

**SỞ GIÁO DỤC VÀ ĐÀO TẠO NAM ĐỊNH TRƢỜNG THPT CHUYÊN LÊ HỒNG PHONG**

**Ngôn ngữ lập trình C++ Những điều thầy muốn kể (Tài liệu lƣu hành nội bộ)**

**Tác giả: Đặng Trung Kiên**

**Trình độ chuyên môn: - Cử nhân Sƣ phạm Tin học**

* **Cử nhân Sƣ phạm Toán học**

**Chức vụ:** **Giáo viên**

**Nơi công tác: Trƣờng THPT chuyên Lê Hồng Phong**

*Ngôn ngữ lập trình C++*



[**MỤC LỤC**](#bookmark=id.30j0zll)

[**BÀI 2. CÁC THÀNH PHẦN CỦA NGÔN NGỮ LẬP TRÌNH**](#bookmark=id.3znysh7)

[1. Các thành phần cơ bản](#bookmark=id.3znysh7) [4](#bookmark=id.3znysh7)

[2. Một số khái niệm](#bookmark=id.3dy6vkm) [7](#bookmark=id.3dy6vkm)

[Bài tập](#bookmark=id.lnxbz9) [14](#bookmark=id.lnxbz9)

[**CHƢƠNG II. CHƢƠNG TRÌNH ĐƠN GIẢN**](#bookmark=id.35nkun2)[**15**](#bookmark=id.35nkun2)

[**BÀI 3. CẤU TRÚC CHƢƠNG TRÌNH**](#bookmark=id.1ksv4uv)[**16**](#bookmark=id.1ksv4uv)

[1. Cấu trúc chung](#bookmark=id.1ksv4uv) [16](#bookmark=id.1ksv4uv)

[2. Các thành phần của chương trình](#bookmark=id.1ksv4uv) [16](#bookmark=id.1ksv4uv)

[3. Ví dụ một chương trình đơn giản](#bookmark=id.3j2qqm3) [20](#bookmark=id.3j2qqm3)

[**BÀI 4. MỘT SỐ KIỂU DỮ LIỆU CHUẨN**](#bookmark=id.2xcytpi)[**23**](#bookmark=id.2xcytpi)

[1. Kiểu số nguyên](#bookmark=id.2xcytpi) [23](#bookmark=id.2xcytpi)

[2. Kiểu thực](#bookmark=id.1ci93xb) [24](#bookmark=id.1ci93xb)

[3. Kiểu kí tự](#bookmark=id.3whwml4) [25](#bookmark=id.3whwml4)

[**BÀI 5. KHAI BÁO BIẾN**](#bookmark=id.2bn6wsx)[**26**](#bookmark=id.2bn6wsx)

[1. Cú pháp khai báo biến](#bookmark=id.2bn6wsx) [26](#bookmark=id.2bn6wsx)

[2. Phạm vi biến](#bookmark=id.qsh70q) [27](#bookmark=id.qsh70q)

[**BÀI 6. PHÉP TOÁN, BIỂU THỨC, CÂU LỆNH GÁN**](#bookmark=id.147n2zr)[**32**](#bookmark=id.147n2zr)

[1. Phép toán](#bookmark=id.147n2zr) [32](#bookmark=id.147n2zr)

[2. Biểu thức số học](#bookmark=id.23ckvvd) [34](#bookmark=id.23ckvvd)

[3. Hàm số học chuẩn](#bookmark=id.ihv636) [35](#bookmark=id.ihv636)

[5. Biểu thức logic](#bookmark=id.2grqrue) [39](#bookmark=id.2grqrue)

[6. Câu lệnh gán](#bookmark=id.2grqrue) [39](#bookmark=id.2grqrue)

[**BÀI 7. CÁC HÀM CHUẨN VÀO/RA ĐƠN GIẢN**](#bookmark=id.3fwokq0)[**41**](#bookmark=id.3fwokq0)

[1. Nhập dữ liệu vào từ bàn phím](#bookmark=id.3fwokq0) [41](#bookmark=id.3fwokq0)

[2. Đưa dữ liệu ra màn hình](#bookmark=id.1v1yuxt) [42](#bookmark=id.1v1yuxt)

[**BÀI 8. SOẠN THẢO, DỊCH, THỰC HIỆN VÀ HIỆU CHỈNH**](#bookmark=id.19c6y18)

[**CHƢƠNG TRÌNH**](#bookmark=id.19c6y18)[**45**](#bookmark=id.19c6y18)

[**BÀI 9. CẤU TRÚC RẼ NHÁNH**](#bookmark=id.37m2jsg)[**49**](#bookmark=id.37m2jsg)

[1. Rẽ nhánh](#bookmark=id.37m2jsg) [49](#bookmark=id.37m2jsg)

[2. Câu lệnh if](#bookmark=id.46r0co2) [51](#bookmark=id.46r0co2)

[3. Câu lệnh ghép](#bookmark=id.46r0co2) [51](#bookmark=id.46r0co2)

[4. Một số ví dụ](#bookmark=id.111kx3o) [53](#bookmark=id.111kx3o)

[5. Bài tập](#bookmark=id.206ipza) [55](#bookmark=id.206ipza)



*Đặng Trung Kiên* *Trang 2*

*Ngôn ngữ lập trình C++*



[**BÀI 10. CẤU TRÚC LẶP**](#bookmark=id.4k668n3)[**56**](#bookmark=id.4k668n3)

[1. Lặp](#bookmark=id.4k668n3) [56](#bookmark=id.4k668n3)

[2. Lặp với số lần biết trước và câu lệnh for](#bookmark=id.2zbgiuw) [57](#bookmark=id.2zbgiuw)

[3. Lặp với số lần chưa biết trước và câu lệnh while](#bookmark=id.4bvk7pj) [64](#bookmark=id.4bvk7pj)

[Bài tập](#bookmark=id.kgcv8k) [69](#bookmark=id.kgcv8k)

[**BÀI 11. KIỂU MẢNG**](#bookmark=id.34g0dwd)[**70**](#bookmark=id.34g0dwd)

[1. Kiểu mảng một chiều](#bookmark=id.34g0dwd) [70](#bookmark=id.34g0dwd)

[2. Khai báo mảng một chiều](#bookmark=id.34g0dwd) [70](#bookmark=id.34g0dwd)

[4. Khai báo mảng nhiều chiều](#bookmark=id.4h042r0) [77](#bookmark=id.4h042r0)

[Bài tập về mảng](#bookmark=id.1baon6m) [79](#bookmark=id.1baon6m)

[**BÀI 12. KIỂU XÂU**](#bookmark=id.pkwqa1)[**82**](#bookmark=id.pkwqa1)

[1. Khai báo và nhập dữ liệu cho xâu](#bookmark=id.pkwqa1) [82](#bookmark=id.pkwqa1)

[2. Các thao tác xử lý xâu](#bookmark=id.3mzq4wv) [88](#bookmark=id.3mzq4wv)

[3. Cách chuyển từ xâu sang số, từ số sang xâu](#bookmark=id.2fk6b3p) [94](#bookmark=id.2fk6b3p)

[**BÀI 13. KIỂU CẤU TRÚC TRONG C++ (STRUCT)**](#bookmark=id.3s49zyc)[**101**](#bookmark=id.3s49zyc)

[**BÀI 14. KIỂU DỮ LIỆU TỆP**](#bookmark=id.36ei31r)[**104**](#bookmark=id.36ei31r)

[1. Vai trò của kiểu tệp](#bookmark=id.36ei31r) [104](#bookmark=id.36ei31r)

[2. Phân loại tệp](#bookmark=id.36ei31r) [104](#bookmark=id.36ei31r)

[**BÀI 15. THAO TÁC VỚI TỆP**](#bookmark=id.45jfvxd)[**106**](#bookmark=id.45jfvxd)

[1. Thao tác với tệp](#bookmark=id.45jfvxd) [106](#bookmark=id.45jfvxd)

[**BÀI 16. VÍ DỤ LÀM VIỆC VỚI TỆP**](#bookmark=id.zu0gcz)[**108**](#bookmark=id.zu0gcz)

[**CHƢƠNG VI. CHƢƠNG TRÌNH CON VÀ LẬP TRÌNH CÓ CẤU**](#bookmark=id.1yyy98l)

[**TRÚC**](#bookmark=id.1yyy98l)[**110**](#bookmark=id.1yyy98l)

[**BÀI 17. CHƢƠNG TRÌNH CON VÀ PHÂN LOẠI**](#bookmark=id.1yyy98l)[**110**](#bookmark=id.1yyy98l)

[1. Khái niệm chương trình con (hàm trong C++)](#bookmark=id.1yyy98l) [110](#bookmark=id.1yyy98l)

[2. Tham trị và tham chiếu](#bookmark=id.4iylrwe) [111](#bookmark=id.4iylrwe)

[a) Truyền tham số bằng tham trị](#bookmark=id.4iylrwe) [111](#bookmark=id.4iylrwe)

[b) Truyền tham số bằng tham chiếu (thêm dấu & trước tham số trong](#bookmark=id.2y3w247)

[hàm)](#bookmark=id.2y3w247) [112](#bookmark=id.2y3w247)

[3. Đệ quy](#bookmark=id.1d96cc0) [113](#bookmark=id.1d96cc0)



*Đặng Trung Kiên* *Trang 3*

*Ngôn ngữ lập trình C++*



**BÀI 2. CÁC THÀNH PHẦN CỦA NGÔN NGỮ LẬP TRÌNH**

Trong bài này, các em được tìm hiểu các thành phần của một ngôn ngữ lập trình. Tựa như việc các em đang học một ngôn ngữ mới.

***1. Các thành phần cơ bản***

Giống như việc học một ngôn ngữ mới, học ngôn ngữ lập trình các em cũng phải học hệ thống các kí hiệu của ngôn ngữ đó, cách các kí tự tổ hợp với nhau theo quy tắc nào (cú pháp) để có nghĩa

Mỗi ngôn ngữ lập trình thường có ba thành phần cơ bản là:

* + **Bảng chữ cái**
  + **Cú pháp**
  + **Ngữ nghĩa**

1. **Bảng chữ cái** là tập các kí tự được dùng để viết chương trình. **Không**

**đƣợc phép dùng bất kỳ kí tự nào ngoài các kí tự quy định trong bảng chữ cái**

**Trong C++, bảng chữ cái bao gồm các kí tự sau:**

* Các chữ cái thường và các chữ cái in hoa của bảng chữ cái Tiếng

Anh

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

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

* 10 chữ số thập phân 0 1 2 3 4 5 6 7 8 9
* Các kí tự đặc biệt

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| + | - | \* | / | = | < | > | [ | ] | . | , |
|  |  |  |  |  |  |  |  |  |  |  |
| ; | # | ^ | $ | @ | & | ( | ) | { | } | : |
|  |  |  |  |  |  |  |  |  |  |  |
| ‗ | % | \_ | \ | ! |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

Các kí tự đặc biệt còn có thêm các dấu không nhìn thấy như dấu cách (số thứ tự trong bảng mã ASCII 32, kí hiệu là ‗ ‗ ), dấu nhảy cách tab, dấu xuống dòng (số thứ tự trong bảng mã ASCII là 13, còn kí hiệu là ‗\n‘)



*Đặng Trung Kiên* *Trang 4*

*Ngôn ngữ lập trình C++*



Việc thực hành thường xuyên sẽ khiến các em nhớ dần các kí hiệu của một ngôn ngữ lập trình. Một số lưu ý dành cho các em

Các em nên nhớ số thứ tự trong bộ mã ASCII của các kí tự, vì có lúc ta sẽ phải dùng đến chúng:

* + Ví dụ, kí tự ‗A‘ có số thứ tự trong bộ mã ASCII là 65, cho tới kí tự ‗Z‘ có số thứ tự trong bộ mã ASCII là 90
  + Kí tự ‗a‘ có số thứ tự trong bộ mã ASCII là 97, cho tới kí tự ‗z‘ có số thứ tự trong bộ mã ASCII là 122
  + Kí tự ‗0‘ có số thứ tự trong bộ mã ASCII là 48, cho tới kí tự ‗9‘ có số thứ tự trong bộ mã ASCII là 57.

1. **Cú pháp**

Khi học Tiếng Việt, có những quy tắc ta tuân theo như chủ ngữ rồi đến vị ngữ. Ví dụ: ―Em sẽ quyết tâm đạt huy chương vàng Tin học quốc tế‖. Có rất nhiều quy tắc phải tuân theo trong Tiếng Việt, những quy tắc như vậy thường được gọi là cú pháp của ngôn ngữ.

Trong ngôn ngữ lập trình, cú pháp là bộ quy tắc để viết chương trình. Dựa vào chúng, người lập trình và chương trình dịch biết được tổ hợp nào của các kí tự trong bảng chữ cái **là hợp lệ** và tổ hợp nào là **không hợp lệ**.Nhờ đó có thể mô tả chính xác thuật toán để máy thực

hiện. **Nếu tổ hợp các kí tự không hợp lệ, chƣơng trình dịch sẽ thông** **báo lỗi cú pháp tới ngƣời lập trình**

Lấy ví dụ, **thông thƣờng** các câu lệnh khi kết thúc sẽ có dấu chấm phẩy để đánh dấu kết thúc câu lệnh đó, giống kiểu viết hết một câu trong Tiếng Việt thì phải có dấu chấm (có những lúc cuối câu lệnh không cần phải có dấu chấm phẩy):

Xét chương trình sau trong ngôn ngữ lập trình C++:

*#include <iostream>*

*using namespace std;*

*int main()*



*Đặng Trung Kiên* *Trang 5*

*Ngôn ngữ lập trình C++*



*{*

*cout << "Hello world!";*

*return 0;*

*}*

Câu lệnh *cout <<"Hello world!" ;* này viết đúng cú pháp. Tuy nhiên, nếu ta bỏ dấu ; ở vị trí cuối câu, chương trình dịch có thể báo lỗi sau: “ expected ; before return” nghĩa là mong đợi người lập trình thêm dấu ; trước return

**c) Ngữ nghĩa**

Tổ hợp các kí tự trong một ngôn ngữ, nếu đúng cú pháp, thường mang một ý nghĩa nhất định. Ví dụ, ―An‖ trong Tiếng Việt có thể mang nghĩa là an lành, bình an, nhưng trong Tiếng Anh nó có nghĩa là một, một cái gì đó. Từ ―Ấy‖ trong Tiếng Việt thì có thể mang rất nhiều ý nghĩa, trong từng ngữ cảnh, ví dụ có thể thay cho một người, có thể thay cho một hành động. Ý nghĩa của tổ hợp các kí hiệu trong một ngôn ngữ thường được gọi là ngữ nghĩa của ngôn ngữ đó.

Đối với ngôn ngữ lập trình cũng vậy, ngữ nghĩa xác định ý nghĩa thao tác cần phải thực hiện, ứng với tổ hợp kí tự dựa vào ngữ cảnh của nó.

Ví dụ: Xét tổ hợp các kí hiệu sau:

1. A+B
2. I+J

Với giả thiết A và B là hai số thực, I, J là hai số nguyên. Khi đó dấu ‗+‘ trong biểu thức (1) được hiểu là cộng hai số thực, kết quả thu được sẽ phải là một số thực, dấu ‗+‘ trong biểu thức (2) được hiểu là cộng hai số nguyên, do đó kết quả thu được sẽ phải là một số nguyên. Như vậy, ngữ nghĩa (ý nghĩa) của dấu ‗+‘ trong hai ngữ cảnh khác nhau là khác nhau.



*Đặng Trung Kiên* *Trang 6*

*Ngôn ngữ lập trình C++*



Tóm lại, cú pháp cho biết cách viết một chương trình hợp lệ, con ngữ nghĩa xác định ý nghĩa của các tổ hợp kí tự trong chương trình.

Các lỗi cú pháp được chương trình dịch phát hiện và thông báo cho người lập trình biết. Chỉ có các chương trình không còn lỗi cú pháp mới dịch được sang ngôn ngữ máy.

Các lỗi ngữ nghĩa khó phát hiện hơn. Phần lớn các lỗi ngữ nghĩa chỉ được phát hiện khi thực hiện chương trình trên dữ liệu cụ thể Ví dụ:

#include <iostream>

using namespace std;

int main()

{ int x=5,y=0; cout <<x/y;

return 0;

}

Đây là chương trình đưa ra màn hình kết quả của phép chia x cho y, không hề có lỗi sai về cú pháp, nên chương trình dịch không hề báo lỗi, nhưng khi chạy chương trình lại xuất hiện lỗi về ngữ nghĩa. Đó là phép chia cho số 0. Tuy nhiên nếu thay y bằng một số khác 0 thì chương trình lại ra một kết quả đúng.

