CHỦ ĐỀ 1

CHƯƠNG TRÌNH DƠN GIẢN

Các bạn có thể sử dụng code block để có thể biên tập và chạy chương trình C/C++. Link tải bản full (nên chọn bản kèm mingw-setup) tại <http://www.codeblocks.org/downloads>

A. LÝ THUYẾT:

I. CÁC KIỂU DỮ LIỆU CƠ BẢN – PHÉP TOÁN

1.1 Kiểu dữ liệu cơ bản:

Kiểu ký tự (char)

Có hai kiểu dữ liệu char: kiểu signed char và unsigned char.

|  |  |  |  |
| --- | --- | --- | --- |
| **Kiểu** | **Phạm vi biểu diễn** | **Số ký tự** | **Kích thước** |
| Char ( Signed char ) | -128 đến 127 | 256 | 1 byte |
| Unsigned char | 0 đến 255 | 256 | 1 byte |

*Phân loại ký tự:*

Có thể chia 256 ký tự làm ba nhóm:

* Nhóm 1: Nhóm các ký tự điều khiển có mã từ 0 đến 31. Chẳng hạn ký tự mã 13 dùng để chuyển con trỏ về đầu dòng, ký tự 10 chuyển con trỏ xuống dòng dưới ( trên cùng một cột ). Các ký tự nhóm này nói chung không hiển thị ra màn hình.
* Nhóm 2: Nhóm các ký tự văn bản có mã từ 32 đến 126. Các ký tự này có thể được đưa ra màn hình hoặc máy in.
* Nhóm 3: Nhóm các ký tự đồ hoạ có mã số từ 127 đến 255. Các ký tự này có thể đưa ra màn hình nhưng không in ra được ( bằng các lệnh DOS ).

# **Kiểu nguyên**

Trong C++ cho phép sử dụng số nguyên kiểu int, số nguyên dài kiểu long và số nguyên không dấu kiểu unsigned. Kích cỡ và phạm vi biểu diễn của chúng được chỉ ra trong bảng dưới đây:

|  |  |  |
| --- | --- | --- |
| **Kiểu** | **Phạm vi biểu diễn** | **Kích thước** |
| int | -2147483648 đến 2147483647 | 4 byte |
| unsigned int | 0 đến 4294967295 | 4 byte |
| long | -2147483648 đến 2147483647 | 4 byte |
| unsigned long | 0 đến 4294967295 | 4 byte |
| long long | ± 9.10^18 | 8 |
| unsigned long long | 0 .. 18.10^18 | 8 |

**Chú ý** **:** Kiểu ký tự cũng có thể xem là một dạng của kiểu nguyên.

# **Kiểu dấu phẩy động**

Trong C cho phép sử dụng ba loại dữ liệu dấu phẩy động, đó là float, double và long double. Kích cỡ và phạm vi biểu diễn của chúng được chỉ ra trong bảng dưới đây:

|  |  |  |  |
| --- | --- | --- | --- |
| **Kiểu** | **Phạm vi biểu diễn** | **Số chữ số có nghĩa** | **Kích thước** |
| float | 3.4e-38 đến 3.4e+38 | 7 đến 8 | 4 byte |
| double | 1.7e-308 đến 1.7e+308 | 15 đến 16 | 8 byte |
| long double | 3.4e-4932 đến 1.1e4932 | 17 đến 18 | 10 byte |

Giải thích :

Máy tính có thể lưu trữ được các số kiểu float có giá trị tuyệt đối từ 3.4E-38 đến 3.4E+38. Các số có giá trị tuyệt đối nhỏ hơn 3.4E-38 được xem bằng 0. Phạm vi biểu diễn của số double được hiểu theo nghĩa tương tự.

# **Định nghĩa kiểu bằng TYPEDEF**

Công dụng:

Từ khoá typedef dùng để đặt tên cho một kiểu dữ liệu. Tên kiểu sẽ được dùng để khai báo dữ liệu sau này. Nên chọn tên kiểu ngắn và gọn để dễ nhớ. Chỉ cần thêm từ khoá typedef vào trước một khai báo ta sẽ nhận được một tên kiểu dữ liệu và có thể dùng tên này để khai báo các biến, mảng, cấu trúc, vv...

Cách viết:

Viết từ khoá typedef, sau đó kiểu dữ liệu ( một trong các kiểu trên ), rồi đến tên của kiểu.

Ví dụ câu lệnh:

typedef int nguyen;

sẽ đặt tên một kiểu int là nguyen. Sau này ta có thể dùng kiểu nguyen để khai báo các biến, các mảng int như ví dụ sau ;

nguyen x,y,a[10],b[20][30];

Tương tự cho các câu lệnh:

typedef float mt50[50];

Đặt tên một kiểu mảng thực một chiều có 50 phần tử tên là mt50.

typedef int m\_20\_30[20][30];

Đặt tên một kiểu mảng thực hai chiều có 20x30 phần tử tên là m\_20\_30.

Sau này ta sẽ dùng các kiểu trên khai báo:

mt50 a,b;

m\_20\_30 x,y;

1.2 Các phép toán cơ bản:

**Toán tử số học trong C++**

Bảng dưới liệt kê các toán tử số học được hỗ trợ bởi ngôn ngữ C++:

Giả sử biến A giữ giá trị 10, biến B giữ 20 thì:

[**Ví dụ toán tử số học**](http://vietjack.com/cplusplus/toan_tu_so_hoc_trong_cplusplus.jsp)

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Miêu tả** | **Ví dụ** |
| **+** | Cộng hai toán hạng | A + B kết quả là 30 |
| **-** | Trừ toán hạng thứ hai từ toán hạng đầu | A - B kết quả là -10 |
| **\*** | Nhân hai toán hạng | A \* B kết quả là 200 |
| **/** | Phép chia | B / A kết quả là 2 |
| **%** | Phép lấy số dư | B % A kết quả là 0 |
| **++** | [**Toán tử tăng (++)**](http://vietjack.com/cplusplus/toan_tu_tang_giam_trong_cplusplus.jsp), tăng giá trị toán hạng thêm một đơn vị | A++ kết quả là 11 |
| **--** | [**Toán tử giảm (--)**](http://vietjack.com/cplusplus/toan_tu_tang_giam_trong_cplusplus.jsp), giảm giá trị toán hạng đi một đơn vị | A-- kết quả là 9 |

## **Toán tử quan hệ trong C++**

Bảng dưới đây liệt kê các toán tử quan hệ được hỗ trợ bởi ngôn ngữ C++:

Giả sử biến A giữ giá trị 10, biến B giữ 20 thì:

[**Ví dụ toán tử quan hệ**](http://vietjack.com/cplusplus/toan_tu_quan_he_trong_cplusplus.jsp)

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Miêu tả** | **Ví dụ** |
| == | Kiểm tra nếu 2 toán hạng bằng nhau hay không. Nếu bằng thì điều kiện là true. | (A == B) là không đúng |
| != | Kiểm tra 2 toán hạng có giá trị khác nhau hay không. Nếu không bằng thì điều kiện là true. | (A != B) là true |
| > | Kiểm tra nếu toán hạng bên trái có giá trị lớn hơn toán hạng bên phải hay không. Nếu lớn hơn thì điều kiện là true. | (A > B) là không đúng |
| < | Kiểm tra nếu toán hạng bên trái nhỏ hơn toán hạng bên phải hay không. Nếu nhỏ hơn thì là true. | (A < B) là true |
| >= | Kiểm tra nếu toán hạng bên trái có giá trị lớn hơn hoặc bằng giá trị của toán hạng bên phải hay không. Nếu đúng là true. | (A >= B) là không đúng |
| <= | Kiểm tra nếu toán hạng bên trái có giá trị nhỏ hơn hoặc bằng toán hạng bên phải hay không. Nếu đúng là true. | (A <= B) là true |

## **Toán tử logic trong C++**

Bảng dưới đây chỉ rõ tất cả các toán tử logic được hỗ trợ bởi ngôn ngữ C.

Giả sử biến A có giá trị 1 và biến B có giá trị 0:

[**Ví dụ toán tử logic**](http://vietjack.com/cplusplus/toan_tu_logic_trong_cplusplus.jsp)

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Miêu tả** | **Ví dụ** |
| && | Được gọi là toán tử logic AND (và). Nếu cả hai toán tử đều có giá trị khác 0 thì điều kiện trở lên true. | (A && B) là false. |
| || | Được gọi là toán tử logic OR (hoặc). Nếu một trong hai toán tử khác 0, thì điều kiện là true. | (A || B) là true. |
| ! | Được gọi là toán tử NOT (phủ định). Sử dụng để đảo ngược lại trạng thái logic của toán hạng đó. Nếu điều kiện toán hạng là true thì phủ định nó sẽ là false. | !(A && B) là true. |

## **Toán tử so sánh bit trong C++**

Toán tử so sánh bit làm việc trên đơn vị bit, tính toán biểu thức so sánh từng bit. Bảng dưới đây về &, |, và ^ như sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **p** | **q** | **p & q** | **p | q** | **p ^ q** |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 |

Giả sử nếu A = 60; và B = 13; thì bây giờ trong định dạng nhị phân chúng sẽ là như sau:

A = 0011 1100

B = 0000 1101

-----------------

A&B = 0000 1100

A|B = 0011 1101

A^B = 0011 0001

~A  = 1100 0011

Các toán tử so sánh bit được hỗ trợ bởi ngôn ngữ C++ được liệt kê trong bảng dưới đây. Giá sử ta có biến A có giá tri 60 và biến B có giá trị 13, ta có:

[**Ví dụ**](http://vietjack.com/cplusplus/toan_tu_so_sanh_bit_trong_cplusplus.jsp)

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Miêu tả** | **Ví dụ** |
| & | Toán tử AND (và) nhị phân sao chép một bit tới kết quả nếu nó tồn tại trong cả hai toán hạng. | (A & B) sẽ cho kết quả là 12, tức là 0000 1100 |
| | | Toán tử OR (hoặc) nhị phân sao chép một bit tới kết quả nếu nó tồn tại trong một hoặc hai toán hạng. | (A | B) sẽ cho kết quả là 61, tức là 0011 1101 |
| ^ | Toán tử XOR nhị phân sao chép bit mà nó chỉ tồn tại trong một toán hạng mà không phải cả hai. | (A ^ B) sẽ cho kết quả là 49, tức là 0011 0001 |
| ~ | Toán tử đảo bit (đảo bit 1 thành bit 0 và ngược lại). | (~A ) sẽ cho kết quả là -61, tức là 1100 0011. |
| << | Toán tử dịch trái. Giá trị toán hạng trái được dịch chuyển sang trái bởi số các bit được xác định bởi toán hạng bên phải. | A << 2 sẽ cho kết quả 240, tức là 1111 0000 (dịch sang trái hai bit) |
| >> | Toán tử dịch phải. Giá trị toán hạng trái được dịch chuyển sang phải bởi số các bit được xác định bởi toán hạng bên phải. | A >> 2 sẽ cho kết quả là 15, tức là 0000 1111 (dịch sang phải hai bit) |

## **Toán tử gán trong C++**

Dưới đây là những toán tử gán được hỗ trợ bởi ngôn ngữ C++:

[**Ví dụ**](http://vietjack.com/cplusplus/toan_tu_gan_trong_cplusplus.jsp)

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Miêu tả** | **Ví dụ** |
| = | Toán tử gán đơn giản. Gán giá trị toán hạng bên phải cho toán hạng trái. | C = A + B sẽ gán giá trị của A + B vào trong C |
| += | Thêm giá trị toán hạng phải tới toán hạng trái và gán giá trị đó cho toán hạng trái. | C += A tương đương với C = C + A |
| -= | Trừ đi giá trị toán hạng phải từ toán hạng trái và gán giá trị này cho toán hạng trái. | C -= A tương đương với C = C - A |
| \*= | Nhân giá trị toán hạng phải với toán hạng trái và gán giá trị này cho toán hạng trái. | C \*= A tương đương với C = C \* A |
| /= | Chia toán hạng trái cho toán hạng phải và gán giá trị này cho toán hạng trái. | C /= A tương đương với C = C / A |
| %= | Lấy phần dư của phép chia toán hạng trái cho toán hạng phải và gán cho toán hạng trái. | C %= A tương đương với C = C % A |
| <<= | Dịch trái toán hạng trái sang số vị trí là giá trị toán hạng phải. | C <<= 2 tương đương với C = C << 2 |
| >>= | Dịch phải toán hạng trái sang số vị trí là giá trị toán hạng phải. | C >>= 2 tương đương với C = C >> 2 |
| &= | Phép AND bit | C &= 2 tương đương với C = C & 2 |
| ^= | Phép OR loại trừ bit | C ^= 2 tương đương với C = C ^ 2 |
| |= | Phép OR bit. | C |= 2 tương đương với C = C | 2 |

## **Các toán tử hỗn hợp trong C++**

Dưới đây là một số toán tử hỗn hợp quan trọng được hỗ trợ bởi ngôn ngữ C++.

|  |  |
| --- | --- |
| **Toán tử** | **Miêu tả** |
| sizeof | [**Toán tử sizeof trong C++**](http://vietjack.com/cplusplus/toan_tu_sizeof_trong_cplusplus.jsp) trả về kích cỡ của một biến. Ví dụ: sizeof(a), với a là integer, sẽ trả về 4 |
| Điều kiện ? X : Y | [**Toán tử điều kiện trong C++**](http://vietjack.com/cplusplus/toan_tu_dieu_kien_trong_cplusplus.jsp). Nếu Condition là true ? thì nó trả về giá trị X : nếu không thì trả về Y |
| , | [**Toán tử Comma trong C++**](http://vietjack.com/cplusplus/toan_tu_comma_trong_cplusplus.jsp) làm cho một dãy hoạt động được thực hiện. Giá trị của toàn biểu thức comma là giá trị của biểu thức cuối cùng trong danh sách được phân biệt bởi dấu phảy |
| . (dot) và -> (arrow) | [**Toán tử thành viên trong C++**](http://vietjack.com/cplusplus/toan_tu_thanh_vien_trong_cplusplus.jsp) được sử dụng để tham chiếu các phần tử đơn của các lớp, các cấu trúc, và union |
| Cast | [**Toán tử ép kiểu (Casting) trong C++**](http://vietjack.com/cplusplus/toan_tu_ep_kieu_trong_cplusplus.jsp) biến đổi một kiểu dữ liệu thành kiểu khác. Ví dụ: int(2.2000) sẽ trả về 2 |
| & | [**Toán tử con trỏ & trong C++**](http://vietjack.com/cplusplus/toan_tu_con_tro_trong_cplusplus.jsp) trả về địa chỉ của một biến. Ví du: &a; sẽ trả về địa chỉ thực sự của biến này |
| \* | [**Toán tử con trỏ \* trong C++**](http://vietjack.com/cplusplus/toan_tu_con_tro_trong_cplusplus.jsp) là trỏ tới một biến. Ví dụ: \*var sẽ trỏ tới một biến var |

## **Thứ tự ưu tiên toán tử trong C++**

Thứ tự ưu tiên toán tử trong C++ xác định cách biểu thức được tính toán. Ví dụ, toán tử nhân có quyền ưu tiên hơn toán tử cộng, và nó được thực hiện trước.

