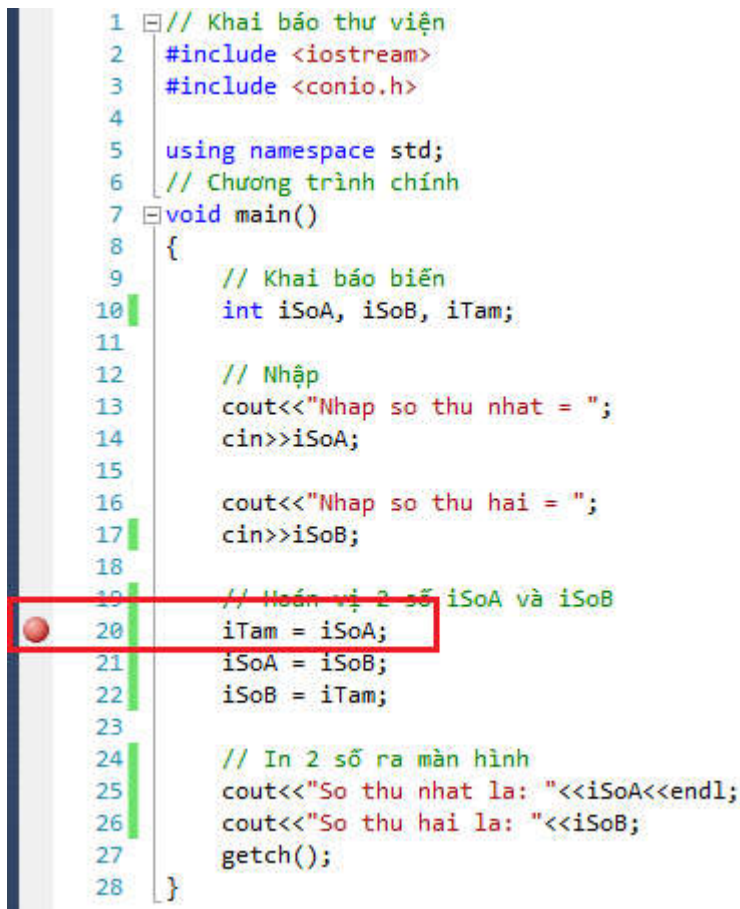
Ví dụ: Vị trí của lỗi số 1 được đánh dấu như hình bên dưới



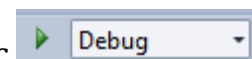
## 2. Lỗi logic

### 2.1. Tạo break point:

- Đưa con trỏ tới dòng cần debug
- Nhấn phím **F9**



### 2.2. Chạy chương trình ở chế độ debug: nhấn phím **F5** hoặc



- Chương trình biên dịch, nếu có lỗi thì phải sửa lỗi chương trình.
- Chương trình sẽ bắt đầu thực thi lần lượt các dòng lệnh trong hàm `main()`
- Chương trình sẽ tạm dừng thực thi tại dòng được đặt break point

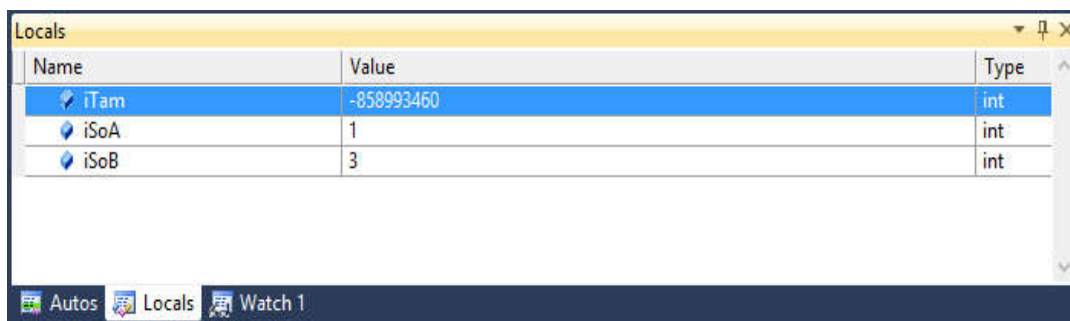


```

1 // Khai báo thư viện
2 #include <iostream>
3 #include <conio.h>
4
5 using namespace std;
6 // Chương trình chính
7 void main()
8 {
9     // Khai báo biến
10    int iSoA, iSoB, iTam;
11
12    // Nhập
13    cout<<"Nhập số thứ nhất = ";
14    cin>>iSoA;
15
16    cout<<"Nhập số thứ hai = ";
17    cin>>iSoB;
18
19    // Hoán vị 2 số iSoA và iSoB
20    iTam = iSoA;
21    iSoA = iSoB;
22    iSoB = iTam;
23
24    // In 2 số ra màn hình
25    cout<<"Số thứ nhất là: "<<iSoA<<endl;
26    cout<<"Số thứ hai là: "<<iSoB;
27    getch();
28 }

```

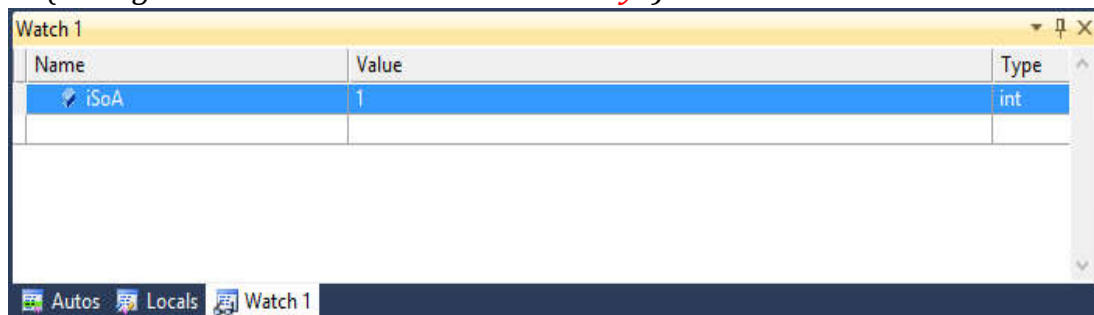
### 2.3. Xem giá trị các biến trong panel Local (Debug → Windows → Local)



Name	Value	Type
iTam	-858993460	int
iSoA	1	int
iSoB	3	int




Có thể xem giá trị các biến bằng cách nhập tên các biến vào panel *Watchxyz*

(Debug → Windows → Watch → Watchxyz)



Name	Value	Type
iSoA	1	int

### 2.4. Di chuyển con trỏ đến các dòng lệnh trong khi debug:

- Step Over  (F10) : di chuyển con trỏ qua từng dòng lệnh
- Step Into  (F11) : di chuyển con trỏ vào bên trong hàm
- Step Out  (Shift + F11) : di chuyển con trỏ ra khỏi khối lệnh, trở về hàm ban đầu

2.5. Tiếp tục chạy chương trình mà không debug: nhấn phím **F5** hoặc



2.6. Dừng debug: nhấn tổ hợp phím **SHIFT + F5** hoặc



2.7. Bỏ break point:

- Đưa con trỏ tới dòng cần có break point
- Nhấn phím F9