1. *Một số kháiNiệm* **a) Tên**

Mọi đối tượng trong chương trình đều phải được đặt tên theo quy tắc của ngôn ngữ lập trình và từng chương trình dịch cụ thể. Tên hay còn

gọi là định danh (identificater).

**Quy tắc đặt tên trong ngôn ngữ lập trình C++:** Tên là dãy kí tự liềnnhau gồm các chữ cái a..z (từ a tới z), A..Z (từ A tới Z), các chữ số 0..9 (từ 0 tới 9), và dấu gạch nối dưới (dấu \_). **Tên bắt đầu bằng chữ cái**



*Đặng Trung Kiên* *Trang 7*

*Ngôn ngữ lập trình C++*



**hoặc dấu gạch dƣới**. Tuy nhiên, ***tên không được bắt đầu bằng chữ số.***

***Tên không được chứa các kí tự đặc biệt*** như dấu cách, dấu chấm câu,….

Ví dụ về cách đặt tên đúng: toi\_yeu\_lop10tin, dp, dfs

Ví dụ về cách đặt tên sai:

toi-yeu-lop10tin (sai vì chứa dấu gạch ngang)

3PI (sai vì bắt đầu bằng chữ số)

Pi$ (sai vì chứa kí tự đặc biệt $)

―pi‖ (sai vì chứa kí tự đặc biệt ―)

**Lƣu ý: Khác với Pascal, Ngôn ngữ lập trình C++ có phân biệt chữ hoa và chữ thƣờng. Do vậy, tên n sẽ khác tên N**

Ngôn ngữ lập trình C++ chia ra làm ba loại tên:

* Tên dành riêng
* Tên chuẩn
* Tên do người lập trình đặt

**Tên dành riêng**

Một số tên được ngôn ngữ lập trình quy định dùng với ý nghĩa riêng xác định, ***người lập trình không sử dụng với ý nghĩa khác***. Những tên này được gọi là tên dành riêng (còn được gọi là từ khóa).

**Chú ý: Không đƣợc đặt tên (tên biến, tên hằng, tên kiểu, tên hàm,…)**

**trùng với các từ khóa**

Ví dụ: Một số tên dành riêng: main, include, if, while, void

Main: ý nghĩa là hàm chính, chương trình C++ có thể có nhiều hàm, khi chạy chương trình, sẽ chạy từ hàm chính trước.

Include: trong Tiếng Anh là bao gồm, bao hàm, thường được dùng với ý nghĩa khai báo các thư viện

If: nghĩa là nếu, xuất hiện trong câu lệnh rẽ nhánh nếu…thì, nếu một điều kiện xảy ra thì thực hiện lệnh,….



*Đặng Trung Kiên* *Trang 8*

*Ngôn ngữ lập trình C++*



**Tên chuẩn**

Một số tên được ngôn ngữ lập trình dùng với ý nghĩa nhất định nào đó. Những tên này được gọi là tên chuẩn. ***Tuy nhiên, người lập trình có*** ***thể khai báo và dùng chúng với ý nghĩa và mục đích khác***.

* nghĩa của các tên chuẩn được quy định trong các thư viện của ngôn ngữ lập trình

Ví dụ trong ngôn ngữ lập trình C++

Cin cout min

Ví dụ về tên **min**,

* nghĩa 1: trong thư viện mang tên <algorithm> của C++ (thư viện của ngôn ngữ lập trình là nơi chứa một số chương trình thông dụng đã được lập trình sẵn), min được dùng với ý nghĩa tên của hàm dùng để tìm giá trị nhỏ nhất trong hai số

#include <iostream>

#include<algorithm>

using namespace std;

int main()

{ cout<<min(6,8); return 0;

}

Chương trình trên sẽ đưa ra giá trị 6, vì 6 là giá trị nhỏ nhất trong hai số 6 và 8

Ý nghĩa khác của tên **min**

#include <iostream>

#include<algorithm>

using namespace std;

int main()

{ int min=9;



*Đặng Trung Kiên* *Trang 9*

*Ngôn ngữ lập trình C++*



cout<<min;

return 0;

}

Câu lệnh int min=9 có ý nghĩa khai báo min là một biến nguyên và giá trị ban đầu của số đó bằng 9. Như vậy ở đây từ min lại mang một ý nghĩa khác.

**Tên do ngƣời lập trình đặt**

Tên do người lập trình đặt được dùng với ý nghĩa riêng, xác định bằng cách khai báo trước khi sử dụng. Các tên này không được trùng với tên dành riêng.

Ví dụ:

Tên do người lập trình đặt:

TIN

TUHOC

THANHCONG

1. **Hằng và biến Hằng**

Hằng là đại lượng có giá trị không thay đổi trong quá trình thực hiện chương trình.

Trong ngôn ngữ lập trình C++, có các loại hằng sau:

* + Hằng số học là các số nguyên hay số thực (dấu phẩy tĩnh hoặc dấy phẩy động)
  + Hằng logic là giá trị đúng hoặc sai tương ứng với true hoặc false.
  + Hằng xâu là dãy kí tự trong bộ mã ASCII. Khi viết, dãy kí tự này được đặt trong dấu nháy kép ―
  + Hằng kí tự chỉ gồm một kí tự trong bộ mã ASCII. Khi viết, kí tự này được đặt trong dấu nháy đơn ‗

**Ví dụ:**



*Đặng Trung Kiên* *Trang 10*

*Ngôn ngữ lập trình C++*



**Hằng số học** 2 10 -5 +18

1e6 (nghĩa là 1 nhân với 10 mũ 6, e nghĩa là 10)

**Hằng logic**

Trong C++:

true false

Chú ý là C++ phân biệt hoa thường nên không được viết là TRUE FALSE

Về hằng logic trong C++, true còn mang giá trị 1, false còn mang giá trị 0, nên khi đưa giá trị các biến ra màn hình, có thể sẽ hiện ra 1 hoặc 0, chứ không hiện ra true hoặc false. Tuy nhiên, với một biến x nếu mang giá trị logic, ta gán cho x bằng 1 hoặc true đều giống nhau.

* **Hằng xâu** ―Information‖, ―Lop 10 Tin‖

Lưu ý: ở đây xâu ―Lop 10 Tin‖ có dấu cách giữa các từ, theo quy tắc đặt tên thì không được phép có dấu cách, tuy nhiên với tên xâu thì không cần theo quy tắc đó, dấu cách ở trong xâu này chính là một kí tự trong bộ mã

ASCII.

- **Hằng kí tự** ‗a‘ ‗0‘ ‗1‘

Lưu ý là ‗0‘ ở đây là kí tự 0, chứ không phải là số 0 **Biến**

Biến là đại lượng được đặt tên, dùng để lưu trữ giá trị có thể được thay đổi trong quá trình thực hiện chương trình. Ví dụ, biến x, biến y, biến z,….

**c) Chú thích**

Có thể đặt các đoạn chú thích trong chương trình nguồn. Các chú thích này giúp cho người đọc chương trình nhận biết ý nghĩa của chương trình đó dễ hơn. **Chú thích không ảnh hƣởng đến nội dung chƣơng** **trình nguồn và đƣợc chƣơng trình dịch bỏ qua**.

**Các cách tạo chú thích trong C++ là:**

* **Đặt // trên một dòng**



*Đặng Trung Kiên* *Trang 11*

*Ngôn ngữ lập trình C++*



* **Đặt giữa cặp dấu /\* và \*/**

**Ví dụ:**

#include <iostream>

#include<algorithm>

using namespace std;

int main()

{ for(long i=2;i<=100;i++)

if (i%2==0)cout<<i<<" ";//Neu i la so chan thi dua gia tri i ra man hinh

return 0;



*Đặng Trung Kiên* *Trang 12*

*Ngôn ngữ lập trình C++*



**TÓM TẮT**

* Các thành phần của ngôn ngữ lập trình: bảng chữ cái, cú pháp, và ngữ nghĩa
* Mọi đối tượng trong chương trình đều phải được đặt tên**:**
  + Tên dành riêng: Được dùng với ý nghĩa riêng, không được dùng với ý nghĩa khác.
  + Tên chuẩn: Tên dùng với ý nghĩa nhất định, khi cần dùng với ý nghĩa khác thì phải khai báo
  + Tên do người lập trình đặt: Cần khai báo trước khi sử dụng
* Hằng: Đại lượng có giá trị không thay đổi trong quá trình thực hiện chương trình
* Biến: Đại lượng được đặt tên. Gía trị của biến có thể thay đổi trong quá trình thực hiện chương trình.



*Đặng Trung Kiên* *Trang 13*

*Ngôn ngữ lập trình C++*



***Bài tập***

1. Chương trình dịch là gì? Tại sao cần phải có chương trình dịch?

2. Biên dịch và thông dịch khác nhau như thế nào?

3. Hãy cho biết các điểm khác nhau giữa tên dành riêng và tên chuẩn?

4. Hãy tự viết ra ba tên đúng theo quy tắc đặt tên của ngôn ngữ lập trình C++?

5. Hãy cho biết những biểu diễn nào dưới đây không phải là biểu diễn hằng trong ngôn ngữ lập trình C++ và chỉ rõ lỗi trong từng trường hợp, nếu đúng biểu diễn hằng trong ngôn ngữ lập trình C++, hãy chỉ rõ đó là hằng gì:

1. 150.0
2. -22
3. 6,23
4. ‗43‘
5. A20
6. 1.06E-15
7. 4+6

h)‘C‘

i) ‗TRUE‘



*Đặng Trung Kiên* *Trang 14*

*Ngôn ngữ lập trình C++*



**CHƢƠNG II. CHƢƠNG TRÌNH ĐƠN GIẢN**

Trong chương trình này, các em được tìm hiểu cấu trúc một chương trình viết bằng ngôn ngữ lập trình C++, các kiến thức cơ bản về kiểu dữ liệu, phép toán, biểu thức, câu lệnh gán, các tổ chức vào/ra đơn giản, và cách thực hiện chương trình viết bằng ngôn ngữ lập trình C++, trên phần mềm Codeblock



*Đặng Trung Kiên* *Trang 15*

*Ngôn ngữ lập trình C++*



**BÀI 3. CẤU TRÚC CHƢƠNG TRÌNH *1. Cấu trúc chung***

Chương trình viết bằng ngôn ngữ lập trình C++ gồm hai phần: phần khai báo và phần thân. Phân thân chương trình nhất thiết phải có. Phần khai báo có thể có hoặc không tùy theo từng chương trình cụ thể.

Khi diễn giải cú pháp của ngôn ngữ lập trình người ta thường sử dụng ngôn ngữ tự nhiên. Các diễn giải bằng ngôn ngữ tự nhiên được đặt giữa cặp dấu < và dấu >. Các thành phần của chương trình có thể có hoặc không được đặt trong cặp dấu [ và ]

Với quy ước trên, cấu trúc của một chương trình viết bằng ngôn ngữ lập trình C++ có thể được mô tả như sau:

**[<phần khai báo>]**

**<phần thân>**

1. ***Các thành phần của chương trình* a) Phần khai báo**

Hiểu một cách đơn giản, phần khai báo được dùng để nói với chương trình dịch những thứ cần thiết có thể được dùng trong phần thân. Giống như phần mở bài trong một đoạn văn, chủ đề tôi cần nói là gì, tôi sẽ đề cập đến nội dung nào, vấn đề nào trong thân bài?. Vậy thì phần khai báo cũng vậy, ví dụ, nếu ta muốn hướng dẫn máy tính làm phép tính x+y trong phần thân thì trong phần khai báo, ta sẽ phải nói x, y là biến gì, biến nguyên hay biến thực. Nếu không khai báo x, y là biến gì, thì trong phần thân, chương trình dịch không thể hiểu x, y là biến gì, sẽ báo lỗi cú pháp. Tôi muốn tìm min của hai số x, và y, tôi sẽ phải dùng đến hàm tìm min, hàm này lại nằm trong thư viện <algorithm>, do vậy, tôi phải khai báo thư viện <algorithm> để chương trình dịch hiểu rằng, tôi có thể sử dụng hàm min trong thư viện <algorithm>

**Có thể có các khai báo cho:**



*Đặng Trung Kiên* *Trang 16*

*Ngôn ngữ lập trình C++*



* Thư viện
* Hằng
* Biến
* Chương trình con (gọi là hàm trong C++)

**Khai báo thƣ viện**

Mỗi ngôn ngữ lập trình thường có sẵn một số thư viện cung cấp một số chương trình thông dụng đã được lập sẵn. Để sử dụng các chương trình đó cần phải khai báo thư viện chứa nó. Ví dụ: Khai báo thư viện trong C++

#include<iostream>

Thư viện *iostream* trong C++ cung cấp các chương trình (lệnh, hàm) thực hiện các nhiệm vụ vào ra dữ liệu chuẩn (input, output). Ví dụ muốn đưa màn hình dòng chữ ―Xin chào‖, ta dùng lệnh ***cout<<”Xin chao”;***

Tuy nhiên nếu ta dùng lệnh ***cout<<”Xin chao”;*** luôn, mà không khai báo thư viện *#include<iostream>* thì chương trình dịch sẽ báo lỗi cú pháp: ―cout was not declare in this scope‖ nghĩa là từ cout lúc này lại được hiểu là một định danh, và chương trình dịch cho rằng nó không được khai báo kiểu dữ liệu. Do vậy, muốn sử dụng lệnh cout với ý nghĩa đưa ra màn hình dữ liệu nào đó, ta phải khai báo thư viện chứa cout chính là *#include<iostream>*

***Cách khai báo thư viện trong C++:***

***#include<tên thư viện>***

***Khai báo hằng***

Ví dụ: Khai báo hằng trong C++

const int maxn=1000;

const double pi=3.1416

***Cú pháp của khai báo hằng trong C++ là:***



*Đặng Trung Kiên* *Trang 17*

*Ngôn ngữ lập trình C++*



const <tên kiểu dữ liệu> <tên biến> = <giá trị>

Khai báo hằng thường được sử dụng cho những giá trị xuất hiện nhiều lần

trong chương trình

**Khai báo biến**

**Chú ý**: ***Tất cả các biến trong chương trình đều phải được đặt tên và* *khai báo cho chương trình dịch biết để lưu trữ, và xử lý***. Biến chỉ nhậnmột giá trị tại mỗi thời điểm thực hiện chương trình được gọi là biến đơn. Cách khai báo biến sẽ được trình bày riêng trong các bài sau. Ví dụ, khai báo biến x có kiểu dữ liệu là số nguyên như sau:

using namespace std;

**long x=0; //1**

int main()

{

**cout<<x+5; //2**

return 0;

}

Đoạn chương trình trên có ý nghĩa khai báo biến x có kiểu dữ liệu là long (nghĩa là số nguyên với phạm vi trong khoảng -2147483648 đến 2147483647), và gán giá trị ban đầu của x=0. Câu lệnh 2 dùng để đưa giá trị x+5 ra màn hình. Nếu ta bỏ đi câu lệnh 1, thì chương trình dịch sẽ báo lỗi: ―***x was not declared in this scope***‖, nghĩa là chương trình dịch không hiểu x có kiểu dữ liệu là gì, hay x không được khai báo (declared).