Ví dụ, x = 7 + 3 \* 2; ở đây, x được gán giá trị 13, chứ không phải 20 bởi vì toán tử \* có quyền ưu tiên cao hơn toán tử +, vì thế đầu tiên nó thực hiện phép nhân 3 \* 2 và sau đó thêm với 7.

Bảng dưới đây liệt kê thứ tự ưu tiên của các toán tử. Các toán tử với quyền ưu tiên cao nhất xuất hiện trên cùng của bảng, và các toán tử có quyền ưu tiên thấp nhất thì ở bên dưới cùng của bảng. Trong một biểu thức, các toán tử có quyền ưu tiên cao nhất được tính toán đầu tiên.

[**Ví dụ**](http://vietjack.com/cplusplus/thu_tu_uu_tien_toan_tu_trong_cplusplus.jsp)

|  |  |  |
| --- | --- | --- |
| **Loại** | **Toán tử** | **Thứ tự ưu tiên** |
| Postfix | () [] -> . ++ - - | Trái sang phải |
| Unary | + - ! ~ ++ - - (type)\* & sizeof | Phải sang trái |
| Tính nhân | \* / % | Trái sang phải |
| Tính cộng | + - | Trái sang phải |
| Dịch chuyển | << >> | Trái sang phải |
| Quan hệ | < <= > >= | Trái sang phải |
| Cân bằng | == != | Trái sang phải |
| Phép AND bit | & | Trái sang phải |
| Phép XOR bit | ^ | Trái sang phải |
| Phép OR bit | | | Trái sang phải |
| Phép AND logic | && | Trái sang phải |
| Phép OR logic | || | Trái sang phải |
| Điều kiện | ?: | Phải sang trái |
| Gán | = += -= \*= /= %=>>= <<= &= ^= |= | Phải sang trái |
| Dấu phảy | , | Trái sang phải |

II. KHAI BÁO HẰNG

- Hằng là một đại lượng có giá trị không thay đổi trong suốt chương trình.

- Cú pháp:

const <kiểu dữ liệu> <Tên hằng> = <Giá trị>;

**-** Ví dụ:

const int n = 2020;

\* Các loại hằng:

- Hằng số

- Hằng xâu: Trong C++ hằng xâu được đặt trong cặp dấu nháy kép “Chao ban”, “ABC”…

III. KHAI BÁO BIẾN

- Biến là một đại lượng mà giá trị của nó có thể thay đổi trong quá trình thực hiện chương trình. Trong C++ ta có thể khai báo biến bất cứ vị trí nào trong chương trình lúc cần đến.

- Cú pháp:

<Tên KDL> <Tên biến>;

Ví dụ:

long long n, m ;

***Chú ý***: *Ta có thể vừa khai báo biến, vừa gán giá trị khởi đầu cho biến bằng cách sử dụng cú pháp như sau*:

long long m = 0, n;

* Lưu ý: Tên biến phân biệt hoa thường **N** và **n** là hai biến khác nhau trong C++

IV. KHAI BÁO THƯ VIỆN

- Cú pháp:

#include <tên thư viện>

- Ví dụ:

#include <bits/stdc++.h>

Khai báo trên có ưu điểm sẽ khai báo thư viện gọi đến tất cả các thư viện khác giúp người mới học không nhớ hết được các thư viện. Tuy nhiên tốc độ có chậm hơn việc khai báo từng thư viện một. Các thư viện khác các em tìm đọc tại [**http://www.cplusplus.com/reference/clibrary/**](http://www.cplusplus.com/reference/clibrary/)

IV. CÂU LỆNH

**5.1. Câu lệnh gán - ghép:**

\* ***Câu lệnh gán***:

<Tên biến> **=** <Biểu thức>;

**-** Ví dụ: S = (a+b)\*2 ;

**\* Câu lệnh ghép:** Sử dụng để ghép nhiều lệnh với nhau thành 1 câu lệnh ghép. Kí hiệu bởi cặp dấu nháy kép {….}

if(delta>=0)**{**

cout <<”pt co hai nghiem”;

cout <<(-b+sqrt(delta))/(2\*a)<<” “ <<(-b+sqrt(delta))/(2\*a);

}

**5.2 Các lệnh xuất nhập dữ liệu**

***5.2.1. Lệnh xuất dữ liệu***

- Cú pháp:

cout <<{kết quả ra} ;

- Trong đó:

+ {kết quả ra} có thể là hằng, biến, biểu thức, hàm

- Ví dụ: Thông báo diện tích S ra màn hình

cout <<"Dien tich la " <<s;

- Để in số thực s làm tròn 3 chữ số thập phân:

cout <<fixed <<setprecision(3) <<s;

- Để in và xuống dòng: thêm kí tự xuống dòng endl hoặc “\n”

cout <<"Chao ban\n";

cout <<"Toi la Phu";

hoặc

cout <<"Chao ban" <<"\n" <<"Toi la Phu";

\*Lưu ý:

- Nên sử dụng “\n” cho tốc độ in ra nhanh hơn

- Trường hợp nhập xuất số lượng dữ liệu lớn thì nên thêm lệnh sau để tắt đồng bộ nhập xuất, giúp cin và cout nhập xuất nhanh hơn.

ios::sync\_with\_stdio(0); cin.tie(0); cout.tie(0);

Các em em vì sao nhanh hơn ở đây:

<https://stackoverflow.com/questions/31162367/significance-of-ios-basesync-with-stdiofalse-cin-tienull>

***5.3.2. Nhập dữ liệu***

- Cú pháp:

cin >>{Tên biến} ;

- Ví dụ: nhập hai cạnh a và b vào:

cin >>a >>b;

- Ví dụ: Nhập vào xâu họ tên S

cin >>s;

Lệnh này không nhận S chứa dấu cách, để nhập xâu chứa dấu cách ta dùng:

getline(cin, s);

\* Lưu ý: Xem và tìm hiểu thêm về lệnh nhập xuất scanf, printf

VI. CẤU TRÚC CHƯƠNG TRÌNH:

#include <bits/stdc++.h>

using namespace std;

const double n = 2020.293;

string s;

int x;

int main(){

…

return 0;

}

VII. MỘT SỐ HÀM TOÁN HỌC

Tóm tắt một số các hàm toán học hay dùng. Các em có thể xem bản đầy đủ tại :

<http://www.cplusplus.com/reference/cmath/>

**1. Các hàm số học**

* abs(x), labs(x), fabs(x) : trả lại giá trị tuyệt đối của một số nguyên, số nguyên dài và số thực.
* pow(x, y) : Trả lại giá trị x lũy thừa y (x^y).
* exp(x) : Trả lại giá trị e mũ x (e^x).
* log(x), log10(x) : trả lại lôgarit cơ số e và lôgarit thập phân của x (lnx, logx) .
* sqrt(x) : trả lại số thực là căn bậc 2 của x
* cbrt(x) : trả lại số thực là căn bậc 3 của x

**2. Các hàm lượng giác**

* sin(x), cos(x), tan(x) : trả lại các giá trị sinx, cosx
* asin(x) : trả về số đo góc a có sin(a) = x. (Đơn vị rad)
* acos(x), atan(x) : tương tự asin

**3. Các hàm làm tròn số:**

* Hàm round(x) : Làm tròn về số nguyên gần nhất so với số thực x.
* Hàm trunc(x) : Trả về số thực có giá trị bằng phần nguyên của x.
* Hàm ceil(x) : Làm tròn lên số thực x. Trả về số thực có giá trị bằng số nguyên nhỏ nhất lớn hơn hoặc bằng x.
* Hàm floor(x) : Làm tròn xuống số thực x. Trả về số thực có giá trị bằng số nguyên lớn nhất nhỏ hơn hoặc bằng x.

Giá trị   round   floor   ceil    trunc

-----   -----   -----   ----    -----

2.3     2.0     2.0     3.0     2.0

3.8     4.0     3.0     4.0     3.0

5.5     6.0     5.0     6.0     5.0

-2.3     -2.0    -3.0    -2.0    -2.0

-3.8     -4.0    -4.0    -3.0    -3.0

-5.5     -6.0    -6.0    -5.0    -5.0

**BÀI TẬP CHƯƠNG TRÌNH CĂN BẢN**

**Bài 1.1:** (**SHCN.CPP**) Lập trình tính và thông báo ra màn hình diện tích hình chữ nhật với hai cạnh a và b là các số nguyên được nhập vào từ bàn phím (1 ≤ a,b ≤ 106)?

**Bài 1.2:** (**BIEUTHUC.CPP**) Lập trình nhập từ bàn phím các số nguyên a, b, c, d và x từ bàn phím. Tính và đưa ra màn hình giá trị biểu thức ax3 + bx2 + cx + d? (1 ≤ a, b, c, d, x ≤ 104)

**Bài 1.3:** (**PHONE.CPP**) Hoa gọi điện thoại cho Huệ để trao đổi bài. Cứ mỗi phút dùng điện thoại phải trả số tiền là x đồng. Cuộc trao đổi kéo dài t phút. Em hãy lập trình tính số tiền mà Hoa phải thanh toán cho cuộc trao đổi này. Với x và t là các số nguyên dương ≤ 10000 được nhập vào từ bàn phím.

**Bài 1.4:** (**STGV.CPP**) Lập trình tính và thông báo ra màn hình diện tích tam giác vuông với hai cạnh góc vuông a và b là các số nguyên (1 ≤ a, b ≤ 106). Kết quả làm tròn 2 chữ số thập phân.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 3 4 | 6.00 |
| 3 5 | 7.50 |

**Bài 1.5:** (**SHTRON.CPP**) Lập trình tính và thông báo ra màn hình diện tích hình tròn có bán kính R là số nguyên dương được nhập vào từ bàn phím. 0 ≤ R ≤ 109. Kết quả làm tròn đến phần nguyên.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 7 | 154 |

**Bài 1.6:** (**VNON.CPP**) Cho một hình nón có chiều cao h và bán kính đáy R. Em hãy lập trình tính thể tích của hình nón nói trên. h và R là các số nguyên dương được nhập vào từ bàn phím. Kết quả làm tròn 8 chữ số thập phân.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 2 7 | 102.62536002 |

**Bài 1.7:** (**TRUNGBINH.CPP**) Lập trình nhập từ bàn phím 2 số thực a và b, tính và đưa ra màn hình trung bình cộng hai số và trung bình cộng của các bình phương 2 số. Kết quả làm tròn 5 chữ số thập phân.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 550369.44000  138661.41600 | 344515.42800  161066754386.51929 |

**Bài 1.8:** (**KC.CPP**) Một người đi xe đạp với tốc độ 10km/h và một người đi xe máy với tốc độ 30km/h cùng xuất phát tại một vị trí, cùng thời điểm và cùng hướng.

Lập trình tính khoảng cách giữa hai người sau t giờ (t là số nguyên dương, t≤15). Dữ liệu t được nhập vào từ bàn phím, kết quả in ra màn hình.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 7 | 140 |

**Bài 1.9:** (**DENBU.CPP**) Để bù đắp thiệt hại cho nhân dân trong công tác phòng chống dịch cúm gia cầm, nhà nước hỗ trợ cho các hộ có gà thiêu hủy theo quy định: Loại 1: dưới một tháng tuổi a đồng/con; Loại 2: trên một tháng và dưới ba tháng là b đồng/con; Loại 3: từ 3 tháng trở lên là c đồng/con.

Một hộ nông dân có n con gia cầm phải tiêu hủy trong đó m con loại 1, k con loại 2 còn lại loại 3. (0 ≤ m, n, k ≤ 100.000)

Lập trình nhập dữ liệu a,b,c,m,n,k từ bàn phím và thông báo số tiền được đền bù của bác nông dân ra màn hình.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 5 5 5 1 6 3 | 30 |

**Bài 1.10:** (**SHTHANG.CPP**) Một hình thang **cân** có đáy lớn là a, đáy bé là b và góc nhọn đáy lớn là α (độ) (Tất cả đại lượng là số thực). Em hãy lập trình tính chu vi và diện tích của hình thang này và thông báo ra màn hình. Kết quả làm tròn 4 chữ số thập phân, các đại lượng a, b, α được nhập vào từ bàn phím.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 2 6 45 | 13.6569  8.0000 |

**Bài 1.11:** (**TONGCS.CPP**) Một số nguyên N có 4 chữ số, em hãy lập trình tính tổng các chữ số của N. Ví dụ nhập vào số N là 2345 thì kết quả là 14 (2+3+4+5 = 14).

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1232 | 8 |
| 2345 | 14 |

**Bài 1.12:** (**HERON.CPP**) Bạn Nam được bạn nhờ tính diện tích của tam giác khi biết độ dài ba cạnh là a, b, c. Em hãy lập trình giúp bạn tính diện tích của tam giác đó với a, b, c là các số nguyên < 3000 được nhập vào từ bàn phím. Kết quả thông báo ra màn hình, làm tròn 2 chữ số thập phân. *(Gợi ý: Sử dụng công thức Heron)*

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 3 4 5 | 6.00 |

**Bài 1.13:** (**QMOBILE.CPP**) Bạn Lan sử dụng điện thoại Q.Mobile, bạn thắc mắc là làm thế nào mà khi có người gọi điện thoại đến thì điện thoại có thể đọc số của người gọi đến. Ví dụ 0942250787 🡪 0 9 4 2 2 5 0 7 8 7.

Em hãy lập trình giúp cho bạn nhập vào một số điện thoại có 10 chữ số và đọc các số này ra màn hình, các số cách nhau một dấu cách.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 942250787 | 9 4 2 2 5 0 7 8 |

**Bài 1.14:** (**TBC2.CPP**) Em hãy lập trình tính trung bình cộng của 4 số nguyên. Với điều kiện chỉ được dùng 2 biến. Kết quả làm tròn 2 chữ số thập phân.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1 2 3 4 | 2.50 |

**Bài 1.15:** (**HOANDOI.CPP**) Lập trình nhập vào từ bàn phím 2 số nguyên a và b, hoán đổi giá trị của chúng cho nhau rồi in kết quả 2 số đã hoán đổi ra màn hình.

**Bài tập:** Lập trình thông báo ra màn hình số lớn hơn trong hai số nguyên a và b được nhập vào từ bàn phím. (0 < a, b < 20000)

\* Xác định bài toán:

Input: A, b

Output: Số lớn hơn

\* Tổ chức dữ liệu: a, b:số nguyên

\* Ý tưởng thuật toán:

Nhập vào 2 số nguyên a và b từ bàn phím.

Nếu a>b thì thông báo a còn không thì thông báo b.

\* Thuật toán:

B1: Nhập a và b

B2: Nếu a> b thì thông báo a còn không thì thông báo b.

B3: Kết thúc.

\* Chương trình:

Em hãy viết chương trình giải bài toán trên? Cho biết gặp phải vấn đề gì chưa làm được?

**CHỦ ĐỀ 2**

CÂU LỆNH RẼ NHÁNH (IF – THEN)

I. LÝ THUYẾT

- Cấu trúc hành động rẽ nhánh:

* *Nếu … thì …*
* *Nếu … thì … còn không thì …*

- Để mô tả cấu trúc rẽ nhánh, C++ dùng câu lệnh IF – else

***a) Cú pháp:***

if (<điều kiện>)

<câu lệnh 1>;

else

<câu lệnh 2>;

***b) Giải thích:***

*- <điều kiện>: Là một biểu thức Logic*

*- <câu lệnh 1>, <câu lệnh 2> : là một câu lệnh của Pascal.*

***c) Ý nghĩa hoạt động:***

*Nếu điều kiện đúng thì thực hiện câu lệnh 1 còn không thì thực hiện câu lệnh 2.*

**d) Ví dụ:**

- Ví dụ 1: kết luận nghiệm của PT bậc 2

if(delta>=0)

cout <<"PT co nghiem";

else

cout <<"PT vo nghiem";

- Ví dụ 2: Viết câu lệnh kiểm tra và thông báo x ra màn hình nếu x là số âm.

- Ví dụ 3: Kiểm tra một số x là số chẵn hay lẻ.

if(x % 2 == 0) cout <<"X la so chan"; else cout <<"X la so le";

- Ví dụ 4: Tìm số bé hơn trong hai số a và b.

if (a<b) min:=a; else min:=b;

**e) Câu lệnh ghép:**

Chú ý: Sau **then** và **Else** là một câu lệnh, trong trường hợp nhiều lệnh thì phải dùng đến lệnh ghép.

- Cú pháp:

{

Dãy câu lệnh

}

- Ví dụ: Nếu Delta > 0 thì ta thực hiện 3 việc: tính x1, x2, thông báo còn không …

if(delta>=0){

cout <<”pt co hai nghiem ”;

cout <<(-b+sqrt(delta))/(2\*a)<<” “ <<(-b+sqrt(delta))/(2\*a);

}

else

cout <<”pt vo nghiem”;

II. BÀI TẬP:

**Bài 2.1:** *Hoàn thiện chương trình cho bài toán mở đầu.*

**Bài 2.2:** (**MAX3.CPP**) Lập trình tìm số lớn nhất (*Max)* trong 3 số nguyên a, b, c được nhập vào từ bàn phím.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 5 9 3 | So lon nhat: 9 |

**Bài 2.3:** (**KTTG.CPP**) Lập trình nhập vào từ bàn phím 3 số nguyên x, y, z là ba cạnh của một tam giác. Kiểm tra và thông báo ra màn hình đây có phải là 3 cạnh của một tam giác cân hay không.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1 2 3 | KHONG PHAI TAM GIAC CAN |
| 3 4 3 | LA TAM GIAC CAN |

**Bài 2.4:** (**BPSA.CPP**) Lập trình nhập vào từ bàn phím ba số nguyên m, n, k. Tính và thông báo ra màn hình bình phương của số âm và lập phương của số dương.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1 2 -3 | 1 8 9 |

**Bài 2.5:** (**CSC.CPP**) Cho ba số nguyên m, n, k (m < n < k). Nếu ba số này lập thành một cấp số cộng thì thì tính tổng 3 số này còn không thông báo không phải là cấp số cộng.

Em hãy lập trình thực hiện yêu cầu trên. Với m, n, k là các số nguyên nhập vào từ bàn phím.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1 2 3 | 6 |
| 1 3 9 | KHONG PHAI CAP SO CONG |

**Bài 2.6:** (**CSN.CPP**) Cho ba số nguyên p, q, r (p < q < r và ≠ 0). Kiểm tra ba số này theo thứ tự nhập vào có tạo thành một cấp số nhân hay không.

Em hãy lập trình thực hiện yêu cầu trên. Kết quả thông báo ra YES hoặc NO tương ứng.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 2 4 6 | YES |
| 2 4 9 | NO |

**Bài 2.7:** (**TOADO.CPP**) Lập trình nhập từ bàn phím hai cặp số nguyên (x,y) (u,v) tương ứng là tọa độ điểm A và B trong mặt phẳng tọa độ. Tính và thông báo ra màn hình điểm nào gần gốc tọa độ hơn. *(Gợi ý: Tính khoảng cách d2 = x2 + y2)*

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1 1  2 2 | A |
| 3 3  1 0 | B |
| 1 1  -1 1 | BANG NHAU |

**Bài 2.8:** (**PTBACNHAT.CPP**) Em hãy lập trình giải phương trình *ax+b=0* với a và b nguyên nhập vào từ bàn phím. Kết quả làm tròn 2 chữ số thập phân.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 0 0 | VO SO NGHIEM |
| 0 1 | VO NGHIEM |
| 1 2 | PT CO NGHIEM  X = -2.00 |

**Bài 2.9:** (**TRANSL.CPP**) Chương trình dịch tiếng Anh các thứ trong tuần.

Em hãy lập trình cho phép người dùng nhập vào một số nguyên từ 2 đến 8 và thông báo thứ tương ứng ra màn hình.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| **Monday** | **Tuesday** | **Wednesday** | **Thursday** | **Friday** | **Saturday** | **Sunday** |

**Câu 2.10:** (**PTBH.CPP**) Viết chương trình giải phương trình: ax2 + bx + c = 0, với a, b, c là các hệ số nguyên nhập vào từ bàn phím. Kết quả thông báo theo yêu cầu, nghiệm làm tròn hai chữ số thập phân.

**Ví dụ:**

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả** |
| 1 2 3 | VO NGHIEM |
| 1 2 -3 | PT CO HAI NGHIEM  X1 = 1.00  X2 = -3.00 |
| 1 -2 1 | PT CO NGHIEM KEP  X = 1.00 |

**Câu 2.11:** (**TGGI.CPP**) Viết chương trình nhập vào ba số thực a, b, c. Sau đó kiểm tra xem ba số này có tạo thành cạnh của tam giác hay không. Nếu là tam giác thì kiểm tra xem đó là loại tam giác nào trong số các loại sau:

VUONG CAN, VUONG, CAN, DEU, THUONG, KHONGPHAITAMGIAC

**CHỦ ĐỀ 3**

LẶP VỚI SỐ LẦN BIẾT TRƯỚC VÀ CÂU LỆNH FOR

I. LÝ THUYẾT

- Trong nhiều trường hợp có một số hành động lặp đi lặp lại với một số lần nhất định.

**VD1** : Mẹ bảo ra rút 10 bao rơm. Trong ví dụ này thì hành động rút một bao rơm được lặp đi lặp lại 10 lần, từ lần 1 đến lần 10: Mỗi lần như vậy ta ra rút một bao rơm mang vào.

**- Trong C++:** Dùng câu lệnh For để mô tả thao tác lặp với số lần biết trước.

***\* Cú pháp:***

**for**(Init statement; Condition; Update statement) **<Câu lệnh>;**

*Trong đó:*

* Init statement là biểu thức khởi tạo các giá trị ban đầu cho các tham số. Các tham số này dùng để kiểm tra điều kiện trong biểu thức Condition.
* Condition là biểu thức điều kiện của vòng lặp. Vòng lặp sẽ kết thúc khi biểu thức này trả về giá trị false.
* Update statement là biểu thức cập nhật giá trị cho các tham số.

***\* Ví dụ:***

(1) *for(int i=1; i<=5; i++) s = s + i ;*

*Trong lệnh (1): i sẽ nhận giá trị lần lượt từ 1 đến 5: Với mỗi i đó thì S:= S+i*

*Như vậy thì i sẽ đi từ 1 đến 5 và S sẽ là: S = 1+ 2 + 3 + 4 + 5.*

*(2)* for(int i=1; i<=5; i+=2) s = s + i ;

*Trong lệnh (2): Ban đầu i=1, sau mỗi lần lặp i tăng lên 2 đơn vị. Vòng lặp dừng lại khi i>5;*

*(3)* for(int i=5; i>=1; i--) s += i ;

Trong lệnh (3): Ban đầu i=5, sau mỗi lần lặp i giảm đi một đơn vị. Nên s = 5 + 4 + 3 + 2 + 1

**\* Lược bỏ một số thành phần trong vòng lặp for**

Một đặc điểm nổi bật hơn so với các vòng lặp khác là vòng lặp for cho phép lập trình viên lược bỏ các thành phần nếu không cần sử dụng. Ví dụ:

int loop = 0

for ( ; loop <= 10; )

{

cout << loop++ << " ";

}

Trong ví dụ trên, mình không cần sử dụng tới thành phần khởi tạo biến, cũng như thành phần cập nhật giá trị biến. Lúc này, vòng lặp này hoàn toàn giống với vòng lặp while.

Chúng ta có thể lược bỏ luôn cả 3 thành phần cơ bản của vòng lặp for:

for ( ; ; )

{

// do something

}

Khi biểu thức điều kiện trong for được bỏ trống, nó đồng nghĩa với việc biểu thức điều kiện luôn luôn đúng.

**\* Nesting for loops:**

Tương tự như vòng lặp while hay do-while, vòng lặp for có thể chứa nhiều vòng lặp khác trong khối lệnh của nó.

for(int i = 0; i < 5; i++) {

for(int j = 0; j < 10; j++){

cout << "\* ";

}

cout << “\n”;

}

Kết quả của đoạn code này là:

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

II. BÀI TẬP:

**Bài tập 3.1:**

*Lập trình tính S.*

*S = 1 + 2 + 3 + 4 + 5 + … + n (Với n nhập vào từ bàn phím)*

*\* Ý tưởng thuật toán:*

*- Ý tưởng:*

* *ban đầu S 🡨 0*
* *Cho i đi từ 1 đến n: S := S + i;*

*Giải thích thêm:*

* *Ta sẽ cho i đi từ 1 đến n mục đích để cộng từ 1 đến n vào S*

*- Thuật toán: Tự xây dựng dựa trên ý tưởng.*

*\* Chương trình:*

#include <bits/stdc++.h>

using namespace std;

long long n, s;

int main(){

cin >>n;

s = 0;

for(int i=1; i<=n; i++) s+=i;

cout <<s;

return 0;

}

**Bài 3.2:** (**TONGLE.CPP**) Lập trình tính tổng các số lẻ từ 1 đến N (1 < N < 1 triệu). Thông báo kết quả ra màn hình.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 5 | 9 |
| 3 | 4 |

**Bài 3.3:** (**TONGCHANAB.CPP**) Với a và b là hai số nguyên được nhập vào từ bàn phím. Em hãy lập trình tính tổng các số chẵn trong phạm vi từ a đến b. (1 < a < b < 226)

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1 5 | 5 |
| 2 4 | 6 |

**Bài 3.4:** (**BOIBA.CPP**) Lập trình giải bài toán sau:

Với m và n là hai số nguyên được nhập vào từ bàn phím (0 < a < b < 106). Đếm số lượng các số chia hết cho 3 trong phạm vi từ a đến b. Đưa ra màn hình kết quả là số lượng đếm được.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1 10 | 3 |
| 1 4 | 1 |

**Bài 3.5:** (**GIAITHUA.CPP**) *Giai thừa*

Lập trình nhập n từ bàn phím (0 < n < 20), tính giai thừa của n (n!).