**b) Phần thân chƣơng trình (thân hàm trong c++)**

Phần thân chương trình giống như phần thân bài và phần kết luận của một bài văn. Nếu trong Toán học, các em hiểu rằng hàm số là giá trị cho tương ứng mỗi biến x thuộc X, một và chỉ một giá trị y thuộc Y sao cho f(x)=y. Thì trong ngôn ngữ lập trình C++ cũng có quan niệm tương ứng. Phần thân chương trình của C++ được quan niệm là một ***hàm số***, trong Tiếng Anh hàm số nghĩa là function (chức năng), nghĩa là thực hiện



*Đặng Trung Kiên* *Trang 18*

*Ngôn ngữ lập trình C++*



một chức năng, nhiệm vụ nào đó. Hàm này gọi là hàm main. Tuy nhiên nó không theo một biến x,y nào cả, bên trong hàm này là các câu lệnh thực hiện một nhiệm vụ nào đó, và cuối cùng sẽ giá về một giá trị (thường mặc định là 0), hoặc nếu người lập trình quên mất không trả về giá trị của hàm này , thì khi kết thúc chương trình, hàm sẽ tự trả về giá trị mặc định là 0

Sau đây là cú pháp của phần thân chương trình:

**int main()**

**{ [khai báo biến] [các câu lệnh]**

**[khai báo biến]**

**[các câu lệnh]**

**return 0;**

**}**

* int là kiểu dữ liệu trả về của hàm, ở đây int là viết tắt của integer (nghĩa là số nguyên)
* main(), main trong tiếng anh nghĩa là chính,khi chạy chương trình trong C++, chương trình sẽ tìm đến hàm main()này đầu tiền (vì sau này có thể có nhiều hàm được cài đặt trong C++- chi tiết có thể tham khao chương trình con trong C++). Hai dấu () là cần thiết phải có, nhưng bên trong không chứa gì cả.
* Return nghĩa là trả về, giá trị trả về của hàm số. Ở đây giá trị trả về của hàm số này thường là 0. Nếu không có câu lệnh return, hàm sẽ tự trả về giá trị 0 khi kết thúc chương trình.
* Trong hàm main, ta có thể khai báo biến (có thể có hoặc không), sau đó dùng tới biến đó (muốn dùng biến nào thì khai báo trước biến đó). **Các biến đƣợc khai báo bên trong một hàm (ở đây là** **hàm main) đƣợc gọi là biến cục bộ (local variable) (biến cục bộ**



*Đặng Trung Kiên* *Trang 19*

*Ngôn ngữ lập trình C++*



**trong C++ là biến đƣợc khai báo bên trong cặp dấu { }, biến đƣợc khai báo bên ngoài cặp dấu {} đƣợc gọi là biến toàn cục)**

* + Cặp dấu {} được gọi là cặp dấu mở đầu và kết thúc cho một chương trình, ở đây là mở đầu và kết thúc cho hàm main.

1. ***Ví dụ một chương trình đơn giản***

Như vậy, ta có thể tóm lại toàn bộ cấu trúc của một chương trình viết bằng ngôn ngữ lập trình C++ là:



**[khai báo thư viện]**

**[khai báo hằng]**

**[khai báo biến**, **biến ở đây gọi là biến toàn cục**]

**[khai báo chương trình con (hàm số)]**

**int main()**

**{**

**[khai báo biến (biến ở đây gọi là biến cục bộ)] [các câu lệnh]**

**return 0**;

**}**

Sau đây ta sẽ lấy ví dụ về một chương trình đơn giản viết bằng ngôn ngữ

lập trình C++

Ví dụ 1:

* my first program in C++ #include <iostream>



*Đặng Trung Kiên* *Trang 20*

*Ngôn ngữ lập trình C++*



int main()

{

std::cout << "Hello World!";

}

Chương trình này làm nhiệm vụ đưa ra màn hình dòng chữ Hello World.

Giải thích:

// my first program in C++

Đây là dòng chú thích

#include <iostream>

Khai báo thư viện <iostream> chứa các chương trình vào ra chuẩn

std::cout << "Hello World!";

***Std:cout***, std là viết tắt của ***standard character output device*** (thiết bị rachuẩn, ở đây thường là màn hình máy tính). Toán tử **<<** biểu thị những gì sẽ được đưa ra sau câu lệnh std:cout. Cuối cùng, ―Hello word!!‖, là nội dung được đưa ra màn hình. Kết thúc câu lệnh này là dấu chấm phẩy ; giống như dấu ngắt trong một câu nói. Tất cả các câu lệnh trong C++ thường kết thúc bằng dấu chẩm phẩy. **Một trong những lỗi lập trình**

**phổ biến của ngƣời lập trình C++ là quên viết dấu chẩm phẩy cuối câu lệnh**

Ví dụ 2. Nhập vào hai số nguyên x, y. Đưa ra màn hình tổng giá trị của hai số x và y

#include<iostream> //1

using namespace std; //2

long x,y; //3

int main() //4

{ cin>>x>>y;//5 cout<<x+y;//6

return 0;//7

} //8



*Đặng Trung Kiên* *Trang 21*

*Ngôn ngữ lập trình C++*



Dòng 3 là khai báo 2 biến toàn cục (global variable) (biến nằm ngoài hai cặp dấu {}) x và y. Dòng 5 là câu lệnh nhập vào giá trị cho hai biến x và y từ màn hình. Toán tử >> thể hiện sẽ nhập dữ liệu cho biến nào ngay sau nó. Dòng 6 là câu lệnh đưa giá trị x+y ra màn hình

**Nhƣng tại sao dòng 6 tôi chỉ cần sử dụng cout mà không dùng std::cout?? nhƣ ví dụ 1**

Vì cout là một thành phần trong thư viện chuẩn của C++. Mà tất cả các thành phần trong thư viện chuẩn của C++ được khai báo bên trong một thứ gọi là **không gian tên, ở đây cout thuộc không gian tên std.** Hiểu một cách đơn giản thế này, không gian tên của người Anh chứa các tên của người Anh, gồm tên trước họ sau, không gian tên của người Việt chứa các tên của người Việt gồm họ trước tên sau và có cả dấu nữa. Vậy thì cout nó thuộc về không gian tên liên quan đến thủ tục vào ra màn hình std.

Do vậy tại dòng thứ 2, ta đã nói với chương trình dịch rằng tôi sắp sử

dụng các từ thuộc về không gian tên std như cout, cin,….. Do đó, tại dòng

thứ 6, ta không cần phải dùng thêm std::cout

**Nhƣ vậy từ nay trở đi, ta sẽ thêm dòng lệnh số 2 vào chƣơng trình**

using namespace std; //2

nghĩa là sử dụng không gian tên std. Từ đó cout, cin có thể viết mà không cần dùng std::cin hoặc std::cout

**Ví dụ này liên quan tới các vấn đề sẽ tìm hiểu sâu hơn trong các bài học tiếp theo, đó là biến cục bộ, biến toàn cục, sự khác nhau giữa biến cục bộ và biến toàn cục, câu lệnh vào/ra**



*Đặng Trung Kiên* *Trang 22*

*Ngôn ngữ lập trình C++*



**BÀI 4. MỘT SỐ KIỂU DỮ LIỆU CHUẨN**

Các bài toán trong thực tế thường có dữ liệu vào và kết quả ra thuộc những kiểu dữ liệu quen biết như số nguyên, số thực, kí tự,… Khi cần lập trình cho những bài toán như vậy, người lập trình sử dụng các kiểu dữ liệu đó thường gặp một số hạn chế nhất định, phụ thuộc vào các yếu tố như dung lượng bộ nhớ, khả năng xử lý của CPU,…

Vì vậy, ***mỗi ngôn ngữ lập trình thường cung cấp một số kiểu dữ*** ***liệu chuẩn cho biết phạm vi giá trị có thể lưu trữ, dung lượng bộ nhớ cần thiết để lưu trữ và các phép toán tác động lên dữ liệu***. Dưới đây xétmột số kiểu dữ liệu chuẩn thường dùng cho các biến đơn trong C++

***1. Kiểu số nguyên***



*Đặng Trung Kiên* *Trang 23*

*Ngôn ngữ lập trình C++*



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Bộ nhớ** |  | **Xấp xỉ** |
|  |  |  |
| **Nhóm** | **Kiểu dữ liệu** | **lƣu trữ** | **Phạm vi giá trị** | **theo** |
|
| **một giá** | **đơn vị** |
|  |  |  |
|  |  |  |
|  |  | **trị** |  | **mũ 10** |
|  |  |  |  |  |
|  | **signed char** | 1 Byte | -128->127 | 102 |
|  |  |  |  |  |
|  | *signed* **short** *int* | 2 Byte | -32768 to 32767 | 104 |
|  |  |  |  |  |
| Số | *signed* **int** | 4 Byte | -2147483648 to | 109 |
| 2147483647 |  |
| nguyên |  |  |  |
|  |  |  |  |
|  |  | -2147483648 to | 109 |
| có dấu | *signed* **long** *int* | 4 Byte |
| 2147483647 |  |
|  |  |  |  |
|  |  |  |  |  |
|  | *signed* **long long** *int* | 8 Byte | -9.2e+18 -> 9.2e+18 | 1018 |
|  |  |  |  |  |
|  | **unsigned char** | 1 Byte | 0-255 (=28-1) | 102 |
|  |  |  |  |  |
| Số | **unsigned short** *int* | 2 Byte | 0->65535 (=216-1) | 104 |
|  |  |  |  |  |
| nguyên | **unsigned** *int* | 4 Byte | 0->4294967295 (=232-1) | 109 |
| không |  |  |  |  |
| **unsigned long** *int* | 4 Byte | 0->4294967295 (=232-1) | 109 |
| dấu |  |  |  |  |
| **unsigned long long** *int* | 8 Byte | 0->1.8\*1019 (=264-1) | 1019 |
|  |  |  |  |  |
|  |  |  |  |  |



**Các chữ in nghiêng có thể không cần viết** khi khai báo kiểu dữliệu. Ví dụ, ta chỉ cần khai báo *short* thay vì viết hẳn ra *signed short int,* ta chỉ cần khai báo *long long* thay vì viết đầy đủ khai báo *signed long* *long int*

***2. Kiểu thực***

Có nhiều kiểu dùng để khai báo các đại lượng nhận giá trị là số thực. Thường dùng hơn cả là kiểu dữ liệu được liệt kê trong bảng sau:



*Đặng Trung Kiên* *Trang 24*

*Ngôn ngữ lập trình C++*



**Bộ nhớ lƣu trữ một giá**

**Phạm vi giá trị**

**Kiểu** **trị**

|  |  |  |
| --- | --- | --- |
| double | 8 Byte | +/- 1.7e +/- 308 (xấp xỉ 15 chữ số) |

***3. Kiểu kí tự***

Ta hiểu kí tự là các kí tự thuộc bộ mã ASCII gồm 256 kí tự có mã ASCII thập phần từ 0 đến 255.

Ví dụ, kí tự ‗A‘ có mã ASCII là 65, kí tự ‗a‘ có mã ASCII là 97. Kí tự đặc biệt, dùng để thể hiện sự ngăn cách giữa hai từ viết liên tiếp trong các văn bản, là dấu cách. Dấu cách được gõ bằng phím Space-phím dài nhất trên bàn phím và có mã ASCII bằng 32. Kí tự xuống dòng trong C++ là ‗\n‘.

|  |  |  |  |
| --- | --- | --- | --- |
| **Kiểu** | **Bộ nhớ lƣu trữ một giá trị** | | **Phạm vi giá trị** |
|  |  |  |  |
| char | 1 Byte | | 256 kí tự trong bộ mã ASCII |
|  |  |  |  |
| **4. Kiểu logic** | | |  |
|  |  |  |  |
|  |  | **Bộ nhớ lƣu trữ một giá** | **Phạm vi giá trị** |
| **Kiểu** |  | **trị** |
|  |  |
|  |  |  |  |
| Bool |  | 1 Byte | true hoặc false |
|  | 1 hoặc 0 |
|  |  |  |
|  |  |  |  |



*Đặng Trung Kiên* *Trang 25*

*Ngôn ngữ lập trình C++*



**BÀI 5. KHAI BÁO BIẾN *1. Cú pháp khai báo biến***

Như đã nêu ở trên, mọi biến dùng trong chương trình đều cần khai báo tên và kiểu dữ liệu. Tên biến dùng để xác lập quan hệ giữa biến với địa chỉ bộ nhớ lưu trữ giá trị biến. Mỗi biến chỉ được khai báo một lần. Trong phần này, ta chỉ xét khai báo các biến đơn. Trong C++, khai báo biến có cú pháp như sau:

***<tên kiểu dữ liệu của biến> <danh sách biến>;***

Trong đó:

* Danh sách biến là một hoặc nhiều tên biến, các tên biến được viết cách nhau bởi dấu phẩy
* Kiểu dữ liệu thường là một trong các kiểu dữ liệu chuẩn được nêu
* bài 4, hoặc kiểu dữ liệu do người lập trình định nghĩa (được trình bày trong chương IV).

Ví dụ 1. Giả sử trong chương trình cần các biến thực A, B, C, D, X1,X2 và các biến nguyên M, N. Khi đó có thể khai báo các biến đó như sau:

double A,B,C,D,X1,X2; //khai báo số thực

long m,n; // khai báo số nguyên

Ví dụ 2. Xét khai báo biến

double x,y,z;

char c;

long i,j;

long long k;

* Trong khai báo này có ba biến thực là x,y,z. Bộ nhớ cấp phát cho ba biến này là 8byte\*3=24 byte.
* Một biến kí tự (hoặc biến nguyên) c, bộ nhớ cấp phát cho biến này là 1 byte.



*Đặng Trung Kiên* *Trang 26*

*Ngôn ngữ lập trình C++*



* Hai biến nguyên i,j với phạm vi giá trị trong khoảng -2147483648

tới 2147483647. Bộ nhớ cấp phát cho 2 biến i,j này là 4\*2=8 byte Như vậy, tổng bộ nhớ dành cho các biến đã khai báo là 24+1+8=25+8=33 (byte)

***Kinh nghiệm rút ra***: Ví dụ 2 là một kinh nghiệm để lựa chọn các biếnsao cho bộ nhớ cấp phát trong giới hạn cho phép. Trong các đề thi của quốc tế, người ta thường giới hạn bộ nhớ dành cho các biến, ví dụ, giới hạn 500MB, hoặc 1GB,…... Do vậy cần phải tính toán hợp lý các

biến sao cho vẫn trong phạm vi cho phép, nhưng đồng thời lại tiết kiệm được bộ nhớ.

**Một số chú ý khi khai báo biến (kỹ năng đặt tên biến):**

* + **Cần đặt tên biến sao cho gợi nhớ đến ý nghĩa của biến đó. Điều này rất có lợi cho việc đọc, hiểu và sửa đổi chƣơng trình khi cần thiết**. Ví dụ, cần đặt tên hai biến biểu diễn điểm toán, điểm tinthì không nên vì ngắn gọn và đặt tên biến là d1,d2 mà nên đặt là dtoan,dtin.
  + **Không nên đặt tên biến quá ngắn hay quá dài, dễ mắc lỗi khi viết nhiều lần tên biến**.Ví dụ, không nên dùng d1,d2, haydiemmontoan, diemmontin cho điểm toán, điểm tin của học sinh
  + **Khi khai báo biến cần đặc biệt lƣu ý đến phạm vi giá trị của nó**.Ví dụ, khai báo biến biểu diễn số học sinh của lớp có thể sửdụng kiểu dữ liệu char, nhưng biểu diễn số học sinh của toàn trường, thì phải thuộc kiểu dữ liệu short