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 2 | 2 |
| 7 | 5040 |

**Bài 3.6:** (**PHATKEO.CPP**) Một giáo viên phát kẹo cho các em học sinh theo qui tắc sau: em đầu tiên được nhận 1 viên kẹo, em tiếp theo nhận nhiều hơn em trước đó 3 viên. Hỏi nếu có N học sinh thì tổng số kẹo phải có để Giáo viên phát đủ cho cả lớp là bao nhiêu? (0 < N < 45000)

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1 | 1 |
| 2 | 4 |

**Bài 3.7:** (**DUONGLE.CPP**) Lập trình tính tổng các số nguyên dương lẻ nhỏ hơn một số nguyên x cho trước (0 < x < 2.109).

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 6 | 9 |
| 37 | 324 |

**Bài 3.8:** (**MIMA.CPP**) Nhập n rồi nhập vào một dãy n số nguyên bất kỳ từ bàn phím. Tìm giá trị lớn nhất và nhỏ nhất của dãy số đó. in kết quả tìm được ra màn hình. ( 0 < n < 10000)

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 5  1 2 3 4 5 | SO LON NHAT: 5  SO BE NHAT: 1 |

**Bài 3.9:** (**DADD.CPP**) Nhập n số nguyên bất kỳ từ bàn phím. Đếm số các số dương và số các số âm. In kết quả ra màn hình.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 5  1 -2 -3 -4 5 | SO DUONG: 2  SO AM: 3 |

**Bài 3.10:** (**TONGS.CPP**) Tính tổng: S = 1 + 1/2! + 1/3! + ... + 1/n!, với giá trị của n nhập vào từ bàn phím. Kết quả làm tròn 10 chữ số thập phân. (1 < N < 21)

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 62 | 1.7182818285 |

**Bài 3.11**: (**GACHO.CPP**) Em hãy lập trình giải bài toán sau;

*Vừa Gà vừa Chó, Bó lại cho tròn, Ba mươi sáu con, Một trăm chân chẵn.*

Hỏi có bao nhiêu Gà, bao nhiêu Chó?

**Bài 3.12**: (**TRAMTRAU.CPP**) Em hãy lập trình giải bài toán sau:

*Trăm trâu, trăm cỏ; Trâu đứng ăn năm; Trầu nằm ăn ba; Lụ khụ trâu già; Ba con một bó.*

Hỏi có bao nhiêu trâu đứng, bao nhiêu trâu nằm, bao nhiêu trâu già? Nếu có nhiều đáp án thì liệt kê mỗi đáp án một dòng bao gồm 3 số nguyên là số trâu đứng, trâu nằm, trâu già cách nhau dấu cách.

**Bài 3.13:** (**HOANHAO.CPP**) Số hoàn hảo là số có tổng các ước số nhỏ hơn nó bằng chính nó. Ví dụ 28 là số hoàn hảo (28 = 1+2+4+7+14).

Cho một số nguyên n, hãy kiểm tra xem n có phải là số hoàn hảo hay không.

Dữ liệu nhập: Là số nguyên n (2 ≤ n ≤ 107)

Dữ liệu xuất: Nếu n là số hoàn hảo in ra YES. Nếu không in ra NO.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 28 | YES |
| 29 | NO |

**Bài 3.14:** (**SEMAPHORE.CPP**) Tuấn tham gia sinh hoạt Đội, cậu được phát một lá cờ semaphore, lá cờ có dạng là một tam giác vuông cân. Với một số nguyên n, bạn hãy giúp Tuấn vẽ một lá cờ semaphore.

Dữ liệu vào: Nhập số nguyên n ( 1<= n <= 100).

Dữ liệu ra: Là tam giác vuông cân gồm các dấu \*, cạnh góc vuông có chiều dài n. Xem thêm ví dụ để hiểu cách xuất dữ liệu.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 5 | \*\*\*\*\*  \*\*\*\*  \*\*\*  \*\*  \* |

**Bài 3.15:** (**BIPHAT.CPP**) Bạn Nam mất trật tự trong giờ học thể dục nên bị thầy giáo phạt. Hình phạt của thầy như sau: bạn Nam đứng nghiêm, khi thầy hô "trái" thì Nam bước sang trái một mét, thầy hô "phải" thì Nam bước sang phải một mét. Hỏi sau **n** lần thầy hô như vậy thì bạn Nam cách xa vị trí ban đầu bao nhiêu mét?

**Dữ liệu nhập:**

- Dòng thứ nhất là số n (1 ≤ n ≤ 1000).

- Dòng tiếp theo gồm n số 1 hoặc 2, mỗi số cách nhau một khoảng trắng. Nếu là số 1 thì thầy giáo hô "trái", nếu là số 2 thì thầy giáo hô "phải".

**Dữ liệu xuất:** Là khoảng cách của Nam sau n lần hô so với vị trí ban đầu.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 4  1 1 2 1 | 2 |

**CHỦ ĐỀ 4**

LẶP VỚI SỐ LẦN CHƯA BIẾT TRƯỚC VÀ CÂU LỆNH WHILE

I. LÝ THUYẾT

- Trong nhiều trường hợp có một số hành động lặp đi lặp lại nhưng chưa rõ số lần lặp.

**VD1** : Mẹ bảo ra rút một số bao rơm khi nào mẹ cho thì nghỉ. Trong ví dụ này thì hành động rút một bao rơm được lặp đi lặp lại nhưng chưa rõ là bao nhiêu lần. Khi mà điều kiện Mẹ chưa cho nghỉ thì công việc rút rơm còn lặp lại. 🡪 Lặp với số lần chưa biết.

***\* Cú pháp:***

*While (<Điều kiện>)*

*{*

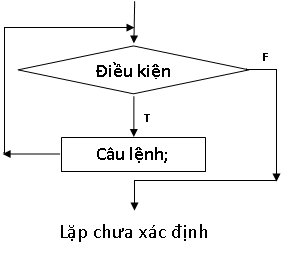
*<Câu lệnh>;*

*}*

***\* Giải thích:***

* *Điều kiện: Là biểu thức Logic.*
* *Câu lệnh: là một lệnh đơn hoặc ghép.*

***\* Hoạt động:***

**

*- Khi mà <điều kiện> còn đúng thì câu lệnh lặp còn lặp.*

***\* Lưu ý:*** *Trong câu lệnh lặp cần có thao tác làm thay đổi biểu thức điều kiện để lệnh lặp kết thúc*

**\* Infinite loops**

Vòng lặp vô tận xảy ra trong trường hợp không có sự tác động đến biểu thức điều kiện của vòng lặp while và nó luôn luôn đúng. Ví dụ:

int dem = 0;

while (dem < 10){

cout << dem << " ";

}

Biến dem trong trường hợp này không bị ai tác động đến giá trị, nên nó vẫn là 0 và luôn bé hơn 10. Dó đó, điều kiện luôn luôn đúng và vòng lặp không thể kết thúc được.

Chúng ta có thể cố ý khai báo vòng lặp vô hạn bằng cách sau:

while (true)

{

//This loop will be executed forever

}

Cách duy nhất để thoát ra khỏi vòng lặp vô tận là sử dụng từ khóa break, return, goto

\* **Do – while**

do

{

Câu lệnh;

}

while (<Điều kiện>);

while là vòng lặp kiểm tra điều kiện trước và sẽ lặp ít nhất 0 lần.

do while là vòng lặp kiểm tra điều kiện sau và sẽ lặp ít nhất 1 lần.

Cả 2 đều giống nhau ở chỗ là khi điều kiện còn đúng thì còn làm

II. BÀI TẬP:

**Bài tập 4.1:**

*Lập trình tính S.*

cho đến khi nào

*- Ý tưởng:*

* *ban đầu S 🡨 0, i 🡨 1*
* *Khi 1/I > 0.001 thì lặp lại:*

*(Giải thích thêm:*

* *việc cộng 1/i vào tổng chưa biết bao nhiêu lần* 🡪 *dùng lặp While*
* *Điều kiện còn lặp lại là Và khi thì dừng lặp*
* *Sau mỗi lần lặp thì ta tăng i lên 1. Ví dụ từ 1/1 ta tăng lên 1 để sang ½ …*

*- Thuật toán: Tự xây dựng dựa trên ý tưởng.*

*\* Chương trình:*

#include <bits/stdc++.h>

using namespace std;

double s, i;

int main(){

s = 0; i = 1;

while(1.0/i > 0.001){

s += 1.0/i;

i++;

}

cout <<s;

return 0;

}

**Bài 4.2:** (**TONGE.CPP**) **Nhập vào số thực N (0 < N < 1.5). Tính và đưa giá trị E ra màn hình. Kết quả làm tròn 9 chữ số thập phân.**  cho đến khi E > N.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1.376 | 1.423611111 |

**Bài 4.3:** (**SLE.CPP**) Lập trình tính S nhỏ nhất thỏa mãn S ≥ N, với 1 < N < 109.



|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 7 | 9 |
| 15 | 16 |

**Bài 4.4:** (**TKNH.CPP**) *Tiết kiệm ngân hàng*

Bạn có **a** đ gửi ngân hàng với lãi suất 1.1% mỗi tháng *(Lãi suất cộng dồn vào vốn gốc)*.

Lập trình tìm và thông báo số tháng cần gửi nếu bạn muốn nhận được số tiền ít nhất là x.

Biết rằng a và x là các số nguyên chưa đến 1 tỉ đồng và nhập vào từ bàn phím.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 2000000 2500000 | 21 |

**Bài 4.5:** (**GTKEP.CPP**) Viết chương trình tính n!! (giai thừa kép) với n!! được định nghĩa như sau:

- n!! = 1 với n = 0

- n!! = 1.3.5..n với n lẻ.

- n!! = 2.4.6..n với n chẵn.

Ràng buộc: 0 < N < 20

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 3 | 3 |
| 6 | 48 |

**Bài 4.6:** (**UCLN.CPP**) Viết chương trình tìm ước chung lớn nhất (UCLN) của hai số nguyên a và b được nhập vào từ bàn phím, với yêu cầu sử dụng thuật toán Euclid.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 3 3 | 3 |
| 3 2 | 1 |

**Bài 4.7:** (**BCNN.CPP**) Viết chương trình tìm bội số chung nhỏ nhất của hai số nguyên dương a và b nhập vào từ bàn phím. (0 < a, b < 1010)

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 3 2 | 6 |
| 2 4 | 4 |

**Bài 4.8**: (**DEMCHAN.CPP**) Viết chương trình nhập vào từ bàn phím 2 số nguyên a và b . Đếm số lượng các số chẵn chia hết cho 3 trong khoảng (a,b) và in kết quả ra màn hình.

**Bài 4.9:** (**BOCON.CPP**) Tí sống ở Sao Hoả, Tí có thể thọ tỷ Nam Sơn. Tí có một người Bố thật tuyệt vời, Tí muốn lớn thật nhanh để có thể làm được những việc như bố. Em hãy lập trình nhập vào số tuổi của Tí và bố hiện tại, tính và thông báo ra màn hình liệu sau bao nhiêu năm nữa tuổi bố sẽ gấp đôi tuổi Tí.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 30 5 | 20 |
| 100 65 | KHONG BAO GIO GAP DOI |

**Bài 4.10:** (**SODEP.CPP**) Số tự nhiên N được gọi là số đẹp nếu cộng các chữ số của N lại ta có một số mà kết thúc bằng 9. Ví dụ một số số đẹp là 18 (1+8=9), 234 (2+3+4=9), 658 (6+5+8=19). Cho một số N, hãy kiểm tra xem N có phải là số đẹp hay không.

**Dữ liệu nhập:** Là số nguyên N (0 ≤ n ≤ 109)

**Dữ liệu xuất:**

- Nếu N là số đẹp, in ra "YES", nếu không in ra "NO"

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 27 | YES |
| 26 | NO |

**Bài 4.11:** (**KHUMA.CPP**) Nhân dịp năm mới, hãng bia có chương trình khuyến mãi cho khách hàng. Nội dung của chương trình là nếu khách hàng đem đến cửa hàng 10 vỏ chai bia thì sẽ được nhận 3 chai bia khuyến mãi. Vậy nếu ban đầu anh Bo mua **n** chai bia thì tổng cộng anh có thể uống được tất cả bao nhiêu chai?

**Dữ liệu nhập:** Là số nguyên n (1 ≤ n ≤ 3.000) xác định số chai bia anh Bo mua ban đầu.

**Dữ liệu xuất:** Là số nguyên thể hiện tổng cộng số chai bia anh Bo có thể uống.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 10 | 13 |
| 24 | 33 |

**Bài 4.12:** (**KEBAN.CPP**) Thấy chân bàn bị gập ghềnh nên Nam lấy một tờ giấy lần lượt gấp đôi lại nhiều lần để kê chân bàn. Giả sử tờ giấy có bề dày là **a** thì sau lần gấp đôi thứ nhất bề dày là **2a**, sau lần gấp đôi thứ 2 là **4a**, lần thứ 3 là **8a**,.... Nếu khoảng gập ghềnh là b thì Nam cần gấp đôi giấy bao nhiêu lần để kê chân bàn ít bị gập ghềnh nhất?

Dữ liệu nhập: Là hai số nguyên a, b cách nhau một khoảng trắng (1 ≤ a ≤ b ≤ 109)

Dữ liệu xuất: Là số nguyên xác định số lần gấp giấy.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 1 4 | 2 |
| 2 5 | 1 |

*Chú ý: Vì bàn khá nặng không khiêng được nên bề dày giấy gấp không được lớn hơn khoảng gập ghềnh.*

**Bài 4.13:** (**DABI.CPP**) Một số nguyên dương n được gọi là số đặc biệt nếu n chia hết cho tổng các chữ số của n. Ví dụ, số 27 là số đặc biệt, còn hai số 11 và 2013 thì không phải là số đặc biệt.

Yêu cầu: Cho số nguyên dương n. Hãy kiểm tra xem số n có phải là số đặc biệt hay không?

Dữ liệu nhập: Là số nguyên dương n (1 ≤ n ≤ 1018)

Dữ liệu xuất: Nếu n là số đặc biệt in ra 1, nếu không phải in ra 0.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 27 | 1 |

**Bài 4.14:** (**NGAHA2.CPP**) Anh Bo có một số tiền là n đồng. Anh định đem n đồng này gửi ngân hàng với lãi suất 10% một năm. Cứ sau mỗi năm anh sẽ ra ngân hàng rút tiền lãi rồi nhập chung với tiền gốc để gửi lại cho năm sau (nếu tiền lãi là số thập phân lẻ thì sẽ được làm tròn đến hàng đơn vị, nghĩa là từ 0.5 trở lên thì làm tròn lên 1; dưới 0.5 thì làm tròn về 0). Anh dự định khi nào đủ số tiền m đồng thì sẽ rút toàn bộ để đi mua bò. Hỏi sau mấy năm thì anh Bo sẽ có đủ tiền mua bò?

Dữ liệu nhập: Là hai số nguyên n và m (10 ≤ n < m ≤ 109)

Dữ liệu xuất: - Là số năm cần gửi ngân hàng.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 100 120 | 2 |

**CHỦ ĐỀ 5**

MẢNG MỘT CHIỀU

I. LÝ THUYẾT

- **VD1**: Viết chương trình nhập vào nhiệt độ của một năm 365 ngày. Tính và đưa ra màn hình nhiệt độ trung bình của năm.

*Tổ chức dữ liệu:* ***Tb***là nhiệt độ trung bình, T1, T2 … T365 (Gồm 365 biến, mỗi biến lưu thông tin là nhiệt độ của một ngày)

*Với cách làm* như trên thì ta thấy có một số hạn chế: Số lượng biến khai báo quá lớn, đoạn chương trình nhập và tính nhiệt độ trung bình sẽ khá dài

* Xuất hiện nhu cầu cần một KDL mới để có thể lưu trữ cho 365 biến này 🡪 **Dùng mảng một chiều.**

**1. Khái niệm:**

*Mảng một chiều là một dãy hữu hạn các phần tử cùng kiểu. Mảng được đặt tên và một phần tử có một chỉ số.*

*Để mô tả mảng cần xác định:*

* *Kiểu của các phần tử*
* *Cách đánh số phần tử*

\* Ví dụ: Ta có thể dùng một mảng **TB** để lưu trữ dãy 365 nhiệt độ của năm.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TB** | 37.5 | 39 | 32 | 41.5 | .. | .. | 40 | 42 |
|  | 1 | 2 | 3 | 4 | … | | 364 | 365 |

Ta thấy nếu dùng mảng một chiều để lưu trữ thì chỉ cần dùng một biến kiểu mảng. Biến TB này gồm 365 phần tử, trong đó:

* Có cùng KDL là số thực
* Được đánh số từ 1 đến 365.

**Để truy cập hay tham chiếu đến một phần tử bất kì trong mảng TB ta có cú pháp:**

*TB[<Chỉ số>]*

Ví dụ: Để xem nhiệt độ của ngày thứ 70 ta viết: ***TB[70]***

**2. Khai báo mảng một chiều:**

***a) Cú pháp:***

*- Khai báo trực tiếp:*

<Kiểu dữ liệu> <Tên biến>[số lượng phần tử];

*- Khai báo gián tiếp:*

Typedef <Kiểu dữ liệu> <Tên kiểu>[số lượng phần tử];

<tên kiểu> <tên biến>;

***b) Ví dụ:***

*- Ví dụ 1: Khai báo mảng nhiệt độ 365 ngày.*

*Cách 1*

double T[366];

*Cách 2:*

typedef double mangnhietdo[366];

mangnhietdo T;

Cách khai báo gián tiếp rất hữu ích trong trường hợp truyền tham số mảng một chiều cho các hàm, thủ tục.

- Ví dụ 2:

int dx[]={-1,0,1,0};

Khai báo này mảng dx có 4 phần tử được gán giá trị lần lượt là -1, 0, 1, 0.

\*Lưu ý: Trong C++, các phần tử của mảng được đánh số từ 0.

### \* Nhập dữ liệu cho mảng một chiều (Array input)

Giả sử chúng ta có mảng một chiều dùng để chứa 10 số nguyên (có chỉ số từ 0 đến 9). Để nhập dữ liệu cho từng phần tử trong mảng này, chúng ta có thể sử dụng đối tượng **cin** trong thư viện **iostream** đã được học.

int T[10];

for(int i = 0; i <= 9; i++) {

cin >> T[i];

}

**\* Khởi tạo giá trị cho mảng 1 chiều**

Cách 1: Khởi tạo giá trị cho mọi phần tử của mảng

int array[4] = { 5, 8, 2, 7 };

Cách 2: Khởi tạo giá trị cho 2 phần tử đầu mảng

int array[4] = { 5, 8 };

Cách 3: Khởi tạo giá trị 0 cho mọi phần tử của mảng

int array[4] = { };

Cách 4: Tự động xác định số lượng phần tử

int array[] = { 5, 8, 2, 7 };

3. Kiểu Vector:

Nếu như số lượng phần tử (size) của một mảng là cố định, thì ở vector, nó hoàn toàn có thể thay đổi trong suốt quá trình làm việc của chương trình. Vector hoạt động như một “mảng động”.

So với mảng, cách dùng của vector thì linh hoạt hơn nhiều, và đây là 1 số điểm nổi trội của vector so với mảng array :

* Bạn không cần phải khai báo kích thước của vector
* Nếu thêm 1 phần tử vào vector đã đầy rồi, thì vector sẽ tự động tăng kíck thước của nó lên để dành chỗ cho giá trị mới này.
* Dùng số phần tử âm vẫn được trong vector ví dụ A[-10], A[-3], rất tiện trong việc cài đặt các giải thuật khác.

*3.1 Khai báo:*

vector<int> A ;

Câu lệnh trên định nghĩa 1 vector có kiểu int. Chú ý kiểu của vector được để trong 2 cái ngoặc nhọn.

Vì kích thước của vector có thể nâng lên, cho nên không cần khai báo cho nó có bao nhiêu phần tử cũng được, hoặc nếu thích khai báo thì bạn cũng có thể khai báo một trong những cách như sau :

vector<int> A(10);

vector<int> A(10, 2);

Đồng thời ta cũng có thể khởi tạo cho 1 vector bằng giá trị của 1 vector khác, ví dụ :

vector<int> A(10,2);

vector<int> B(A);

Với dòng lệnh trên thì vector B sẽ là bản sao của vector A.

*3.2 Thao tác với vector*

Để thêm một phần tử vào vị trí sau cùng của vector:

**A**.push\_back(**x**);

Để bỏ đi phần tử cuối cùng của vector:

**A**.pop\_back();

Để bỏ đi tất cả các phần tử của vector:

**A**.clear();

Để lấy ra phần tử vị trí thứ 3 của vector (đếm từ 0):

**A**[2]

Để biết số lượng phần tử của vector:

**A**.size();

Để chèn một phần tử x vào vị trí t trong vector:

**A**.insert(t, x);

Để thay đổi kích thước vector:

**A**.resize(5,1); //a gồm 5 phần tử có giá trị là 1

II. BÀI TẬP

**Bài 5.1:** Ta có 5 số thực và 25 số nguyên. Bạn Nam nói ta chỉ cần xây dựng một mảng một chiều N gồm có 30 phần tử để lưu trữ 30 số này. Em nghĩ gì về ý kiến của bạn Nam?

**Bài 5.2:** Em hãy khai báo biến mảng trong các trường hợp sau bằng cả hai cách:

a) Để lưu trữ 300 số thực.

b) Để lưu trữ 145 số nguyên được đánh số từ 0 đến 144

**Bài 5.3:** Lập trình nhập vào từ bàn phím dãy số nguyên n phần tử A1, A2 … AN. Đếm xem trong dãy có bao nhiêu phần tử là số dương và thông báo ra màn hình. (0 < n < 100; |Ai| < 10^9)

*Thuật toán:*

* + *B1: Nhập n và dãy số A.*
  + *B2: Dem :=0*
  + *B3: Cho i đi từ 1 đến N: Nếu A[i] là số dương thì tăng đếm.*
  + *B4: Thông báo kết quả Dem và kết thúc.*

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 5  1 2 -4 -3 6 | 3 |

**Bài 5.4:** (**MANG54.CPP**) Lập trình nhập vào một số nguyên dương N và dãy số nguyên A gồm N phần tử. Tìm phần tử lớn nhất của dãy số nguyên cùng vị trí của nó. (0 < n < 10000; |Ai| < 10^9). Nếu có nhiều phần tử cùng có giá trị nhỏ nhất thì đáp án là phần tử có vị trí bé nhất.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 5  1 12 -4 -3 12 | 12 2 |

**Bài 5.5:** (**MANG55.CPP**) Cho dãy n số nguyên A1, A2, … An (3 < n < 10000). Em hãy lập trình nhập vào dãy số và đếm số lượng các phần tử số chẵn có trong dãy số và thông báo ra màn hình?

**Bài 5.6:** (**MANG56.CPP**) Hãy lập trình nhập vào từ bàn phím 20 số nguyên không âm, tính và thông báo ra màn hình tổng các phần tử là số chẵn ở vị trí lẻ trong dãy 20 số đó.

**Bài 5.7:** (**MANG57.CPP**) Cho một số nguyên N và dãy số nguyên gồm N phần tử.

* Lập trình nhập vào N và dãy số. ( 1 < N < 200)
* Tính và thông báo ra màn hình số lượng các phần tử là số dương và chia hết cho 5.

**Bài 5.8:** (**MANG58.CPP**) Cho dãy số nguyên A gồm M phần tử. Hãy lập trình nhập vào dãy số A, tìm xem có hay không phần tử có giá trị bằng K, nếu có thông báo ra vị trí bé nhất của phần tử đó. Với K được nhập vào từ bàn phím.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 5 2  3 5 2 8 2 | CO K O VI TRI 3 |
| 5 4  3 5 2 8 2 | KHONG CO K |

**Bài 5.9:** (**MANG59.CPP**) Lập trình nhập vào N và dãy số A gồm N phần tử (5 < N < 1000). Bằng phương pháp nổi bọt em hãy sắp xếp dãy A thành dãy số không giảm và thông báo dãy A ra màn hình sau khi đã sắp xếp.

**Bài 5.10:** (**MANG510.CPP**) Lập trình nhập vào số nguyên n và dãy số nguyên dương B gồm n phần tử. Tìm và thông báo ra tất cả các vị trí cùng giá trị của các phần tử là số nguyên tố. (1 < N < 1000; 0 < Bi < 107)

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 5  1 2 3 5 6 | 2 2  3 3  4 5 |
| 5  4 6 8 10 12 | KHONG CO SO NGUYEN TO |

**Bài 5.11:** (**MANG511.CPP**) Xét dãy số thực B1, B2, … Bn (3 < n < 35). Số Bi (1≤i≤N) được gọi là *Cực tiểu địa phương* nếu thỏa mãn:

* + Nếu i = 1 thì B1 < B2
  + Nếu i = N thì Bn < Bn-1
  + Còn lại thì Bi < Bi-1 và Bi < Bi+1

Lập trình nhập vào dãy số B và thông báo ra màn hình số lượng các *cực tiểu địa phương* trong dãy B.

**Bài 5.12:** (**MANG512.CPP**) Xét dãy số thực A1, A2, … An (3 < n < 35). Số Ai (1≤i≤N) được gọi là *Cực đại địa phương* nếu thỏa mãn:

* + Nếu I = 1 thì A1 > A2
  + Nếu I = N thì An > An-1
  + Còn lại thì Ai-1 < Ai > Ai+1

Lập trình nhập vào dãy số A và thông báo ra màn hình các *cực đại địa phương* cùng chỉ số của nó.

**Bài 5.13:** (**MANG513.CPP**) Cho hai dãy số A1, A2, …An và B1, B2, …Bn. (0 < n < 1000) Dãy Cđược xác định như sau:



Hãy lập trình nhập vào số nguyên n và hai dãy số nguyên A, B. Tạo và thông báo ra màn hình dãy số nguyên C.

**Bài 5.14:** (**MANG514.CPP**) Cho dãy số nguyên P1, P2, … Pm,  Biết rằng không phải mọi phần tử của P đều bằng nhau. Hãy lập trình nhập vào dãy số nguyên P sau đó đổi chổ phần tử có giá trị lớn nhất cho phần tử có giá trị nhỏ nhất.

**Bài 5.15:** (**TRIAN.CPP**) Một cửa hàng kinh doanh thiết bị điện tử tổ chức tuần lễ tri ân khách hàng với thể lệ như sau: Trong tuần lễ tri ân, mỗi khách hàng đến mua hàng sẽ được bốc thăm một con số may mắn trong khoảng từ 1 đến 5000, chương trình sẽ dừng lại nếu hết thời gian hoặc các số đã được bốc hết. Cửa hàng sẽ lưu lại số may mắn của khách hàng theo thứ tự mua hàng.

Kết thúc chương trình cửa hàng tổ chức quay số ngẫu nhiên để tìm ra số may mắn trúng thưởng, chắc chắn rằng số trúng thưởng nằm trong những số khách hàng đã bốc thăm.