1. *Phạm vi biến*

Trong C++, Biến được khai báo bên trong hàm { }, dù là hàm main

hay không phải hàm main, cứ được khai báo trong cặp dấu {} được gọi là

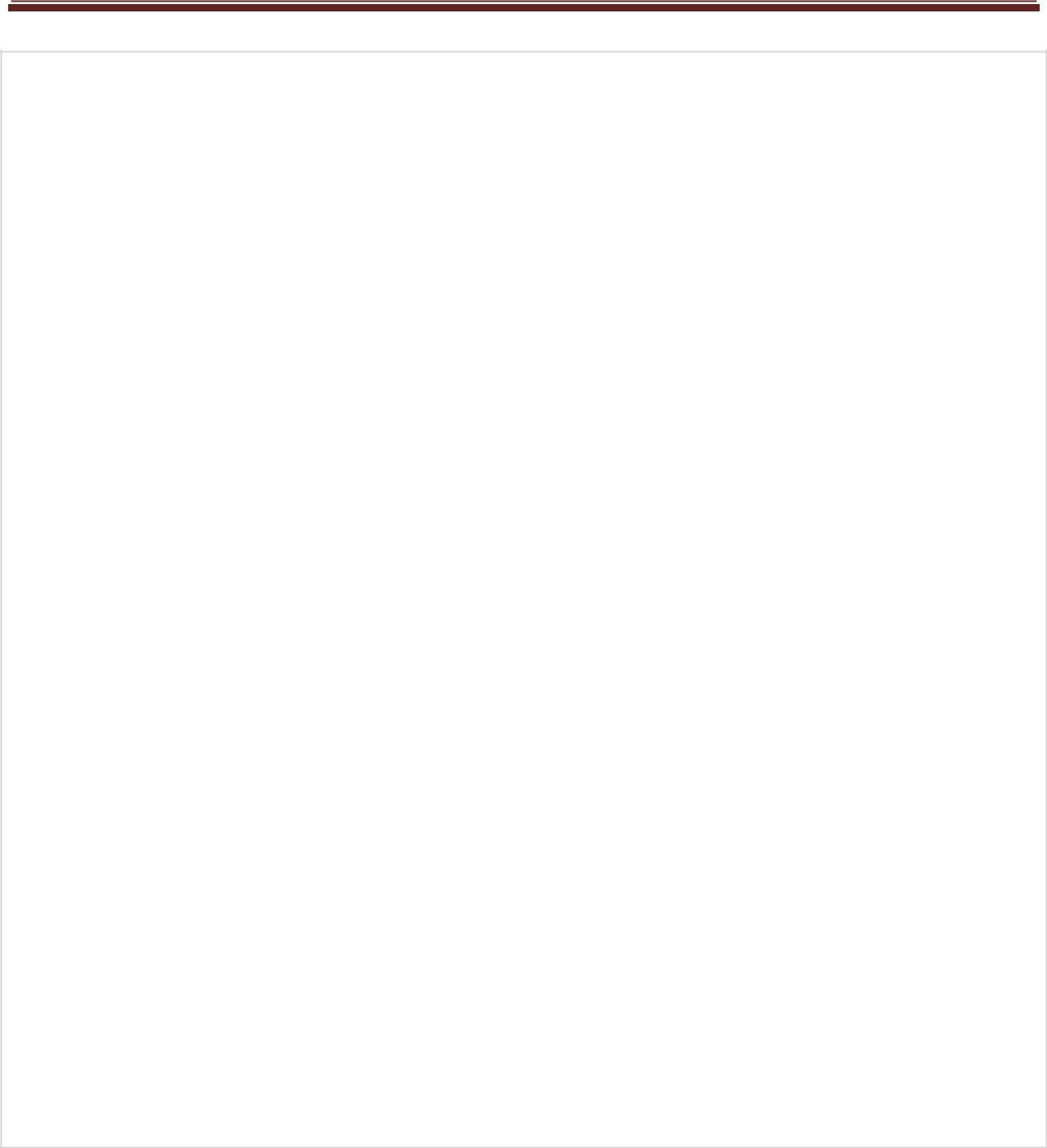
biến cục bộ

Ví dụ



*Đặng Trung Kiên* *Trang 27*

*Ngôn ngữ lập trình C++*



#include <iostream>

using namespace std;

int main ()

{

* Local variable declaration (khai báo biến cục //bộ):

int a, b;

int c;

// actual initialization Khởi tạo giá trị

a = 10;

b = 20;

c = a + b;

cout << c;

return 0;

}

Trong C++, biến được khai báo bên ngoài cặp dấu {}, được gọi là biến toàn cục



#include <iostream>

using namespace std;

// Global variable declaration (biến toàn cục)

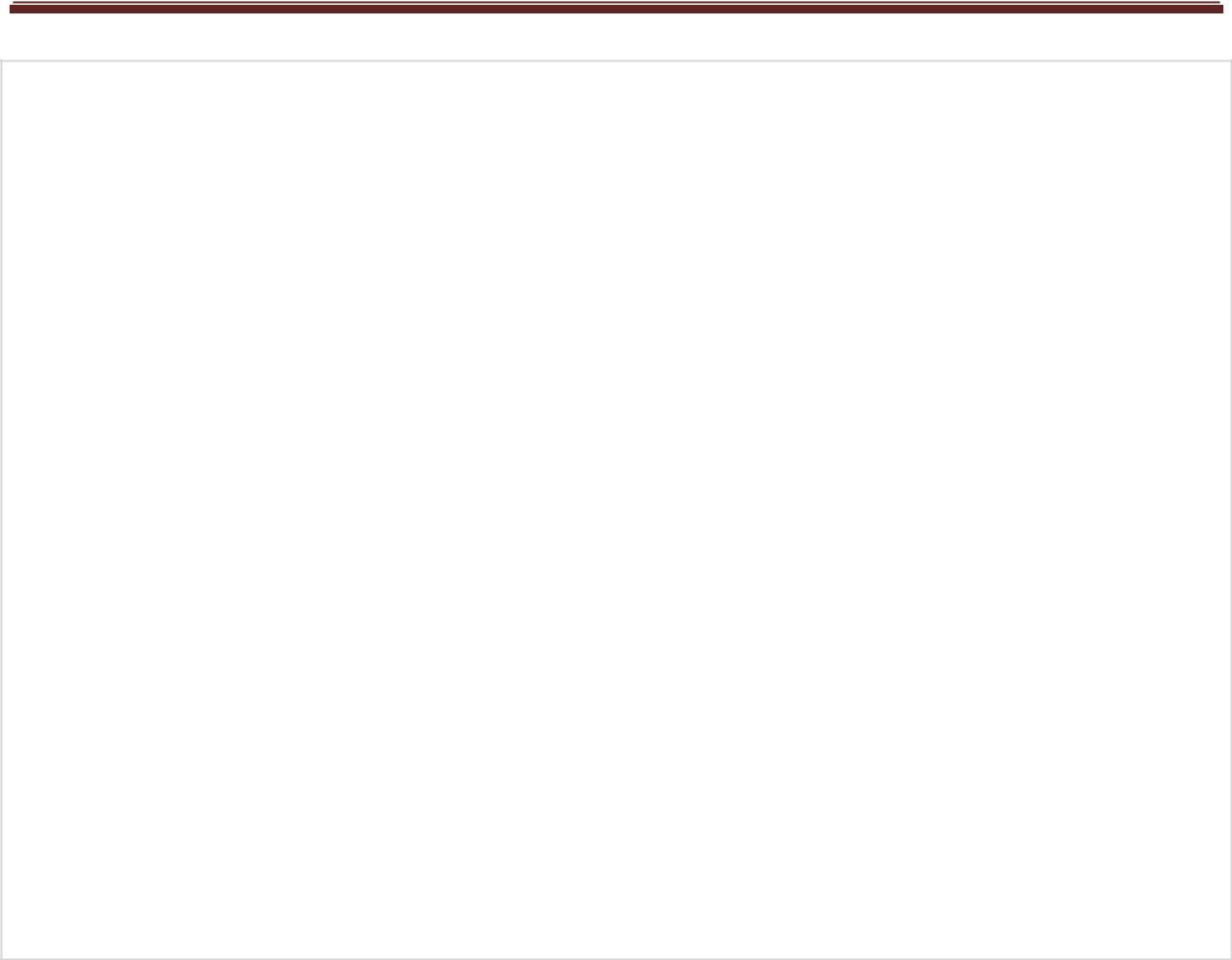
int g;

int main ()



*Đặng Trung Kiên* *Trang 28*

*Ngôn ngữ lập trình C++*



{

// Local variable declaration:biến cục bộ int a, b;

// actual initialization khởi tạo giá trị

a = 10;

b = 20;

g = a + b;

cout << g;

return 0;

}

Bất kỳ một hàm nào cũng có thể sử dụng được biến toàn cục, nhưng riêng biến cục bộ chỉ có phạm vi sử dụng trong hàm khai báo nó, nghĩa là trong cặp dấu{}.

Khi sử dụng cả biến cục bộ và biến toàn cục trong một hàm, biến cục bộ sẽ được ưu tiên



#include <iostream>

using namespace std;

* Global variable declaration: int g = 20;

int main ()

{

* + Local variable declaration: int g = 10;

cout << g; return 0;



*Đặng Trung Kiên* *Trang 29*

*Ngôn ngữ lập trình C++*



}

Ouput: 10

**Khởi tạo biến toàn cục và biến cục bộ**

Khi một biến cục bộ được khai báo, nó sẽ không được khởi tạo giá trị ban đầu, bạn sẽ phải tự khởi tạo cho nó. Khi một biến toàn cục được khai báo, nó sẽ được tự động khởi tạo giá trị ban đầu:



*Đặng Trung Kiên* *Trang 30*

*Ngôn ngữ lập trình C++*



|  |  |
| --- | --- |
|  |  |
| **Kiểu dữ liệu** | **Khởi tạo biến toàn** |
|  | **cục** |
|  |  |
|  |  |
| int | 0 |
|  |  |
| char | '\0' |
|  |  |
| float | 0 |
|  |  |
| double | 0 |
|  |  |
| pointer | NULL |
|  |  |

**Lƣu ý: Luôn quan tâm tới việc khởi tạo biến ban đầu. Khi chấm bài bằng trình biên dịch Themis, nếu biến cục bộ không đƣợc khởi tạo giá trị mà tham gia luôn vào quá trình tính toán, các test có thể không đúng, và bị mất điểm.**



*Đặng Trung Kiên* *Trang 31*

*Ngôn ngữ lập trình C++*



**BÀI 6. PHÉP TOÁN, BIỂU THỨC, CÂU LỆNH GÁN**

Để mô tả các thao tác trong thuật toán, mỗi ngôn ngữ lập trình đều xác định và sử dụng một số khái niệm cơ bản:

* Phép toán
* Biểu thức
* Gán giá trị cho biến

Dưới đây sẽ xét các khái niệm đó trong ngôn ngữ lập trình C++

***1. Phép toán***

Tương tự như trong Toán học, trong các ngôn ngữ lập trình đều có những phép toán số học như cộng, trừ, nhân, chia trên các đại lượng thực, các phép toán chia lấy nguyên và chia lấy phần dư, các phép toán quan hệ,….

Bảng dưới đây là kí hiệu các phép toán đó trong toán và trong C++



*Đặng Trung Kiên* *Trang 32*

*Ngôn ngữ lập trình C++*



|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Phép toán** | |  | **Toán học** |  |  | **C++** |  |  | **Ghi chú** |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | Cộng (+) |  |  | + |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | Trừ (-) |  | - | |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |
|  |  |  |  | Nhân (x) |  |  | \* |  |  |  |  |
|  | **Các phép toán** |  |  |  |  |  |  |  |  |  |  |
|  |  |  | Chia lấy |  | % | |  |  | 5%2=1 (5 chia 2 dư 1) |  |
|  | **số học đối với** |  |  | phần dư |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  | **số nguyên** |  |  | Chia lấy |  |  |  |  |  | 5/2=2(5 chia 2 được phần |  |
|  |  |  |  | phần nguyên |  |  | / |  |  | nguyên là 2). **Chú ý, phép** |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | **chia hai số nguyên thì kết** |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  | **quả là một số nguyên** |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | Cộng (+) |  | + | |  |  |  |  |
|  |  | |  |  |  |  | |  |  |  |  |
|  | **Các phép toán** | |  | Trừ (-) |  |  | - |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |
|  | **số học đối với** | |  | Nhân (x) |  | \* | |  |  |  |  |
|  | **số thực** | |  |  |  |  |  |  |  |  |  |
|  |  | Chia |  |  | / |  |  | 5.0/2.0=2.5 (phép chia hai số |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | thực kết quả là một số thực) |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |
|  |  |  |  | Nhỏ hơn (<) |  | < | |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |
|  |  |  |  | Lớn hơn (>) |  |  | > |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | Nhỏ hơn |  | <= | |  |  |  |  |
|  | **Các phép toán** | |  | hoặc bằng ≤ |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |
|  | **quan hệ** | |  | Lớn hơn |  |  | >= |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  | hoặc bằng ≥ |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |
|  |  |  |  | Bằng = |  | == | |  |  | **Hai dấu bằng** |  |
|  |  |  |  |  |  |  | |  |  |  |  |
|  |  |  |  | Khác ≠ |  |  | != |  |  |  |  |
|  |  | |  |  |  |  |  |  |  |  |  |
|  | **Các phép toán** | |  | Phủ định ¬ |  | ! | |  |  | Một dấu ! |  |
|  |  | |  |  |  |  | |  |  |  |  |
|  | **logic** | |  | Hoặc ∨ |  |  | || |  |  | Hai dấu | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | Và ∧ |  | && | |  |  | Hai dấu & |  |
|  |  | |  |  |  |  |  |  |  |  |  |
|  | **Chú ý:** | | |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  | *Đặng Trung Kiên* | | |  |  |  |  |  |  | *Trang 33* |  |

*Ngôn ngữ lập trình C++*



* + Kết quả của các phép toán quan hệ cho giá trị logic.
  + Một trong những ứng dụng của phép toán logic là để tạo ra các biểu thức phức tạp từ các quan hệ đơn giản

1. *Biểu thức số học*

Trong lập trình, biểu thức số học là một biến kiểu số hoặc một hằng số hoặc các biến kiểu số và các hằng số liên kết với nhau bởi một số hữu hạn phép toán số học, các dấu ngoặc tròn ( và ) tạo thành một biểu thức có dạng tương tự như cách viết trong toán học với những quy tắc sau:

* Chỉ dùng cặp ngoặc tròn để xác định trình tự thực hiện phép toán trong trường hợp cần thiết
* Viết lần lượt từ trái sang phải
* Không được bỏ qua dấu nhân (\*) trong tích
* Thực hiện các phép toán trong ngoặc trước
* Trong dãy các phép toán không chứa ngoặc thì thực hiện từ trái sang phải, theo thứ tự các phép toán nhân (\*), chia (/), lấy phần dư
  + thực hiện trước và các phép toán cộng (+), trừ (-) thực hiện sau

Ví dụ:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Biểu thức trong Toán học | | | | | | | | Biểu thức trong C++ |
|  |  |  |  |  |  |  |  |  |
|  | 5a+6b | | | | | | | 5\*a+6\*b |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  | *xy* | |  |  | x\*y/z |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  | *z* |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  | |  |  |  |  |  |
| Ax2 + *Bx* + *C* | | | | | | | | A\*x\*x+B\*x+C |
|  |  | | | | |  | |  |
|  | *x* + *y* | | | | | *x* − *z* | |  |
|  |  |  | | | − |  |  | (*x* + *y*)(*x* − 1/ 2) − ( *x* − *z*) / (*x* \* *y*) |
|  | *x* − | 1 | | | *xy* | |
|  | 2 | |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |



*Đặng Trung Kiên* *Trang 34*

*Ngôn ngữ lập trình C++*



**Chú ý:**

* + Nếu biểu thức chứa một hằng hay biến kiểu thực thì ta có biểu thức số học thực, giá trị của biểu thức cũng thuộc kiểu thực.
  + Trong một số trường hợp, nên dùng biến trung gian để có thể tránh được việc tính một biểu thức nhiều lần

1. *Hàm số học chuẩn*

Để lập trình được dễ dàng, thuận tiện hơn ,các ngôn ngữ lập trình đều có thư viện chứa một số chương trình tính giá trị những hàm toán học thường dùng. Các chương trình như vậy được gọi là các hàm số học chuẩn. Mỗi hàm chuẩn có tên chuẩn riêng

* Đối số của hàm là một hay nhiều biểu thức số học và được đặt trong cặp dấu ngoặc tròn ( và ) sau tên hàm. Riêng hàm main không có đối số trong cặp dấu ngoặc tròn ( và )
* Bản thân hàm chuẩn cũng được coi là một biểu thức số học và nó có thể tham gia vào biểu thức số học như một toán hạng (giống như biến và hằng).
* Kết quả của hàm có thể là nguyên hoặc thực hay phụ thuộc

và kiểu của đối số

Thư viện chứa các hàm số học chuẩn trong C++ là thư viện <math.h>. Do vậy, trước khi dùng các hàm này phải khai báo #include<math.h> Bảng dưới đây cho biết một số hàm chuẩn thường dùng:



*Đặng Trung Kiên* *Trang 35*

*Ngôn ngữ lập trình C++*



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Hàm** | **Tham số** | **Kiểu** | **Chức năng** | **Ví dụ** | |
|  | **trả về** |
|  |  |  |  |  |  |
|  |  |  |  |  |  | |
|  | sqrt(x) | x là số thực | Số thực | Tính căn bậc hai của x | sqrt(100)=10.0 | |
|  |  | không âm |  | (viết tắt của square |  |  |
|  |  | hoặc nguyên |  | root, root là gốc rễ, là |  |  |
|  |  | không âm |  | căn, square là hình |  |  |
|  |  |  |  | vuông, là bình |  |  |
|  |  |  |  | phương) |  |  |
|  |  |  |  |  |  | |
|  | fabs(x) | x là số thực | Số thực | Tính giá trị tuyệt đối | fabs(-3)=3 | |
|  |  |  |  | của x, viết tắt của |  |  |
|  |  |  |  | float absolute (giá trị |  |  |
|  |  |  |  | tuyệt đối số thực) |  |  |
|  |  |  |  |  |  |  |
|  | sin(x) | x là số thực | Số thực | Tính sinx |  |  |
|  |  |  |  |  |  |  |
|  | cos(x) | x là số thực | Số thực | Tính cosx |  |  |
|  |  |  |  |  |  | |
|  | trunc(x) | x là số thực | Số thực | Viết tắt truncate (chặt | trunc(2.3)=2.0 | |
|  |  |  |  | bớt phần thập phân). | trunc(3.8)=3.0 | |
|  |  |  |  |  | trunc(5.5)=5.0 | |
|  |  |  |  |  | trunc(-2.3)=-2.0 | |
|  |  |  |  |  | trunc(-3.8)=-3 | |
|  |  |  |  |  | trunc(-5.5)=5 | |
|  |  |  |  |  |  | |
|  | floor(x) | Số thực | Số thực | Hàm làm tròn số | trunc(2.3)=2.0 | |
|  |  |  |  | xuống | trunc(3.8)=3.0 | |
|  |  |  |  |  | trunc(5.5)=5.0 | |
|  |  |  |  |  | trunc(-2.3)=-3.0 | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  | *Đặng Trung Kiên* | |  |  | *Trang 36* | |

*Ngôn ngữ lập trình C++*



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  | trunc(-3.8)=-4 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  | trunc(-5.5)=-6 | | | |
|  |  |  |  |  |  |  |  |  |  |  | |  |  | |  |
|  | ceil(x) |  | Số thực | |  | Số thực | |  | Hàm làm tròn số lên | | | trunc(2.3)=3.0 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  | trunc(3.8)=4.0 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  | trunc(5.5)=6.0 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  | trunc(-2.3)=-2.0 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  | trunc(-3.8)=-3.0 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  | trunc(-5.5)=-5.0 | | | |
|  |  |  |  |  |  |  |  |  |  |  | |  |  | |  |
|  | round(x) |  | Số thực | |  | Số thực | |  | Hàm làm tròn số | | | trunc(2.3)=2.0 | | | |
|  |  |  |  |  |  |  |  |  | thông thường như | | | trunc(3.8)=4.0 | | | |
|  |  |  |  |  |  |  |  |  | Toán học. | |  | trunc(5.5)=6.0 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  | trunc(-2.3)=-2.0 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  | trunc(-3.8)=-4.0 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  | trunc(-5.5)=-6.0 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | **round** | |  |  |  |  |  |  |  |  |  |
|  |  |  |  | **(làm tròn** | | |  |  | **floor** |  | **ceil** |  | **Trunc** | |  |
|  | **Value** | |  | **nhƣ toán** | | | **(làm tròn** | | |  | **(làm tròn** |  | **(cắt phần** | |  |
|  |  |  |  |  | **thông** | |  | **xuống)** | |  | **lên)** |  | **thập phân)** | |  |
|  |  |  |  | **thƣờng)** | | |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |
|  | 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 |  |  |
|  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |
|  | *Đặng Trung Kiên* | | | |  |  |  |  |  |  |  |  | *Trang 37* | | |

*Ngôn ngữ lập trình C++*



|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -3.8 | | |  | -4.0 | | -4.0 | | -3.0 |  | -3.0 |
|  |  |  |  |  |  |  |  |  |  |  |
| -5.5 | | |  | -6.0 | | -6.0 | | -5.0 |  | -5.0 |
|  |  |  |  |  |  |  |  |  |  |  |
| Ví dụ: | | | |  |  |  |  |  |  |  |
|  |  |  | |  |  |  |  |  |  |  |
| Toán học | | | |  |  |  | C++ |  |  |  |
|  |  | | |  |  |  |  | | |  |
|  |  |  |  |  |  |  | (-b+sqrt(b\*b-4\*a\*c))/(2\*a) | | |  |
|  | −*b* + *b*2−4*ac* | | |  |  |  |
|  | 2*a* | | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |



**4. Biểu thức quan hệ**

Hai biểu thức cùng kiểu liên kết với nhau bởi phép toán quan hệ cho ta một biểu thức quan hệ.

***Biểu thức quan hệ có dạng:***

***<biểu thức 1> <phép toán quan hệ> <biểu thức 2>* Trong đó, biểu thức 1 và biểu thức 2 cùng là xâu hoặc cùng là biểu thức số học**

Ví dụ

x<5

i+1>=2\*j

Biểu thức quan hệ được thực hiện theo trình tự:

* Tính giá trị các biểu thức
* Thực hiện các phép toán quan hệ

Kết quả của biểu thức quan hệ là giá trị logic: true (đúng), hoặc false(sai). Trong ví dụ trên, nếu x có giá trị 3 thì biểu thức x<5 cho giá trị true. Nếu i có giá trị 2 và j có giá trị 3 thì biểu thức i+1>=2\*j sẽ cho giá trị false. Ví dụ

Điều kiện để điểm M có tọa độ (x,y) thuộc hình tròn tâm I(a,b) bán kính R là: sqrt((x-a)\*(x-a)+(y-b)\*(y-b))<=R Hoặc (x-a)\*(x-a)+(y-b)\*(y-b)<=R\*R



*Đặng Trung Kiên* *Trang 38*

*Ngôn ngữ lập trình C++*



***5. Biểu thức logic***

**Biểu thức logic đơn giản là biến logic hoặc hằng logic**

Biểu thức logic là các biểu thức logic đơn giản, các biểu thức quan hệ liên kết với nhau bởi các phép toán logic. Giá trị biểu thức logic là true hoặc false. Các biểu thức quan hệ thường đặt trong cặp ngoặc ( và ) Phép toán ! được viết trước biểu thức cần phủ định, ví dụ:

!(x<1) thể hiện phát biểu ―x không nhỏ hơn 1‖ và điều này tương đương với biểu thức quan hệ x>=1

Các phép toán và (&&) và hoặc (||) trong C++ dùng để kết hợp nhiều biểu thức logic hoặc quan hệ thành một biểu thức, thường được dùng để diễn tả các điều kiện phức tạp Ví dụ 1

Để thể hiện điều kiện 5 ≤ *x* ≤11, trong C++ cần phải tách phát biểu dưới dạng ― 5 ≤ *x* và *x* ≤11 ―

(5<=x) && (x<=11)

Ví dụ 2

Giả thiết M và N là hai biến nguyên. Điều kiện xác định M và N đồng thời chia hết cho 3 hay đồng thời không chia hết cho 3 được thể hiện trong C++ như sau:

((*M* %3 == 0) & &(*N*%3 == 0)) || ((*M* %3! = 0) & &(*N*%3! = 0))

*6. Câu lệnh gán*

Lệnh gán là một trong những lệnh cơ bản nhất của các ngôn ngữ lập trình

***Trong C++, câu lệnh gán có dạng:***

***<tên biến>=<biểu thức>;***

Trong trường hợp đơn gỉn, tên biến là tên của biến đơn. Kiểu của giá trị biểu thức phải phù hợp với kiểu của biến.

Chức năng của lệnh gán là đặt cho biến có tên ở vế trái dấu ‗=‘ giá trị mới bằng giá trị của biểu thức ở vế phải



*Đặng Trung Kiên* *Trang 39*

*Ngôn ngữ lập trình C++*



Ví dụ

x1=(-b-sqrt(b\*b-4\*a\*c))/(2\*a);

x2=-b/a-x1;

z=z-1;

i=i+1;

Trong ví dụ trên, ý nghĩa của lệnh gán thứ ba là giảm trị của biến z một đơn vị. Ý nghĩa của lệnh gán thứ tư là tăng giá trị của biến i lên một đơn vị.



*Đặng Trung Kiên* *Trang 40*

*Ngôn ngữ lập trình C++*



**BÀI 7. CÁC HÀM CHUẨN VÀO/RA ĐƠN GIẢN**

Để khởi tạo giá trị ban đầu cho biến, ta có thể dùng lệnh gán để gán một giá trị cho biến. Như vậy, mỗi chương trình luôn làm việc với một bộ dữ liệu vào. Để chương trình có thể làm việc với nhiều bộ dữ liệu vào khác nhau, thư viện của các ngôn ngữ lập trình cung cấp một số chương trình dùng để đưa dữ liệu vào và đưa dữ liệu ra.

Những chương trình đưa dữ liệu vào cho phép đưa dữ liệu từ bàn phím hoặc từ đĩa vào gán cho các biến, làm cho chương trình trở nên linh hoạt, có thể tính toán với nhiều bộ dữ liệu đầu vào khác nhau. Kết quả tính toán được lưu trữ tạm thời trong bộ nhớ. Những chương trình đưa dữ liệu ra dùng để đưa các kết quả này ra màn hình, in ra giấy hoặc lưu trên đĩa.

Các chương trình đưa dữ liệu vào và ra đó được gọi chung là các hàm chuẩn vào/ra đơn giản.

Trong phần này, ta sẽ xét các hàm chuẩn vào/ra đơn giản của C++ để nhập dữ liệu vào từ bàn phím và đưa thông tin ra màn hình.

***1. Nhập dữ liệu vào từ bàn phím***

Việc nhập dữ liệu từ bàn phím được thực hiện bằng hàm chuẩn:

***Cin>>biến vào>>biến vào>>biến vào>>….>>biến vào;***

Trong đó biến vào là biến đơn. Dấu …. Thể hiện có thể có nhiều

biến vào, các biến vào được cách nhau bởi toán tử >> Ví dụ

cin>>n;

cin>>a>>b>>c;

Lệnh thứ nhất để nhập một giá trị từ bàn phím và gán giá trị đó cho biến n. Lệnh thứ hai dùng để nhập lần lượt ba giá trị từ bàn phím và gán các giá trị đó tương ứng cho ba biến a,b và c.



*Đặng Trung Kiên* *Trang 41*

*Ngôn ngữ lập trình C++*



Khi nhập giá trị cho nhiều biến, những giá trị này được gõ cách nhau bởi ít nhất một dấu cách hoặc ký tự xuống dòng (nhấn phím Enter). Các giá trị tương ứng với biến nguyên phải được biểu diễn dưới dạng số nguyên (không có dấu chấm thập phân). Các giá trị ứng với biến thực có thể dược nhập dưới dạng số nguyên, số thực dạng thông thường hoặc số thực dạng dấu phẩy động.

Ví dụ, để nhập các giá trị 1, -5 và 6 cho các biến thực a,b,c trong hàm thứ hai trong ví dụ trên, có thể gõ:

1 -5 6 rồi nhấn phím Enter

Hoặc 1.0 -5 rồi nhấn phím Enter

6 rồi nhấn phím Enter

***2. Đưa dữ liệu ra màn hình***

Để đưa dữ liệu ra màn hình, C++ cung cấp hàm chuẩn:

***Cout<<kết quả ra<<kết quả ra<<….<<kết quả ra;***

Trong đó, kết qủa ra có thể là tên biến đơn, biểu thức hoặc hằng. Các hằng xâu thường được dùng để tách các kết quả ra hoặc đưa ra chú thích. Dấu … thể hiện có thể đưa nhiều kết quả ra ra màn hình. Nếu đưa nhiều kết quả ra màn hình, chúng phải cách nhau bởi dấu <<

Với cout, sau khi đưa các kết quả ra màn hình, con trỏ không di chuyển xuống dòng tiếp theo. Câu lệnh sau sẽ khiến con trỏ sẽ chuyển xuống đầu dòng tiếp theo

***Cout<<kết quả ra<<kết quả ra<<….<<kết quả ra<<endl;***

***Cout<<kết quả ra<<kết quả ra<<….<<kết quả ra<<’\n’;***

Ví dụ:

Để nhập giá trị cho biến M từ bàn phím, người ta thường dùng cặp hàm:

cout<<‖Hay nhap gia tri M‖;

cin>>M;

Khi thực hiện các lệnh này, trên màn hình xuất hiện dòng thông báo:



*Đặng Trung Kiên* *Trang 42*

*Ngôn ngữ lập trình C++*



Hay nhap gia tri M:

Và con trỏ sẽ ở vị trí tiếp theo trên dòng, chờ ta gõ giá trị của M.

Để chương trình được sử dụng một cách tiện lợi, khi nhập giá trị từ bàn phím cho biến, ta nên có thêm xâu kí tự nhắc nhở việc nhập giá trị cho biến nào, kiểu dữ liệu gì,…. Ví dụ, khi cần nhập số nguyên dương N(N<=100) từ bàn phím ta có thể sử dụng cặp hàm sau:

cout<<”Nhap so nguyen duong N<=100”; cin>>N;

Sau đây là một chương trình hoàn chỉnh có sử dụng các hàm chuẩn vào và ra

Ví dụ: Nhập vào hai số thực x và y. Hãy tính diện tích của hình chữ nhật có hai cạnh là x và y

#include <iostream>

using namespace std;

double x,y;

int main()

{

cout <<"Nhap vao gia tri thuc x va y"<<endl; cin>>x>>y;

cout<<"Dien tich cua hinh chu nhat la"<<x\*y; return 0;

}

**Quy cách in ra màn hình**

**Xét ví dụ sau:**

#include <iostream>

#include<iomanip>

using namespace std;

int main () {

double a = 3.1415926534;



*Đặng Trung Kiên* *Trang 43*

*Ngôn ngữ lập trình C++*



double b = 206.0;

double c = 1.0e-10;

cout<<fixed<<setprecision(8);//chú ý

cout << a << '\n' << b << '\n' << c << '\n'; return 0;

}

Sau khi chạy chương trình, màn hình sẽ hiển thị:

3.1415926534

206.00000000

0.00000000

**Giải thích:**

Setprecision nằm trong thư viện <iomanip>, nghĩa là đặt số chữ số xuất hiện sau phần thập phân. Ở đây đặt số chữ số xuất hiện sau phần thập phân là 8

fixed nghĩa là cố định, tức là chữ số hiển thị phần thập phân phải đảm bảo đủ 8 chữ số, nếu không đủ phải thêm 0

Nếu không có câu lệnh *//chú ý* thì kết quả hiển thị trên màn hình sẽ mặc định là:

3.14159

206

1e-010



*Đặng Trung Kiên* *Trang 44*

*Ngôn ngữ lập trình C++*



**BÀI 8. SOẠN THẢO, DỊCH, THỰC HIỆN VÀ HIỆU CHỈNH CHƢƠNG TRÌNH**

Để có thể thực hiện chương trình được viết bằng một ngôn ngữ lập trình, ta cần soạn thảo, sử dụng chương trình dịch để dịch chương trình đó sang ngôn ngữ máy. Các hệ thống lập trình cụ thể thường cung cấp phần mềm phục vụ cho việc soạn thảo, dịch và hiệu chỉnh chương trình

Với ngôn ngữ lập trình C++, người ta thường dùng phần mềm CodeBlocks với phiên bản hiện thời là 16.01, hoặc phần mềm Dev C++. Trong các kì thi học sinh giỏi vùng đồng bằng duyên hải, người ta thường dùng phần mềm CodeBlocks.