Em hãy lập trình để nhập vào danh sách các số bốc thăm của khách hàng, số trúng thưởng mà cửa hàng quay số được và thông báo ra màn hình số thứ tự của khách hàng trúng thưởng.

**Dữ liệu nhập:**

* số nguyên n (1 ≤ n ≤ 5000) – số khách hàng.
* n số nguyên a1, a2, …, an, trong đó ai (1 ≤ ai ≤ 5.10^3) – số may mắn của khách hàng thứ i.
* k – số may mắn trúng thưởng mà cửa hàng công bố.

**Kết quả:** số nguyên i – số thứ tự của khách hàng trúng thưởng.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 4  4 2 3 8  8 | 4 |

**Bài 5.16:** (**IQCHECK.CPP**) Tèo đang chuẩn bị vượt qua bài kiểm tra IQ. Nhiệm vụ thường xuyên nhất trong bài kiểm tra này là để tìm ra một trong những số n cho trước khác với các số khác. Tèo quan sát thấy một số thường khác với số khác về tính chẵn lẻ.

Em hãy giúp Tèo viết chương trình - để kiểm tra câu trả lời của mình.

Chương trình sẽ nhập vào n ( 3 ≤  n  ≤ 100 ) và dãy số nguyên A1, A2 … An. Dữ liệu đảm bảo chỉ có một số khác loại với các số còn lại.

Kết quả: in ra màn hình số khác loại đó.

**Bài 5.17:** (**THUCUNG.CPP**) Ở Quốc gia thú cưng Berland đang là kì nghỉ bình đẳng. Để vinh danh ngày lễ, nhà vua đã quyết định cân bằng phúc lợi cho mọi công dân bằng chi phí của kho bạc nhà nước. Toàn bộ ở Berland có n công dân, phúc lợi của mỗi người trong số họ được ước tính là một số nguyên, đơn vị là burle (burle là tiền tệ ở Berland). Bạn là thủ quỹ của hoàng gia, chịu trách nhiệm đếm những khoản chi phí của vương quốc.

**Dữ liệu nhập:**

* số nguyên n (1 ≤ n ≤ 100) – số công dân trong vương quốc Berland.
* n số nguyên a1, a2, …, an, trong đó ai (0 ≤ ai ≤ 106) – phúc lợi của công dân thứ i.

**Kết quả:** số nguyên S – số lượng burle tối thiểu cần chi trả cho toàn bộ công dân, sao cho mỗi công dân đều có được phúc lợi như nhau.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 4  0 2 4 3 | 7 |

**Bài 5.18:** (**XEBUYT.CPP**) Sắp đến tỉnh quyết định đầu tư tuyến xe điện từ biển Triệu Lăng lên Đông Hà. Tuyến có n điểm dừng, được đánh số từ 1 đến n theo thứ tự chuyển động của xe điện. Theo khảo sát tại điểm dừng thứ i sẽ có a[i] khách xuống xe và b[i] khách lên xe. Xe điện trống trước khi nó đến điểm dừng chân đầu tiên. Ngoài ra, khi xe điện đến điểm dừng cuối cùng, tất cả hành khách sẽ xuống để nó trở nên trống rỗng.

Nhiệm vụ của bạn là tính toán số ghế tối thiểu của xe điện sao cho bất cứ lúc nào mỗi khách hàng đề có ghế riêng để ngồi. Lưu ý rằng tại mỗi điểm dừng, tất cả hành khách xuống sẽ xuống trước khi bất kỳ hành khách nào vào xe điện.

**Dữ liệu nhập:**

* Nhập n ( 2 ≤  n  ≤ 1000 ) - số điểm dừng của xe điện.
* Sau đó n lần tiếp theo, mỗi lần nhập hai số nguyên ai và bi ( 0 ≤  ai ,  bi  ≤ 1000.000) - số lượng hành khách lên và xuống tại điểm dừng i. Các điểm dừng được đưa từ điểm dừng đầu tiên đến điểm dừng cuối cùng theo thứ tự chuyển động của xe điện.

**Kết quả:** in một số nguyên biểu thị số ghế tối thiểu cần có.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 4  0 3  2 5  4 2  4 0 | 6 |

**Bài 5.19:** (**TEOTI.CPP**) Tí và Tèo cùng chơi một trò chơi. Ban đầu có n số nguyên A1, A2, …, An được viết trên bảng. Mỗi lượt một người chơi chọn một số và xóa nó khỏi bảng. Điều này tiếp tục cho đến khi chỉ có một số còn lại trên bảng, Tèo thực hiện di chuyển đầu tiên, sau đó người chơi thay phiên nhau.

Tèo muốn giảm thiểu số cuối cùng còn lại trên bảng, trong khi Tí muốn tối đa hóa nó.

Bạn muốn biết số nào sẽ được để lại trên bảng nếu cả hai người chơi đều chơi tối ưu.

**Dữ liệu vào:**

* + Dòng đầu tiên chứa một số nguyên n ( 1 ≤ n ≤ 1000) - số lượng các con số trên bảng.
  + A1, A2, …, An ( 1 ≤ Ai ≤ 106)

**Đầu ra:** in một số sẽ được để lại trên bảng.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 3  2 1 3 | 2 |

**CHỦ ĐỀ 6**

MẢNG HAI CHIỀU

I. LÝ THUYẾT

Mảng hai chiều là một **bảng** các phần tử có cùng kiểu. Ta có thể tưởng tượng nó gồm chiều ngang và chiều dọc. Nếu theo chiều ngang thì bảng có m dòng: Mỗi dòng như vậy có n phần tử tương ứng với n cột.

Ví dụ: Bảng **A** gồm 3 dòng và 4 cột.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 |
| 1 |  |  |  |  |
| 2 |  |  | 25 |  |
| 3 |  |  |  |  |

Mảng A gồm 3 dòng được đánh số từ 1 đến 3 và 4 cột từ 1 đến 4.

Bây giờ muốn lấy giá trị của 25. Tức là truy xuất đến phần tử dòng 2, cột 3 ta viết như sau: **A[2][3]**

Hay muốn truy xuất đến một ô dòng **i** cột **j** ta viết:

**A[i][j]**

1. **Khai báo mảng hai chiều:**

Có hai cách khai báo:

*- Khai báo trực tiếp:*

<Kiểu dữ liệu> <Tên biến> [số hàng][số cột];

*- Khai báo gián tiếp:*

Typedef <Kiểu dữ liệu> <Tên kiểu> [số hàng][số cột];

<tên kiểu> <tên biến>;

- Ví dụ 1: Khai báo mảng hai chiều gồm 5 dòng 4 cột.

Cách 1:

int a[5][4];

Cách 2:

**typedef** int mang2chieu[5][4];

mang2chieu a;

1. **Nhập mảng hai chiều**

**-** Ví dụ: Nhập mảng hai chiểu gồm m dòng n cột.

cin >>m >>n;

for(int i=1; i<=m; i++)

for (int j=1; j<=n; j++) cin >>a[i][j];

*Với câu lệnh như trên thì ta sẽ lần lượt nhập vào:*

*A[1][1], A[1][2], A[1][3]…A[1][n]*

*A[2][1], A[2][2], A[2][3]…A[2][n]*

*…*

1. **In mảng hai chiều**

for(int i=1; i<=m; i++){

for (int j=1; j<=n; j++) cout <<a[i][j] <<” “;

cout <<endl;

}

**\* Khởi tạo mảng hai chiều**

Mình lấy lại ví dụ về mảng có tên A như trên, mình sẽ khởi tạo giá trị cho mảng như sau:

int A[3][5] =

{

{ 1, 2, 3, 4, 5 }, //row 1

{ 6, 7, 8, 9, 10 }, //row 2

{ 11, 12, 13, 14, 15 } //row 3

};

Do mảng A có 3 hàng, mỗi hàng lại là một mảng một chiều khác nhau, nên mình đã sử dụng cách khởi tạo của mảng một chiều, áp dụng cho mỗi hàng trong mảng hai chiều A.

Các bạn có thể khởi tạo mảng hai chiều theo cách sau:

int A[3][5] = {

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

};

Nhưng khuyến khích các bạn sử dụng cách mình trình bày ở trước để tránh nhầm lẫn trong việc tổ chức dữ liệu.

Những phần tử chưa được khởi tạo giá trị sẽ được gán bằng giá trị mặc định tùy vào mỗi kiểu dữ liệu khác nhau. Như ví dụ sau mình sử dụng kiểu int để khai báo mảng hai chiều:

int seats[3][5] = {

{ 1, 2 }, //row 1 = 1, 2, 0, 0, 0

{ 6, 7, 8 }, //row 2 = 6, 7, 8, 0, 0

{ 11 }, //row 3 = 11, 0, 0, 0, 0

};

Tương tự mảng một chiều, nếu các bạn khởi tạo mảng hai chiều ngay khi khai báo, compiler có thể tự xác định số hàng cần cấp phát:

int A[][4] =

{

{ 1, 2, 3, 4 },

{ 5, 6, 7, 8 }

};

Có thể bỏ trống phần khai báo số lượng hàng, nhưng không thể không khai báo số lượng cột.

II. BÀI TẬP

**Bài 6.1:** (**MANG61.CPP**) Viết chương trình nhập vào ma trận số nguyên gồm n dòng m cột. đếm số lượng các số chia hết cho 3 có trong ma trận đó. (1 < n, m < 20)

\* Ý tưởng – Thuật toán: Nhập ma trận vào, duyệt qua các phần tử của mảng: Nếu phần tử nào chia hết cho 3 thì đếm.

B1: Nhập mảng A

B2: Dem :=0

B3: Cho i đi từ 1 đến n

Cho j đi từ 1 đến m: Nếu a[i,j] mod 3 = 0 thì tăng đếm.

B4: Thông báo đếm và kết thúc.

\* Chương trình:

**Bài 6.2:** (**MANG62.CPP**) Cho mảng hai chiều kích thước NxN (N dòng và N cột). Lập trình nhập vào mảng này, tìm và thông báo ra màn hình các số lớn nhất trên mỗi hàng. (1 < N < 15)

**Bài 6.3:** (**MANG63.CPP**) Cho ma trận số nguyên dương kích thước NxN. Lập trình nhập vào mảng này, tính tổng số chẵn trên mỗi hàng và thông báo ra màn hình các tổng đó mỗi tổng một hàng. (1 < N < 111)

**Bài 6.4:** (**MANG64.CPP**) Lập trình tạo ma trận hai chiều kích thước MxM theo quy luật như sau: Ví dụ M=4

2 3 4 5

3 4 5 6

4 5 6 7

5 6 7 8

Nhập vào từ bàn phím số nguyên M và thông báo ra màn hình ma trận vừa tạo theo quy luật trên.

**Bài 6.5:** (**MANG65.CPP**) Lập trình nhập vào từ bàn phím ma trận vuông số nguyên kích thước NxN. Tính tổng các phần tử trên đường chéo chính và thông báo ra màn hình. (1 < N < 100)

**Bài 6.6:** (**MANG66.CPP**) Lập trình nhập vào mảng hai chiều gồm m dòng n cột chỉ gồm các số nguyên. Tính tổng các phần tử trên mỗi cột và thông báo các tổng này ra màn hình. (1 < m, n < 100)

**Bài 6.7:** (**MANG67.CPP**) Lập trình nhập vào mảng hai chiều gồm m dòng n cột. Sắp xếp các giá trị trong mảng tăng dần theo từng hàng. In mảng sau khi sắp xếp ra màn hình.

Ví dụ: Nhập m=3, n=4

|  |  |
| --- | --- |
| Nhập vào | Kết quả |
| 3 4  4 5 2 1  7 3 5 6  9 8 6 4 | 1 2 4 5  3 5 6 7  4 6 8 9 |

**Bài 6.8:** (**MANG68.CPP**) Lập trình nhập vào mảng hai chiều gồm m dòng n cột. Sắp xếp các giá trị trong mảng tăng dần từ trái sang phải từ trên xuống dưới. In mảng sau khi sắp xếp ra màn hình.

Ví dụ: Nhập m=3, n=4

|  |  |
| --- | --- |
| Nhập vào | Kết quả |
| 3 4  4 5 2 1  7 3 5 6  9 8 6 4 | 1 2 3 4  4 5 5 6  6 7 8 9 |

**CHỦ ĐỀ 7**

KIỂU DỮ LIỆU XÂU

I. LÝ THUYẾT

Thư viện chuẩn C/C++ cung cấp một kiểu lớp **String** mà hỗ trợ tất cả hoạt động liên quan tới chuỗi đã đề cập ở trên, và bổ sung thêm nhiều tính năng nữa. Chúng ta sẽ học lớp này trong Thư viện chuẩn C/C++ (C++ Standard Library)

**1. Khái niệm:**

**-** Xâu là một dãy kí tự trong bảng mã ASCII. Mỗi kí tự được gọi là một phần tử của xâu.

- Số lượng kí tự trong xâu được gọi là độ dài xâu.

- Xâu có độ dài bằng 0 gọi là ***xâu rỗng.***

***(Lưu ý: Ta có thể xem xâu như là mảng một chiều mà mỗi phần tử kiểu kí tự)***

- Tham chiếu đến một phần tử của xâu: **S[Chỉ số]**

Ví dụ: Với xâu S = “Le\_Thanh\_Huy”. Ta có: S[3] là ‘T’, S[0] = ‘L’

**\* Khai báo:**

string <tên biến xâu>;

Ví dụ:

string s;

Các kí tự trong xâu được đánh chỉ số bắt đầu từ **0**.