* + đây, tôi giới thiệu các em phần mềm CodeBlocks

1. **Hƣớng dẫn học sinh tải phần mềm Codeblocks**

**Bƣớc 1. Các em gõ chữ codeblock trên google để tìm trang chủ của codeblock, hoặc các em vào luôn trang chủ:** [**http://www.codeblocks.org/**](http://www.codeblocks.org/)

**Bƣớc 2. Các em click chỗ Downloads**

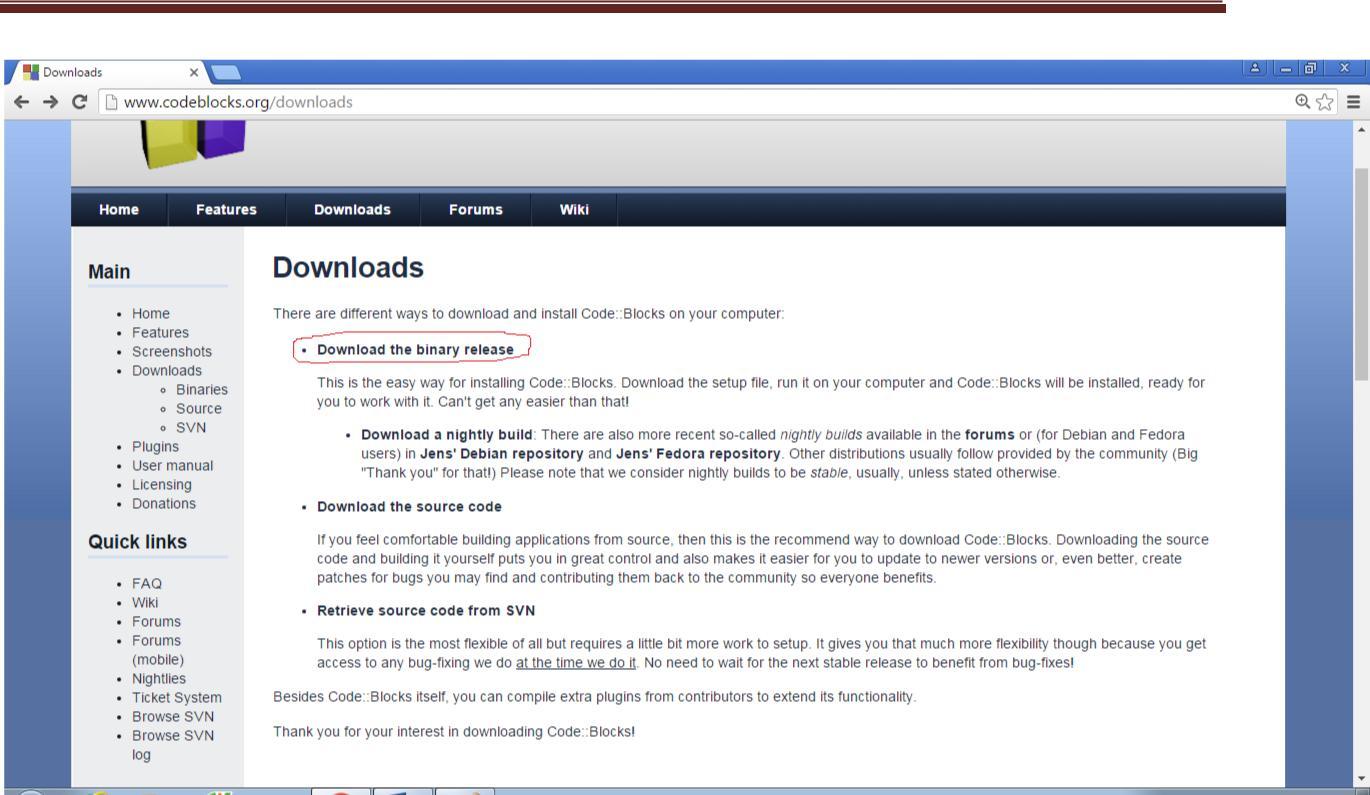


**Bƣớc 3. Các em vào chỗ Download the binary realase**



*Đặng Trung Kiên* *Trang 45*

*Ngôn ngữ lập trình C++*

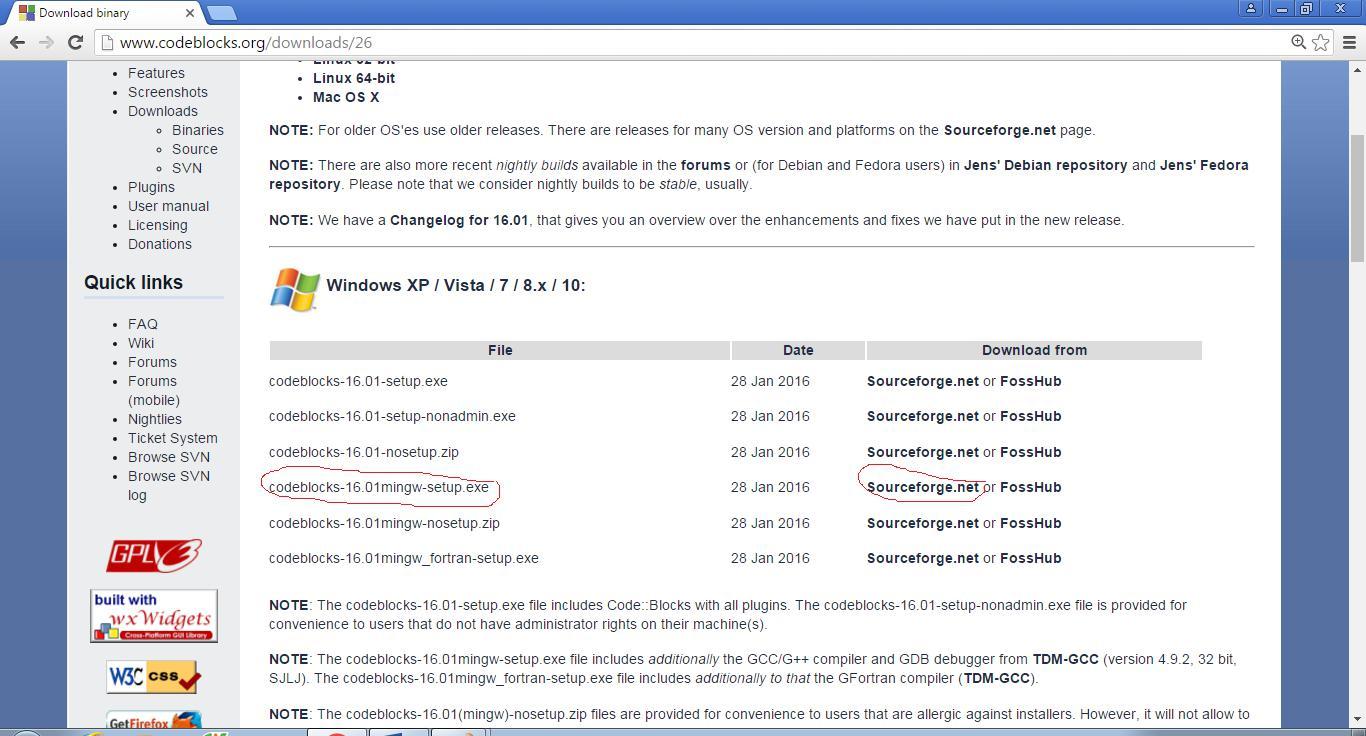


**Bƣớc 4. Các em click vào link sourceforge.net, trên dòng của phần**

**mềm**

**Codeblocks-16.01 mingw-setup.exe**

**(chú ý là chỉ tải phiên bản có chữ mingw này thôi)**



*Đặng Trung Kiên* *Trang 46*

*Ngôn ngữ lập trình C++*

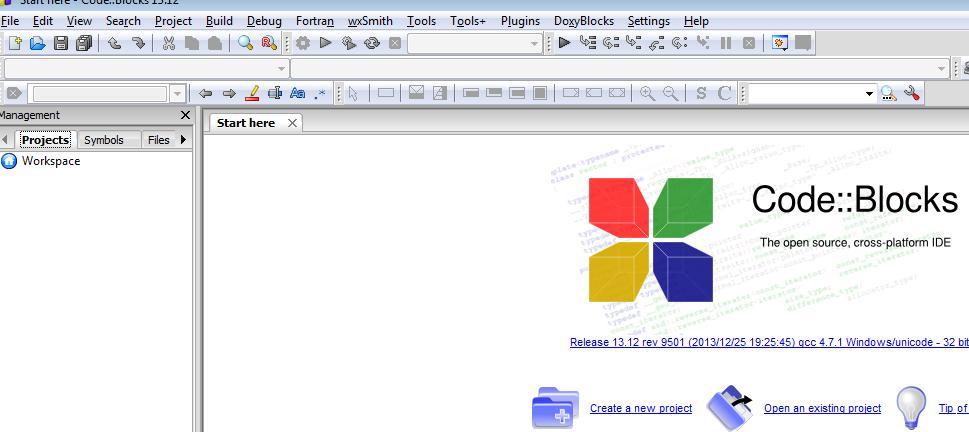


**Bƣớc 5. Tiến hành cài đặt**

Mở phần mềm lên, vào Run->next->I agree->Next->install (cài đặt), cài

xong thì Ok!!

1. **Giao diện Codeblock**

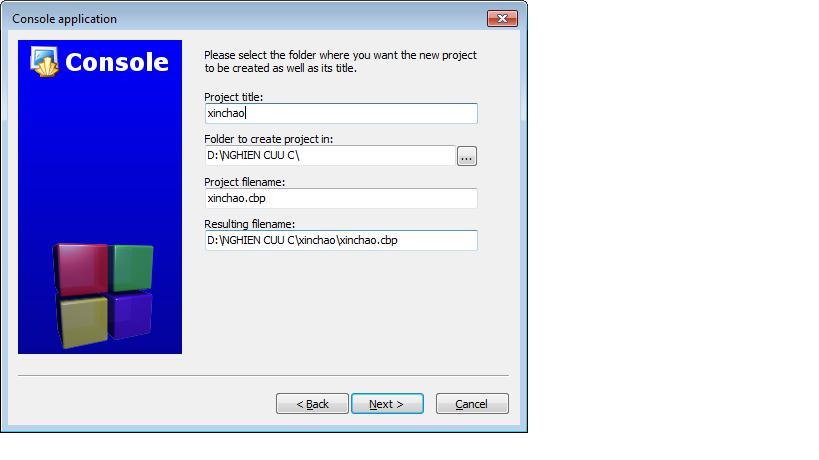


**Thanh trên cùng là thanh bảng chọn.**

**3. Hƣớng dẫn tạo một file C++**

**Bƣớc 1. Vào File->New->Project->Console Application->Next->C++**

**Một thanh bảng chọn hiện ra:**



Tại ô Folder to create project in: bạn chọn đường dẫn chứa thư mục của dự án



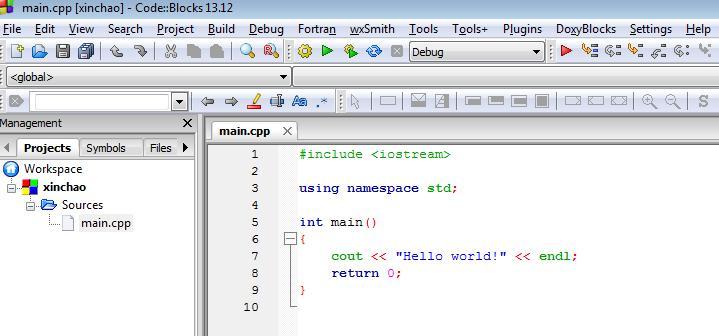
*Đặng Trung Kiên* *Trang 47*

*Ngôn ngữ lập trình C++*



Tại ô Project Title: ta đặt tên cho file dự án

File C++ sẽ có đuôi là .cpp, và mặc định là main.cpp



***Hình 1. Giao diện vùng lập trình***

Dưới đây là một số thao tác và phím tắt thường sử dụng để soạn thảo và thực hiện một chương trình viết bằng ngôn ngữ lập trình C++

* Soạn thảo: Gõ nội dung của chương trình gồm phần khai báo và các lệnh trong thân chương trình. Về cơ bản, việc soạn thảo chương trình tương tự như soạn thảo văn bản. Lưu chương trình bằng cách nhấn phím Ctrl+S (phần mở rộng của tệp ngầm định là

.cpp)

* Biên dịch chương trình: Nhấn tổ hợp phím Ctrl+F9 nếu muốn biên dịch chương trình. Nếu muốn biên dịch và chạy chương trình luôn thì nhấn tổ hợp phím F9. Nếu chương trình có lỗi cú pháp, phần mềm sẽ hiển thị một bảng thông báo. Cần phải sửa lỗi nếu có, lưu lại chương trình rồi tiến hành biên dịch lại cho tới khi không còn lỗi.
* Chạy chương trình F9
* Đóng cửa số chương trình: Alt+F9 hoặc Alt+F3
* Thoát khỏi phần mềm: Nhấn Alt+F4



*Đặng Trung Kiên* *Trang 48*

*Ngôn ngữ lập trình C++*



**CHƢƠNG III. CẤU TRÚC RẼ NHÁNH VÀ CẤU TRÚC LẶP**

**BÀI 9. CẤU TRÚC RẼ NHÁNH**

***1. Rẽ nhánh***

Thường ngày, có rất nhiều việc chỉ được thực hiện khi một điều kiện cụ thể nào đó được thỏa mãn.

Ví dụ, Châu và Ngọc thường cùng nhau chuẩn bị các bài thực hành Tin học

Một lần Châu hẹn với Ngọc: ―Chiều mai nếu trời không mưa thì Châu sẽ tới nhà Ngọc‖

Một lần khác, Ngọc nói với Châu: ―Chiều mai nếu trời không mưa thì Ngọc sẽ đến nhà Châu, nếu mưa thì sẽ gọi điện cho Châu để trao đổi‖

Câu nói của Châu cho ta biết một việc làm cụ thể (Châu đến nhà Ngọc) sẽ được thực hiện nếu một điều kiện cụ thể (trời không mưa) thỏa mãn. Ngoài ra không đề cập đến việc gì sẽ xảy ra nếu điều kiện đó không thỏa mãn (trời mưa).

Ta nói cách diễn đạt như vậy thuộc dạng thiếu:

Nếu….thì…..

Câu nói của Ngọc khẳng định một trong hai việc cụ thể (Ngọc đến nhà Châu hay Ngọc gọi điện cho Châu) chắc chắn sẽ xảy ra. Tuy nhiên, việc nào đó trong hai việc sẽ được thực hiện tùy thuộc vào điều kiện cụ thể (trời không mưa) thỏa mãn hay không

Ta nói cách diễn đạt như vậy thuộc dạng đủ:

Nếu….thì….,nếu không thì……

Từ đó có thể thấy, trong nhiều thuật toán, các thao tác tiếp theo sẽ

phụ thuộc vào kết quả nhận được từ các bước trước đó.

Cấu trúc dùng để mô tả mệnh đề có dạng như trên được gọi là cấu

trúc rẽ nhánh thiếu và đủ

Ví dụ, để giải phương trình bậc hai



*Đặng Trung Kiên* *Trang 49*

*Ngôn ngữ lập trình C++*



ax2 + *bx* + *c* = 0(*a* ≠ 0)

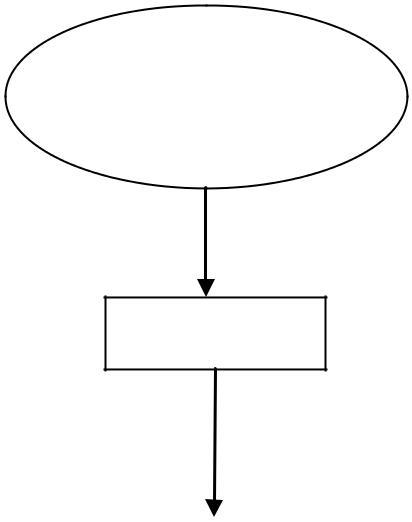
Trước tiên, ta tính biệt số *delta* = *b* \**b* − 4\* *a* \* *c*

Nếu D không âm, ta sẽ đưa ra các nghiệm. Trong trường hợp ngược lại, ta phải thông báo là phương trình vô nghiệm

Như vậy, sau khi tính D, tùy thuộc vào giá trị của D, một trong hai thao tác được thực hiện

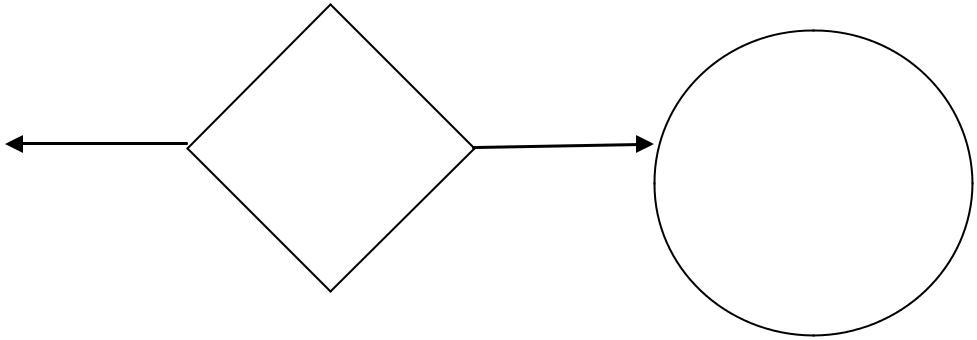
Mọi ngôn ngữ lập trình đều có các câu lệnh để mô tả cấu trúc rẽ

nhánh

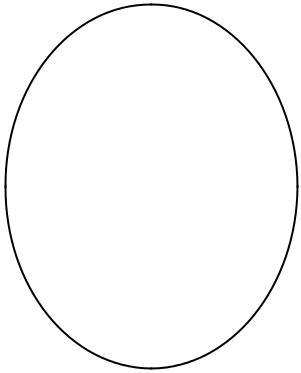


Nhập a,b,c

D<-b2-4ac



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Sai |  | Đúng |  |
| Thông | D≥0? |  | Tính, đưa |
|  |
|  |  |
| báo vô |  |  |  | ra nghiệm |
| nghiệm |  |  |  | thực, rồi |
| rồi kết |  |  |  | kết thúc |
| thúc |  |  |  |  |