**\* Các thao tác làm việc với biến xâu:**

**\* So sánh hai xâu:**

Quy tắc:

- Hai xâu bằng nhau nếu chúng giống nhau hoàn toàn: (Phân biệt hoa thường)

Ví dụ: “Hona” = “Hona” Nhưng “hona” ≠ “HONA”

- Hai xâu khác nhau: Ta sẽ kiểm tra kí tự khác nhau đầu tiên của hai xâu tính từ bên trái sang, nếu kí tự nào có mã ASCII lớn hơn thì xâu đó lớn hơn

Ví dụ: “Anh” < “Amh” Giải thích: Kí tự khác nhau đầu tiên tính từ bên trái sang là n và m. Mà m có mã ASCII lớn hơn nên xâu “Amh” lớn hơn.

Chú ý: - ‘A’ có mã ASCII là 65, ‘B’ là 66…

- ‘a’ có mã ASCII là 97 …

**\* Ghép xâu:** Sử dụng toán tử += hoặc +

Ví dụ:

s = "Chao";

s +=" ban";

**\* Các hàm và thủ tục xử lý xâu:**

Các em xem đầy đủ ở đây <http://www.cplusplus.com/reference/string/string/>

- Lấy độ dài xâu: Hàm length() hoặc size()

s = “chao”;

cout <<s.size();

Kết quả: 4

- Xóa phần tử trong xâu: Hàm erase(vị trí, độ dài)

s = “chao ban”;

s.erase(0, 5);

Kết quả: s = “ban”;

- Chèn thêm vào xâu: Hàm insert

s = “Chao ban”;

s.insert(0, “Xin ”);

Kết quả: s = “Xin chao ban”;

S = “Chao ban”;

R = “ Huy”;

S.insert(8, R);

Kết quả: S = “Chao ban Huy”;

S = “Chao ban”;

R = “ Huy”;

S.insert(8, R, 2, 2);

Kết quả: S = “Chao ban uy”;

- Sao chép xâu: Hàm substr

S = “Chao ban”;

R = S.substr(1, 3);

Kết quả: R = “hao”

- **Tìm kiếm:** Hàm find sẽ trả về vị trí đầu tiên xuất hiện, nếu không tìm thấy sẽ trả về vị trí string::npos

s = "Chao ban";

if(s.find("ban") != string::npos)

cout <<"YES";

Đáp án: in ra YES

Tương tự có: rfind tìm kiếm vị trí xuất hiện cuối cùng

- **Thay thế:** Hàm replace(pos, len, str2); Thay thế len kí tự từ kí tự pos bằng str2

s = "Chao ban";

s.repalce(1,4,’hu’); //s = ‘Chuban’

- Chuyển từ xâu sang số:

Cách 1: stringstream(s) >>x;

Cách 2: x = atoi(s.c\_str()); (Tương tự có: atof, atol, atoll)

Cách 3: x = stoi(s); (Tương tự có: stof, stol, stoll, stoull)

y = stoll(s+r); (Đáp án y là tổng của hai xâu số)

- Chuyển từ số sang xâu: Số x

- Qua xâu cơ số 10

ostringstream ss;

ss <<x;

string r = ss.str();

- Qua xâu hệ cơ số 8

char r[100];

sprintf(r, "%o", x);

- Qua xâu hệ cơ số 16

char r[100];

sprintf(r, "%x", x);

**- in hoa một xâu:**

transform(s.begin(), s.end(), s.begin(), ::**toupper**);

**- in thường xâu s:**

transform(s.begin(), s.end(), s.begin(), ::**tolower**);

II. BÀI TẬP

**Bài 7.1:** (**SINHOA.CPP**) Viết chương trình nhập vào một xâu kí tự chữ cái, in ra màn hình xâu đó in hoa.

**Bài 7.2:** (**SDAONGUOC.CPP**) Viết chương trình nhập vào một xâu S và thông báo ra màn hình xâu đảo ngược.

Ví dụ: ABCD → DCBA

**Bài 7.3:** (**SDEMA.CPP**) Lập trình nhập vào một xâu kí tự, đếm và thông báo ra màn hình số lượng kí tự **A** có mặt trong xâu.

**Bài 7.4:** (**SDAIHON.CPP**) Lập trình nhập vào 2 xâu họ tên, so sánh xem xâu nào dài hơn và thông báo ra màn hình độ dài xâu đó.

|  |  |
| --- | --- |
| **Nhập vào** | **Kết quả** |
| Chao ban nhe  Chao ban | 12 |

**Bài 7.5:** (**SDEMKT.CPP**) Lập trình nhập vào một xâu bất kỳ gồm chữ cái và kí tự số. Đếm và thông báo ra màn hình số kí tự là số có mặt trong xâu.

**Bài 7.6:** (**SCHUANHOA.CPP**) *Chuẩn hóa xâu*

Một xâu được gọi là xâu chuẩn nếu: không có kí tự trắng dư thừa đầu và cuối xâu; không có hai kí tự trắng liền nhau.

Em hãy lập trình nhập vào một xâu và in ra màn hình xâu sau khi đã chuẩn hóa

**Bài 7.7:** (**S5CUOI.CPP**) Viết chương trình nhập vào một xâu và in ra màn hình 5 kí tự cuối cùng của xâu.

Ví dụ: Nhập “Hom nay la thu bay” kết quả: “u bay”

**Bài 7.8:** (**STENCHUAN.CPP**) *Chuẩn hóa họ tên*

Lập trình nhập vào từ bàn phím một xâu họ tên đã chuẩn hóa. Thông báo ra màn hình xâu đó sau khi đã in hoa các kí tự đầu tiên của từ.

Ví dụ: Nhập “le thanh huy” Kết quả: “Le Thanh Huy”

**Bài 7.9:** (**STACHTEN.CPP**) *Tách tên*

Lập trình nhập vào một xâu họ tên chuẩn và thông báo ra màn hình tên của xâu vừa nhập.

Ví dụ: “Le Thanh Huy” Kết quả: “Huy”

**Bài 7.10:** (**STUDAI.CPP**) *Từ dài nhất*

Trong một xâu thì từ dài nhất là từ có nhiều kí tự nhất. Ví dụ xâu “Le Thanh Huy” có 3 từ trong đó từ “Thanh” là từ dài nhất.

Lập trình nhập vào một xâu bất kì có nhiều từ, các từ là dãy các kí tự chữ cái liên tiếp và cách nhau một hoặc nhiều dấu cách. Tìm và thông báo ra màn hình từ dài nhất.

**Bài 7.11**: (**SDEMD.CPP**) *Đếm kí tự*

Cho xâu “BAN NA” thì ta đếm được có 2 kí tự A, 1 kí tự B và 2 kí tự N.

Em hãy lập trình nhập vào từ bàn phím một xâu kí tự hoa bất kì, đếm và thông báo ra màn hình số lần xuất hiện của các kí tự trong xâu.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| ABABBCH | A 2  B 3  C 1  H 1 |

**Bài 7.12:** (**SMASO.CPP**) Nam thường xuyên tham gia thi lập trình trên mạng. Vì đạt được thành tích cao nên Nam được gửi tặng một phần mềm diệt virus. Nhà sản xuất phần mềm cung cấp cho Nam một mã số là một số nguyên dương N có không quá 2555 chữ số. Để cài đặt được phần mềm, Nam phải nhập vào mật khẩu của phần mềm. Mật khẩu là một số nguyên dương M được tạo ra bằng cách tính tổng giá trị các chữ số của N.

Yêu cầu: Hãy tìm số nguyên dương M.

**Dữ liệu nhập:**

- Ghi số nguyên dương N (1 ≤ N < 102555).

**Dữ liệu xuất:**

- Ghi số nguyên dương M tìm được.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 84915388247 | 59 |

**Bài 7.13:** (**SBIGNUM.CPP**) *Số lớn - Big Number*

Với một số lớn thì các kiểu dữ liệu số được học không thể sử dụng được. Ví dụ 20……0 *(gồm 200 chữ số 0)* thì không thể khai báo bằng các KDL số. Khi đó ta có thể sử dụng kiểu xâu để lưu trữ số này. Tuy nhiên khi lưu trữ bằng kiểu xâu thì không thể thực hiện được các phép toán số học với các xâu số này.

Em hãy lập trình tính tổng của hai Big Number được nhập vào từ bàn phím và thông báo ra màn hình.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 55  30 | 85 |

**Bài 7.14:** (**SNENXAU.CPP**) *Nén xâu*

Để tiết kiệm bộ nhớ khi lưu trữ thông tin thì ta có thể sử dụng các kỹ thuật nén thông tin. Ví dụ: Cho xâu “aaabbbbcccccc” ta có thể nén thành “3a4b6c”. Em hãy lập trình:

*a) nhập vào một xâu chưa nén và hãy nén xâu đó.*

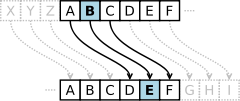
*b) Nhập vào một xâu đã nén và hãy giải nén xâu đ*ó.

Kết quả thông báo ra màn hình. Xâu có độ dài không quá 104.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| aaaaabbEEE  3a5C | 5a2b3E  aaaCCCCC |

**Bài 7.15:** (**SXEDA.CPP**) *Mã hóa xâu*

Để giữ bí mật người ta phải mã hóa các thông tin trước khi truyền đi hoặc lưu trữ. Một trong những cách mã hóa sớm nhất thời cổ đại đó là cách mã hóa do Xe Da (Caesar) đề xuất: Người ta thay mỗi chữ cái bằng chữ cái đứng sau nó K vị trí; việc tìm kiếm và thay thế được thực hiện theo vòng tròn chữ cái.

 Ví dụ: Với K = 3

Hãy lập trình nhập vào từ bàn phím số nguyên K và xâu kí tự S in hoa. Mã hóa theo quy tắc mã Xe Da và thông báo kết quả ra màn hình.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả thông báo** |
| 2  ABC | CDE |

**Bài 7.16:** (**SBIEUTHUC.CPP**) *Tính giá trị của xâu số.*

Một xâu S gồm các kí tự số và kí tự dấu + - Em hãy viết chương trình tính tổng của xâu S đó.

Ví dụ: S = “123+45-67+89” Tong = 190

**CHỦ ĐỀ 8**

KIỂU DỮ LIỆU TỆP

I. LÝ THUYẾT

1. **Vì sao cần dùng kiểu tệp:**

**-** Các KDL ta được học trước đây đều được lưu trữ trong Ram. Ta thấy khi chạy chương trình thì phải nhập dữ liệu vào và khi kết thúc chương trình thì không lưu trữ được dữ liệu 🡪 Cần một KDL gì đó để có thể lưu trữ thông tin để xữ lí lâu dài 🡪 KDL Tệp

- KDL tệp có đặc điểm:

* Dữ liệu lưu trữ lâu dài, không bị mất khi kết thúc chương trình hoặc mất điện.
* Dữ liệu lưu trữ trên mỗi tệp là **Rất lớn** và chỉ phụ thuộc vào dung lượng đĩa.

1. **Làm việc với tệp:**

Ta có thể sử dụng thư viện freopen chuẩn để gắn một tệp cho một stream

Ví dụ:

freopen("vidu.inp","r",stdin);

freopen("vidu.out","w",stdout);

Trong đó: "r" mở để đọc, "w" mở để ghi

II. BÀI TẬP

**Bài 8.1:** (**TEP81.CPP**) Cho tệp *Hinhchunhat.inp* lưu trữ thông tin hai cạnh của một hình chữ nhật. Em hãy đọc thông tin từ tệp này, tính và thông báo ra màn hình chu vi và diện tích của hình chữ nhật đó.

|  |  |
| --- | --- |
| *Hinhchunhat.inp* | Kết quả thông báo trên màn hình |
| 5 7 | Chi vi: 24  Dien tich: 35 |

**Chương trình**

#include <bits/stdc++.h>

using namespace std;

int a, b, s;

int main(){

freopen("Hinhchunhat.inp","r",stdin);

cin >>a >>b;

cout <<"Chu vi: " <<(a+b)\*2 <<"\n";

cout <<"Dien tich: " <<a\*b;

return 0;

}

*Nhận xét: in trực tiếp kết quả chu vi và diện tích 🡪 Tiết kiệm được 2 biến.*

**Bài 8.2:** (**TEPTH.CPP**) Cho tệp *TEPTH.inp* có hai số nguyên trên hai dòng. Em hãy lập trình đọc hai số nguyên từ tệp này, tính tổng, hiệu, tích và thương của hai số đó (Thương làm tròn 2 chữ số thập phân). Kết quả được lưu trữ vào tệp *TEPTH.out*

|  |  |
| --- | --- |
| *TEPTH.inp* | *TEPTH.out* |
| 9  3 | 9 + 3 = 12  9 – 3 = 6  9 \* 3 = 27  9 : 3 = 3.00 |

**Bài 8.3:** (**TEPUOC.CPP**) Cho tệp *TEPUOC.inp* có lưu trữ một số nguyên dương N. Hãy lập trình đọc số N từ tệp và tìm các ước số của N (1 < N < 109). Kết quả ước số được lưu vào tệp *TEPUOC.out*.

|  |  |
| --- | --- |
| TEPUOC*.inp* | TEPUOC*.out* |
| 15 | Cac uoc so cua 15  1  3  5  15 |

**Bài 8.4:** (**TEPSO.CPP**) Cho tệp *TEPSO.inp* gồm nhiều dòng, mỗi dòng là một dãy số nguyên gồm 3 số. Em hãy lập trình đọc thông tin từ tệp này, tính tổng của mỗi dãy số. Kết quả lưu vào tệp *TEPSO.out*.

|  |  |
| --- | --- |
| *TEPSO.inp* | *TEPSO.out* |
| 5 3 5  2 3 4  5 4 3  2 1 3 | 13  9  12  6 |

**Bài 8.5:** (**BANHANG.CPP**) Trong dịp tham dự hội chợ của Trường, lớp 11C đã mở gian hàng bán trái cây. Bạn Lan làm thủ quỷ đã ghi chép kết quả bán được vào tệp *Banhang.inp* trong máy tính của bạn. Mỗi dòng là một lần bán gồm: Số quả bán được và số tiền bán mỗi quả. Em hãy lập trình giúp bạn đọc dữ liệu từ tệp này và tính tổng thu nhập của lớp sau khi kết thúc hội chợ.

|  |  |
| --- | --- |
| *Banhang.inp* | *Banhang.out* |
| 5 2000  2 4000  4 15000  2 7000 | Tong thu: 92000 |

**Bài 8.6:** (**DSLOP.CPP**) Danh sách họ tên các bạn trong lớp đã được Lớp trưởng đánh vào tệp *Dslop.inp* (Tối đa 45 bạn). Tuy nhiên do bạn chưa thành thạo về gõ văn bản nên các xâu họ tên chưa được chuẩn hóa, vẫn còn kí tự trắng dư thừa.