***Hình 1. Sơ đồ thể hiện cấu trúc rẽ nhánh***



*Đặng Trung Kiên* *Trang 50*

*Ngôn ngữ lập trình C++*



***2. Câu lệnh if***

Để mô tả cấu trúc rẽ nhánh, C++ dùng câu lệnh if. Tương ứng với hai dạng thiếu và đủ ở trên. C++ có hai dạng câu lệnh if

***a) Dạng thiếu***

if (điều kiện) {các câu lệnh;};

Lưu ý: Điều kiện bắt buộc phải nằm trong dấu ( và )

***b) Dạng đủ***

if (<điều kiện>) {các câu lệnh1;}; else {các câu lệnh2;} Lưu ý: Điều kiện bắt buộc phải nằm trong dấu ( và ) Trong đó:

* + Điều kiện là biểu thức logic
  + Các câu lệnh ở đây có thể gồm một hoặc nhiều câu lệnh của ngôn ngữ lập trình C++ (nghĩa là có thể có câu lệnh if bên trong câu lệnh if)
* dạng thiếu, điều kiện sẽ được tính và kiểm tra. Nếu điều kiện đúng (có giá trị true) thì câu lệnh sẽ được thực hiện, ngược lại thì câu lệnh sẽ bị bỏ qua
* dạng đủ, điều kiện cũng sẽ được tính và kiểm tra. Nếu điều kiện đúng thì câu lệnh 1 sẽ được thực hiện, nếu ngược lại thì câu lệnh 2 sẽ được thực hiện

Ví dụ 1.

if (d<0) cout<<‖Phuong trinh vo nghiem‖;

if (a%3==0) cout<<‖a chia het cho 3‖ ; else cout<<‖a khong chia het cho 3‖;

Ví dụ 2. Để tìm số lớn nhất max trong hai số a và b, có thể thực hiện theo cách sau:

* Dùng câu lệnh gán max=a, và lệnh if dạng thiếu: if (b>a) max=b;
* Dùng một lệnh if dạng đủ if(b>a) max=b else max=a;

**3. Câu lệnh ghép**

Ngôn ngữ lập trình cho phép gộp một dãy câu lệnh thành một câu lệnh ghép (hay câu lệnh hợp thành). Câu lệnh hợp thành trong ngôn ngữ lập trình C++ có dạng:

{ câu lệnh; Câu lệnh;

…………

Câu lệnh;

}

Các câu lệnh nằm trong cặp dấu mở đầu và kết thúc

{}



*Đặng Trung Kiên* *Trang 51*

*Ngôn ngữ lập trình C++*



**Một ví dụ để thấy sự khác nhau giữa câu lệnh đơn và câu lệnh ghép:** Giả sử x=3. Nếu x chia cho 2 dư 0 thì kết luận x là số chẵn và x chia hết cho 5

Xét chương trình do bạn A viết

#include <iostream>//1

using namespace std;//2

int main()//3

{

long x=3;//4

if(x%2==0) //5

{cout<<"x la so chan";//6 cout<<"x chia het cho 5"//7 }//8

return 0;

}

Xét chương trình do bạn B viết

#include <iostream>//1

using namespace std;//2

int main()//3

{

long x=3;//4

if(x%2==0) //5

cout<<"x la so chan";//6 cout<<"x chia het cho 5"//7

return 0;

}

* Ở đây chương trình của bạn A viết đúng, chương trình của bạn B viết sai. Chương trình của bạn B sẽ đưa ra màn hình dòng chữ ―x chia het cho 5‖, trong khi 3 không chia hết cho 5. Sở dĩ B viết sai vì sau khi kiểm tra 3 không chia hết cho 2, chương trình không thực hiện câu lệnh 6, chương trình tiếp tục thực hiện câu lệnh 7
* Chương trình của bạn A viết đúng, vì sau khi kiểm tra 3 không chia hết cho 2, chương trình không thực hiện câu lệnh 6 và câu lệnh 7, vì lúc này câu lệnh 6, và câu lệnh 7 nằm trong cặp dấu {}, nên thực chất chúng hợp lại thành một câu lệnh ghép, câu lệnh ghép này không được thực hiện vì 3 không chia hết cho 2.



*Đặng Trung Kiên* *Trang 52*

*Ngôn ngữ lập trình C++*



***4. Một số ví dụ***

**Ví dụ 1. Tìm nghiệm thực của phƣơng trình bậc hai**

ax2 + *bx* + *c* = 0(*a* ≠ 0)

Input: các hệ số a,b,c nhập từ bàn phím

Ouput: Đưa ra màn hình các nghiệm thực hoặc thông báo: ―Phuong trinh vo nghiem‖

#include <iostream>

#include<math.h>

#include<iomanip>

using namespace std;

double a,b,c,d,x1,x2;

int main()

{

cin>>a>>b>>c;

d=b\*b-4\*a\*c;

if(d<0) cout<<"Phuong trinh vo nghiem";

else

{

x1=(-b-sqrt(d))/(2\*a); x2=(-b+sqrt(d))/(2\*a); cout<<fixed<<setprecision(3); cout<<"x1="<<x1<<" x2="<<x2;

}

return 0;

}

Ví dụ 2. Tìm số ngày của năm N, biết rằng năm nhuận là năm chia hết cho 400 hoặc chia hết cho 4 nhưng không chia hết cho 100. Ví dụ, các năm 2000, 2004 là năm nhuận có số ngày là 366, các năm 1900, 1945 không phải là năm nhuận có số ngày là 365. Input: N nhập từ bàn phím

Ouput: Đưa số ngày của năm N ra màn hình

#include <iostream>

#include<math.h>

#include<iomanip>

using namespace std;

long N,sn;

int main()

{

cout<<"Nam";

cin>>N;



*Đặng Trung Kiên* *Trang 53*

*Ngôn ngữ lập trình C++*



if((N%400==0)||(N%4==0)||(N%100!=0))

sn=366;else sn=365;

cout<<"so ngay cua nam "<<N<<"la"<<sn; return 0;

}



*Đặng Trung Kiên* *Trang 54*

*Ngôn ngữ lập trình C++*



***5. Bài tập***

Bài 1. Lập trình nhập từ bàn phím ba số nguyên a,b,c (0 ≤ *a*,*b*,*c* ≤1018 ) .

Yêu cầu đưa ra màn hình giá trị lớn nhất trong ba số đó.

Bài 2. Lập trình nhập từ bàn phím bốn số thực a,b,c, d. Yêu cầu đưa ra màn hình giá trị nhỏ nhất và giá trị lớn nhất của các số đó

Bài 3. Lập trình nhập từ bàn phím ba số nguyên m,n, k (0 ≤ *a*,*b*,*c* ≤106 ) .

Tính và đưa ra màn hình bình phương các số không âm và lập phương của các số âm.

Bài 4. Cho ABCD là tứ giác lồi có tọa độ các đỉnh A,B,C,D là các cặp số thực tương ứng *A*(*x*1 , *y*1 ), *B*(*x*2 , *y*2 ),*C*(*x*3 , *y*3 ), *D*(*x*4 , *y*4 ) . Hãy lập trình:

* Nhập từ bàn phím tọa độ các đỉnh của tứ giác
* Tính và đưa ra màn hình độ dài các đường chéo

Bài 5. Một dãy số được gọi là cấp số cộng nếu phần tử sau bằng phần tử trước cộng với một số d không đổi. Ví dụ 1 3 5 7 là một cấp số cộng vì 1+2=3, 3+2=5, 5+2=7. Một dãy số được gọi là cấp số nhân nếu phần tử sau bằng phần tử trước nhân với một số d không đổi. Ví dụ 2 6 18 là một cấp số nhân vì 6=2.3, 18=6.3

Cho ba số nguyên *p*, *q*, *r*(0 ≤ *p*, *q*, *r* ≤1018 ) . Kiểm tra xem ba số này, theo

thứ tự nhập vào, có tạo thành một cấp số nhân hay không. Viết chương trình thực hiện yêu cầu trên.

Bài 6. Viết đoạn chương trình chứa các câu lệnh IF tính:

|  |  |
| --- | --- |
| ⎧*x* + *y*, *neuá x* >1*vaøy* >1 | |
| ⎪ |  |
| ⎪*x* − *y*, *neáu x* <1*vaøy* ≤1 | |
| *v* = ⎨ | > 0 |
| ⎪− *x* + *y*, *neáu x* ≤1*vaøy* |
| ⎪ | ≤ 0 |
| ⎩− *x* − *y*, *neuá x* ≤1*vaøy* |

**Câu hỏi gợi mở tƣ duy:**

* Bài 1,2,3 có rất nhiều cách để giải chúng trong trường hợp có vài phần tử? Em thử nghĩ xem có những cách nào?
* Bài 1,2,3 nếu trong trường hợp là một dãy gồm nhiều số, thuật toán tổng quát của nó là gì? Tham khảo bài Mảng, bài lặp để tìm ra câu trả lời
* Bài 5 liệu có đơn giản như vậy, trong trường hợp là một dãy gồm nhiều số? Tham khảo bài Cấu trúc mảng, Lặp để tìm ra câu trả lời cho một thuật toán mới



*Đặng Trung Kiên* *Trang 55*

*Ngôn ngữ lập trình C++*



**BÀI 10. CẤU TRÚC LẶP**

***1. Lặp***

Với a là số nguyên và a>2, xét các bài toán sau đây:

Bài toán 1. Tính và đưa kết quả ra màn hình tổng:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *S* = | 1 | + | 1 |  | + ..... | 1 |
| *a* | *a* +1 | | *a* +100 |
|  |  |  |

Bài toán 2. Tính và đưa kết quả ra màn hình tổng:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *S* = | 1 | + | 1 |  | + ..... | 1 | | + ... |
| *a* | *a* +1 | | *a* + *N* | |
|  |  |  |  |
| Cho đến khi | | | | | 1 |  | < 0.0001 thì dừng lại | |
|  |  |
|  |  |  |  |  | *a* + *N* | | |  |

Với cả hai bài toán, dễ thấy cách để tính tổng S có nhiều điểm tương tự

* Xuất phát, S được gán giá trị 1*a*



|  |  |  |  |
| --- | --- | --- | --- |
| ∙ | Tiếp theo, cộng vào tổng S một giá trị | 1 | với N=1,2,3,4,5,…. |
|  |
| *a* + *N* |

Việc cộng này được lặp lại một số lần.

Đối với bài toán 1, số lần lặp là 100 và việc cộng vào tổng S sẽ kết thúc khi đã thực hiện việc cộng 100 lần

Đối với bài toán 2, số lần lặp chưa biết trước, nhưng việc cộng vào

|  |  |  |
| --- | --- | --- |
| tổng S sẽ kết thúc khi điều kiện | 1 | < 0.0001 được thỏa mãn |
| *a* + *N* |
|  |  |

Nói chung, trong một số thuật toán có những thao tác phải thực hiện lặp đi lặp lại một số lần. Một trong các đặc trưng của máy tính là có khả năng thực hiện hiệu quả các thao tác lặp. Cấu trúc lặp mô tả thao tác lặp và có hai dạng là lặp với số lần biết trước và lặp với số lần chưa biết trước.

Các ngôn ngữ lập trình đều có các câu lệnh để mô tả cấu trúc lặp.



*Đặng Trung Kiên* *Trang 56*

*Ngôn ngữ lập trình C++*



***2. Lặp với số lần biết trước và câu lệnh for***

Có hai thuật toán Tong\_1a và Tong\_1b để giải bài toán 1 như sau:

Thuật toán Tong\_1a

Bước 1. *S* ← 1/ *a*; *N* ← 0 {Khởi tạo S và N} Bước 2. *N* ← *N* +1

Bước 3. Nếu N>100 thì chuyển đến bước 5 Bước 4. *S* ← *S* + 1/ (*a* + *N* ) rồi quay lại bước 2

Bước 5. Đưa S ra màn hình rồi kết thúc.

Thuật toán Tong\_1b

Bước 1. *S* ← 1/ *a*; *N* ←101 {Khởi tạo s và N}

Bước 2. *N* ← *N* −1

Bước 3. Nếu N<2 thì chuyển đến bước 5

Bước 4. *S* ← *S* + 1/ (*a* + *N* ) rồi quay lại bước 2.

Bước 5. Đưa S ra màn hình rồi kết thúc.

Lưu ý, số lần lặp của cả hai thuật toán trên là biết trước và như nhau (100 lần)

Trong thuật toán Tong\_1a, giá trị N khi bắt đầu tham gia vòng lặp là 1 và sau mỗi lần lặp N tăng lên 1 cho đến khi N>100 (N=101) thì kết thúc vòng lặp (thực hiện đủ 100 lần). Trong thuật toán Tong\_1b, giá trị N bắt đầu tham gia vòng lặp là 100 và sau mỗi lần lặp N giảm đi 1 cho đến khi N<1 (N=0) thì kết thúc lặp (thực hiện đủ 100 lần). Ta nói cách lặp trong thuật toán Tong\_1a là dạng tiến và trong thuật toán Tong\_1b là dạng lùi.

Để mô tả cấu trúc lặp với số lần biết trước, C++ dùng câu lệnh lặp for có cú pháp như sau:

for(khởi tạo;điều kiện, thay đổi)

{câu lệnh;

Câu lệnh;



*Đặng Trung Kiên* *Trang 57*

*Ngôn ngữ lập trình C++*



…..

Câu lệnh

};

Vòng lặp for thực hiện lặp lại câu lệnh chừng nào điều kiện còn nhận giá trị đúng như trong vòng lặp while Nhưng trong for còn chứa khả năng khởi tạo và khả năng làm thay đổi giá trị các biến điều khiển vòng lặp.

***Hoạt động của vòng for như sau:***

Bước 1. Điều kiện khởi tạo được thực hiện thường để đặt một giá trị ban đầu cho các biến điều khiển, điều này chỉ được thực hiện một lần.