Em hãy lập trình giúp bạn đọc dữ liệu từ tệp này, chuẩn hóa các xâu họ tên và lưu chúng vào tệp mới là *DSlop.out*

|  |  |
| --- | --- |
| *DSlop.inp* | *DSlop.out* |
| Le Huu Thang  Tran Van Nam  Nguyen Hoang Nam  … | Le Huu Thang  Tran Van Nam  Nguyen Hoang Nam  … |

**Bài 8.7:** (**LOP.CPP**) Danh sách lớp 11B được lưu trữ trong tệp *Lop.inp*. Trong một cuộc liên hoan của lớp cần bốc thăm lấy ra tên của người sẽ nhận được phần quà may mắn. Lớp trưởng chỉ muốn lấy ra danh sách tên của các thành viên trong lớp để bốc thăm.

Em hãy lập trình giúp cho lớp trưởng thực hiện điều đó. Danh sách tên lấy ra được lưu trữ vào tệp *Lop.Out*.

|  |  |
| --- | --- |
| *Lop.inp* | *Lop.out* |
| Nguyen Quang An  Tran Xuan Bac | An  Bac |

**Bài 8.8:** (**SOCHAN.CPP**) Có N số nguyên dương được lưu trữ trong tệp SOCHAN.inp, các số cách nhau một kí tự trống. Em hãy đọc danh sách N số nguyên dương này ra mảng một chiều. Đếm số lượng các số chẵn và ghi kết quả vào tệp SOCHAN.out.

Ví dụ: Tệp SOCHAN.inp gồm nhiều dòng.

- Dòng 1: Số N (1 < N < 105)

- Các dòng sau: là các số nguyên dương trong dãy N số

Kết quả ra: Ghi vào tệp SOCHAN.out gồm:

- Dòng 1: Số lượng số chẵn.

- Các dòng tiếp theo: liệt kê các số chẵn có trong dãy.

|  |  |
| --- | --- |
| **SOCHAN*.inp*** | **SOCHAN*.out*** |
| 5  3 5 7 8 4 | 2  8 4 |

**Bài 8.9:** (**MATRAN.CPP**) Cho ma trận số thực hai chiều A (m x n) được lưu trữ trong tệp Matran.inp. Em hãy tìm số lớn nhất trên mỗi hàng, lưu kết quả vào tệp matran.out.

Dữ liệu vào: Tệp Matran.inp

- Dòng 1: Hai số nguyên dương M và N (1 < M, N < 100)

- N dòng tiếp theo mỗi dòng là một hàng của ma trận.

Dữ liệu ra: Lưu kết quả lưu vào tệp matran.out

- Dòng i: là số lớn nhất của dòng i trong ma trận A (làm tròn 2 chữ số thập phân)

Ví dụ:

|  |  |
| --- | --- |
| ***matran.inp*** | ***matran.out*** |
| 3  1 2 3  5 3 7  2 8 6 | 3.00  7.00  8.00 |

**CHỦ ĐỀ 9**

CHƯƠNG TRÌNH CON

I. LÝ THUYẾT

1.1 Vì sao cần dùng chương trình con:

Chương trình giải các bài toán phức tạp thường rất dài 🡪 nên ta sẽ khó khăn khi đọc chương trình, tìm và hiệu chỉnh lỗi trong chương trình đó.🡪 Giải pháp: Chia chương trình lớn thành các nhiệm vụ nhỏ, mỗi nhiệm vụ này gọi là một chương trình con.

**🡪 Lưu ý: Chương trình con là một nội dung quan trọng trong lập trình.**

Một số ngôn ngữ lập trình chia ra hai loại chương trình con là Hàm và thủ tục. Tuy nhiên trong C++ chỉ có một loại duy nhất chung là hàm.

Trong C++ chỉ có hàm không trả về giá trị là **void**. Ví dụ: Hàm sau nhập giá trị cho mảng A

void nhap(){

cin >>n;

for(int i=1; i<=n; ++i) cin >>a[i];

}

Mẫu chung của định nghĩa hàm trong Ngôn ngữ C/C++ như sau:

**Kieu\_tra\_ve** Ten\_ham( Danh sach tham so )

{

<Thân hàm>;

}

Một định nghĩa hàm trong ngôn ngữ C/C++ bao gồm đầu hàm và một thân hàm. Dưới đây là các phần của một hàm:

* Kiểu trả về: Một hàm có thể trả về một giá trị. Kieu\_tra\_ve là dạng dữ liệu của giá trị mà hàm trả về. Vài hàm cung cấp các hoạt động và không trả về giá trị nào cả. Đó là hàm **void**.
* Tên hàm: Đây là tên thực sự của hàm. Tên hàm và danh sách tham số cấu tạo nên dấu hiệu hàm.
* Danh sách tham số: Khi hàm được gọi, bạn phải truyền vào danh sách các tham số. Một giá trị hướng đến một tham số thực tế. Danh sách tham số có các kiểu, thứ tự và số lượng các tham số của hàm. Các tham số trong hàm là tùy chọn, nghĩa là một hàm có thể không có tham số.
* Thân hàm: Phần thân của một hàm bao gồm tập hợp các lệnh xác định những gì mà hàm thực hiện.

**Ví dụ:** hàm kiểm tra số nguyên tố: Trả về 1 nếu n là SNT, ngược lại trả về 0

int kiemtraSNT(int n){

if(n<=1) return 0;

for(long long i=2; i\*i<=n ++i) if(n%i==0) return 0;

return 1;

}

Lệnh return trong hàm trên trả về giá trị cho hàm.

**1.2 Một số khái niệm:**

Phân biệt tham số hình thức – Tham số thực sự:

* Tham số hình thức: Là các tham số khi khai báo chương trình con
* Tham số thực sự: Là tham số khi gọi chương trình con

**Phân biệt tham biến – Tham trị**

* **Tham biến:** là các đại lượng có giá trị có thể thay đổi sau khi kết thúc chương trình con.
* **Tham trị:** Có giá trị không thay đổi sau khi thực hiện chương trình con. (Dù chương trình con có thay đổi giá trị của nó đi chăng nữa thì khi kết thúc CT con giá trị của nó vẫn như cũ).

Ví dụ 1: Hàm hoán đổi giá trị 2 biến a và b

void hoandoi(int a, int b){

int tg = a;

a = b;

b = tg;

}

Các bạn chạy hàm trên xem kết quả như thế nào. Giá trị a và b vẫn không được hoán đổi.

Trong hàm trên: lúc này a và b được khai báo là 2 tham trị nên giá trị a à b sẽ không đổi khi hàm thực hiện hàm hoandoi.

Ví dụ 2:

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

a\*=2;

b\*=2;

c\*=2;

}

Trong hàm trên: a, b, c là 3 tham biến nên giá trị sẽ thay đổi sau khi hàm thực hiện hàm duplicate.

**Phân biệt biến toàn cục và biến cục bộ:**

* Biến cục bộ: Chỉ được khai báo và dùng riêng trong chương trình con.
* Biến toàn cục: Được khai báo trong chương trình chính và dùng trong toàn chương trình.

**\* Quá tải các hàm.**

Hai hàm có thể có cũng tên nếu khai báo tham số của chúng khác nhau, điều này có nghĩa là bạn có thể đặt cùng một tên cho nhiều hàm nếu chúng có số tham số khác nhau hay kiểu dữ liệu của các tham số khác nhau (hay thậm chí là kiểu dữ liệu trả về khác nhau). Ví dụ:

#include <iostream.h>

int divide (int a, int b) {

return (a/b);

}

float divide (float a, float b){

return (a/b);

}

int main () {

int x=5,y=2;

float n=5.0, m=2.0;

cout << divide (x,y);

cout << "\n";

cout << divide (n,m);

return 0;

}

Kết quả:

2

2.5

Trong ví dụ này chúng ta định nghĩa hai hàm có cùng tên nhưng một hàm dùng hai tham số kiểu int và hàm còn lại dùng kiểu float. Trình biên dịch sẽ biết cần phải gọi hàm nào bằng cách phân tích kiểu tham số khi hàm được gọi.

**\* Sử dụng hàm:**

Một hàm khi định nghĩa thì chúng vẫn chưa được thực thi trừ khi ta có một lời gọi đến hàm đó.

Cú pháp gọi hàm: <Tên hàm>([Danh sách các tham số])

Ví dụ:

int a = 3, b = 5, c = 9;

duplicate(a, b, c);

cout <<a <<" " <<b <<" " <<c;

Kết quả in ra màn hình: 9 25 49

II. BÀI TẬP

**Bài 9.1:** (**CTC91.CPP**) Em hãy lập trình nhập vào từ bàn phím ba số nguyên và thông báo ra màn hình số nhỏ nhất trong ba số. (Yêu cầu: Xây dựng chương trình con tìm số nhỏ nhất trong 3 số)

\* Ý tưởng: Xây dựng Function tìm Min của 3 số, rồi dùng hàm này để viết chương trình chính.

Ý tưởng tìm số nhỏ nhất trong 3 số: Đơn giản.

Thuật toán:

B1: Nhập vào 3 số a b c

B2: Min 🡨 Mincua3so(a,b,c)

B3: Thông báo Min

\* Viết chương trình

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả** |
| 3 2 4 | 2 |

**Bài 9.2:** (**CTC92.CPP**) Lập trình nhập vào từ bàn phím một xâu S chuẩn, thông báo ra màn hình xâu S sau khi đã in hoa kí tự đầu tiên của mỗi từ trong xâu. Với yêu cầu xây dựng hàm HoadauTu(S) để in hoa kí tự đầu tiên của mỗi từ.

|  |  |
| --- | --- |
| **Dữ liệu nhập** | **Kết quả** |
| le thanh phu | Le Thanh Phu |

**Bài 9.3:** (**CTC93.CPP**) Hãy mô tả hàm Fact(n) để tính giá trị n giai thừa.(0 < N < 20)

Cho một tệp *CTC93.inp* có dòng đầu tiên là số M, m dòng tiếp theo mỗi dòng là một số nguyên dương N. Lập trình đọc dữ liệu từ tệp này, tính giai thừa của m số đó và lưu kết quả vào tệp *CTC93.out*.

|  |  |
| --- | --- |
| *CTC93.inp* | *CTC93.out* |
| 3  2  4  3 | 2  24  6 |

**Bài 9.4:** (**CTC94.CPP**) Hãy mô tả hàm SumDigit(n) trả về giá trị là tổng các chữ số của n, trong đó n là số nguyên không âm.

Lập trình nhập vào dãy số nguyên A1, A2, …, An. Thông báo ra màn hình các số trong dãy cùng tổng các chữ số của chúng.

|  |  |
| --- | --- |
| **CTC94***.inp* | **CTC94***.out* |
| 3  21 30 55 | 21 3  30 3  55 10 |

**Bài 9.5:** (**CTC95.CPP**) Hãy mô tả một chương trình con Swap(a,b) thực hiện nhiệm vụ hoán đổi giá trị của a và b. Lập trình nhập vào dãy số B1, B2, …, Bm. Sắp xếp dãy B thành dãy không giảm (Theo thuật toán [Bubble sort](http://en.wikipedia.org/wiki/Bubble_sort) và thông báo ra màn hình. *(*có sử dụng chương trình con Swap*)*

|  |  |
| --- | --- |
| **CTC95***.inp* | **CTC95***.out* |
| 3  21 30 55 | 21 30 55 |

**Bài 9.6:** (**CTC96.CPP**) Xây dựng chương trình con UCLN(a,b) trả về giá trị là ước số chung lớn nhất của a và b. Cho tệp CTC96*.inp* gồm bốn số nguyên viết trên một dòng. Hãy lập trình đọc dữ liệu từ tệp này, sử dụng chương trình con UCLN để tìm ước số chung lớn nhất của bốn số nguyên này. Kết quả lưu vào tệp CTC96*.out*

|  |  |
| --- | --- |
| **CTC96***.inp* | **CTC96***.out* |
| 2 4 6 8 | 2 |

**Bài 9.7:** (**CTC97.CPP**) Hãy xây dựng chương trình con NORM(S) để xóa các kí tự “0” không có nghĩa của một xâu kí tự số.

Cho hai xâu u và v chỉ chứa các kí tự số thập phân. Hãy xây dựng hàm COMP(u,v) trả về giá trị True nếu số biểu diễn bởi u lớn hơn số biểu diễn bởi v.

Ví dụ: u=“91” và v=“0089” thì COMP(u,v) cho giá trị True

U=“12” và v=“120” thì COMP(u,v) cho giá trị False

Lập trình nhập vào hai xâu kí tự số thập phân và thông báo ra màn hình xâu kí tự biểu diễn số lớn hơn.

|  |  |
| --- | --- |
| **CTC97***.inp* | **CTC97***.out* |
| 0089  120 | 120 |