Bước 2. Điều kiện được kiểm tra, nếu nó nhận giá trị đúng thì câu lệnh được thực hiện, ngược lại thì tới bước 4

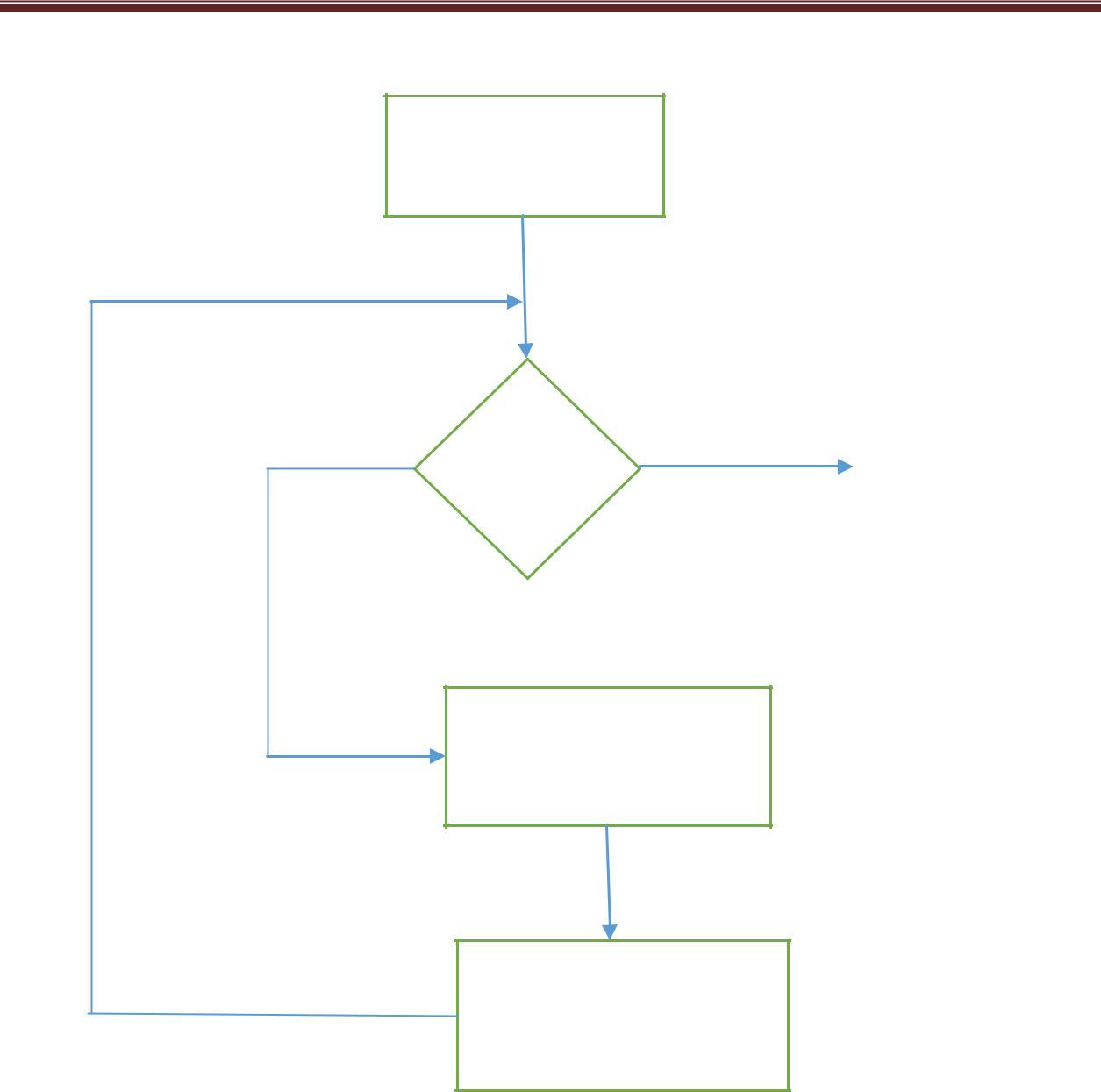
Bước 3. Thay đổi được thực hiện để thay đổi giá trị biến điều khiển, quay về bước 2.

Bước 4. Kết thúc vòng lặp



*Đặng Trung Kiên* *Trang 58*

*Ngôn ngữ lập trình C++*



Khởi tạo

|  |  |  |
| --- | --- | --- |
| Đúng | Điều | Sai |
|  |  |
|  | kiện |  |

Câu lệnh

Thay đổi điều kiện

***Hình 1*** ***: Minh họa câu lệnh for trong C++***

Ví dụ 1. Đưa ra màn hình các số tự nhiên từ 10 đến 1

#include <iostream>

#include<math.h>

#include<iomanip>

using namespace std;

int main()

{

for(int n=10;n>0;n--) cout<<n<<" ";



*Đặng Trung Kiên* *Trang 59*

*Ngôn ngữ lập trình C++*



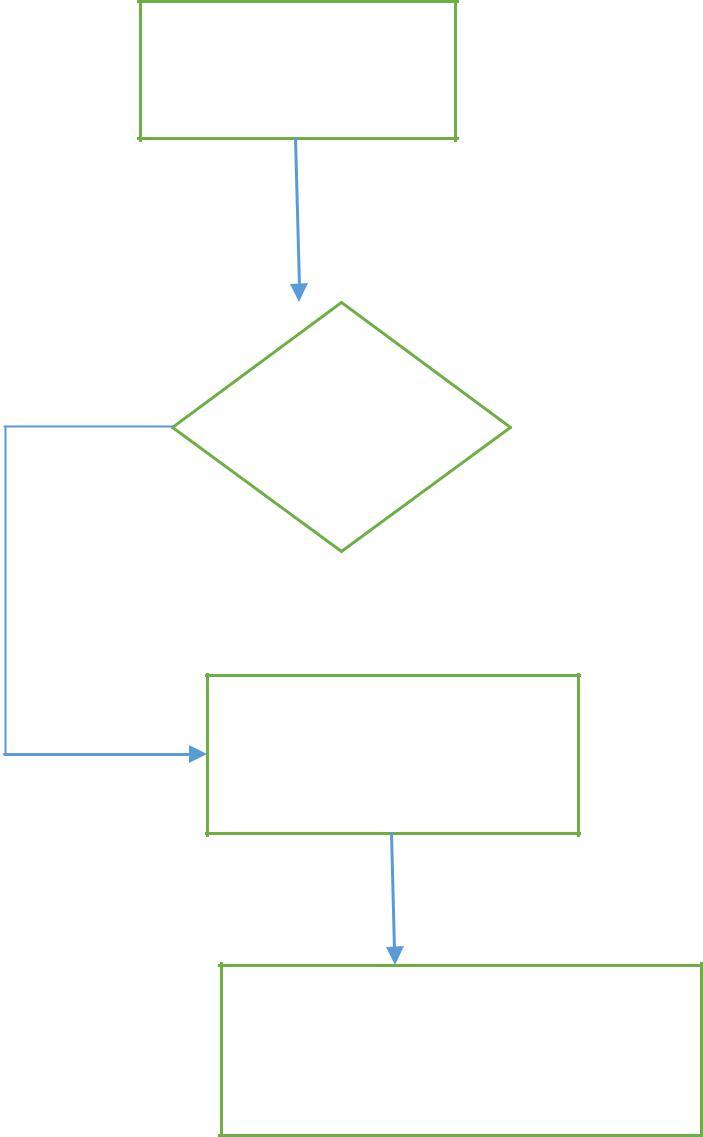
return 0;

}

Ouput

10 9 8 7 6 5 4 3 2 1

Giải thích Lúc đầu n được khởi tạo bằng 10, kiểm tra điều kiện 10 có lớn hơn 0 hay không? Nếu đúng thì đưa n ra màn hình, sau đó thay đổi điều kiện n—- nghĩa là n giảm xuống 1 đơn vị còn bằng 9



Khởi tạo n

bằng 10

Đúng 10>0

Câu lệnh đưa giá

trị 10 ra màn hình

Thay đổi điều kiện

n—nghĩa là n giảm

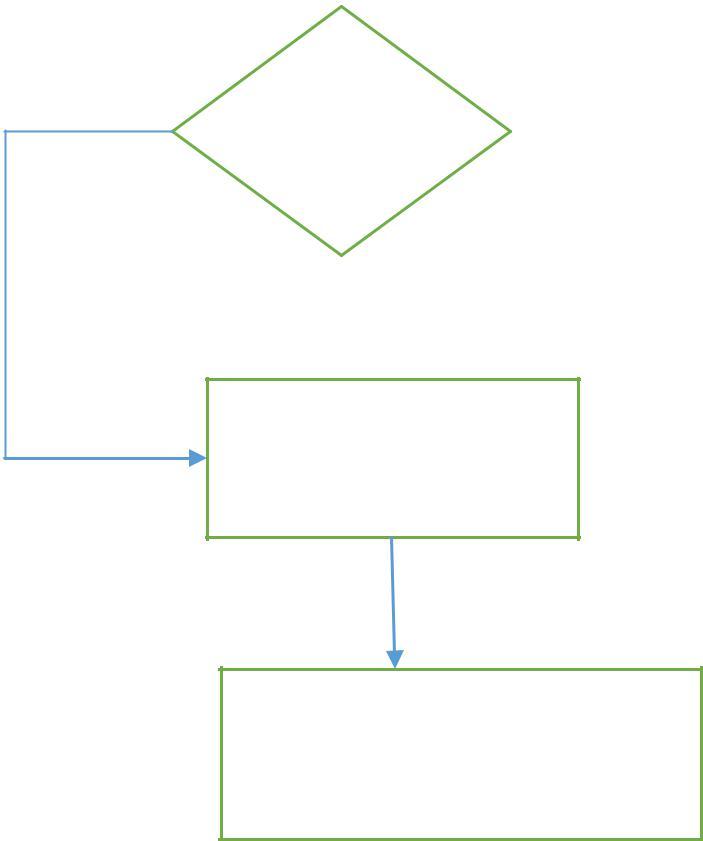
xuống còn bằng 9

Tiếp theo , n bằng 9 quay trở lại kiểm tra điều kiện xem 9 có lớn hơn 0 hay không



*Đặng Trung Kiên* *Trang 60*

*Ngôn ngữ lập trình C++*



Đúng 9>0

Câu lệnh đưa giá

trị 9 ra màn hình

Thay đổi điều kiện

n—nghĩa là n giảm

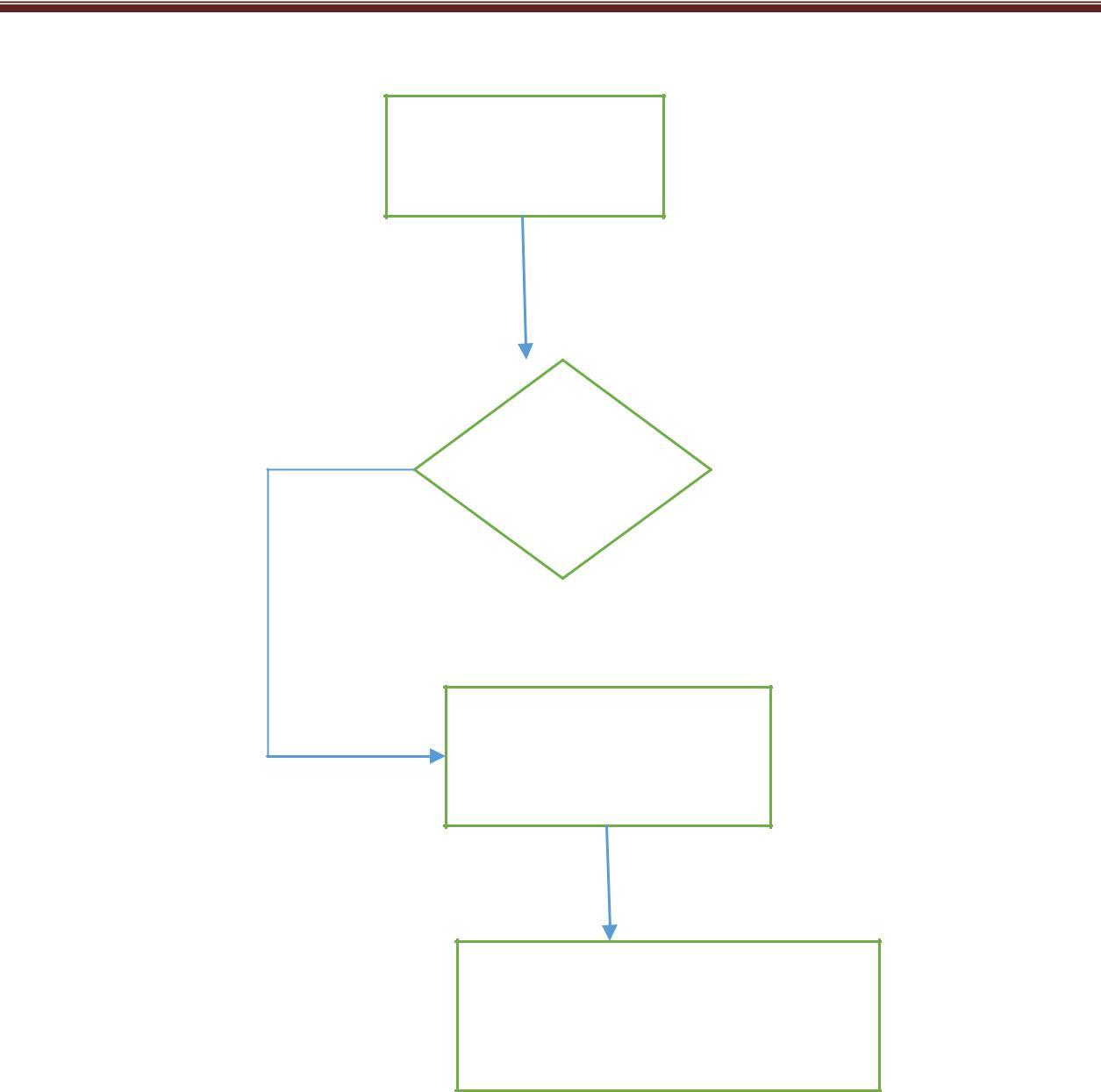
xuống còn bằng 8

Cứ thế, cho đến khi n giảm xuống còn 7,6,5,4,3,2,1



*Đặng Trung Kiên* *Trang 61*

*Ngôn ngữ lập trình C++*



Khởi tạo n

bằng 10

Đúng 1>0

Câu lệnh đưa giá

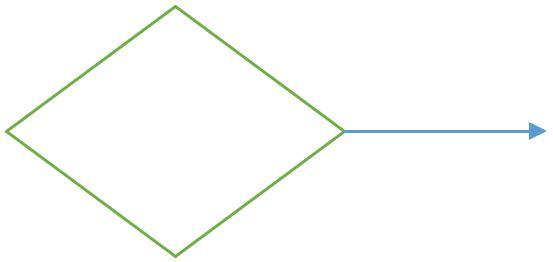
trị 1 ra màn hình

Thay đổi điều kiện

n—nghĩa là n giảm

xuống còn bằng 0

Lúc này, quay trở lại điều kiện, 0 không lớn hơn 0 nên thoát khỏi vòng lặp for



Sai

0>0

Ví dụ 2. Đưa ra màn hình các cặp số tự nhiên khác nhau có tổng bằng 20

#include <iostream>

#include<math.h>

#include<iomanip>



*Đặng Trung Kiên* *Trang 62*

*Ngôn ngữ lập trình C++*



using namespace std;

int main()

{

for(int n=20,i=0;n!=i;n--,i++)

cout<<n<<","<<i<<'\n';

return 0;

}

Giải thích: Khởi tạo n ban đầu bằng 20, i bằng 0. Sau đó kiểm tra điều kiện, nếu thỏa mãn thì n giảm một đơn vị, i tăng 1 đơn vị. Vòng lặp kết thúc khi gặp cặp số n=10;i=10

Ví dụ 3. Sau đây là chương trình cài đặt các thuật toán Tong\_1a và Tong\_1b

#include <bits/stdc++.h>

using namespace std;

double s;

int a,n;

int main()

{ cout<<"Nhap vao gia tri cua a"; cin>>a;

s=1.0/a;

for(n=1;n<=100;n++) s=s+1.0/(a+n); cout<<fixed<<setprecision(4)<<"tong s la"<<s;

return 0;

}

Chương trình tính tong\_1b

#include <bits/stdc++.h>

using namespace std;

double s;

int a,n;



*Đặng Trung Kiên* *Trang 63*

*Ngôn ngữ lập trình C++*



int main()

{ cout<<"Nhap vao gia tri cua a"; cin>>a;

s=1.0/a;

for(n=100;n>=1;n--) s=s+1.0/(a+n); cout<<fixed<<setprecision(4)<<"tong s la"<<s;

return 0;

}

Ví dụ 4. Chương trình sau thực hiện việc nhập từ bàn phím hai số nguyên dương M và N (M<N), tính và đưa ra màn hình tổng các số chia hết cho 3 hoặc 5 trong phạm vi từ M đến N.

#include <bits/stdc++.h>

using namespace std;

double s;

int m,n,i,t;

int main()

{ cout<<"Nhap so M nho hon N"; cout<<"M=";

cin>>m;

cout<<"N=";

cin>>n;

t=0;

for(i=m;i<=n;i++) if((i%3==0)||(i%5==0)) t=t+i;

cout<<"ket qua"<<t; return 0;

}

***3. Lặp với số lần chưa biết trước và câu lệnh while***

Có thể xây dựng thuật toán Tong\_2 như sau để giải bài toán 2.



*Đặng Trung Kiên* *Trang 64*

*Ngôn ngữ lập trình C++*



Thuật toán Tong\_2

Bước 1. *S* ← 1/ *a*; *N* ← 0; {khởi tạo S và N}

Bước 2. Nếu 1/ (*a* + *N* ) < 0.0001thì chuyển đến bước 5; Bước 3. *N* ← *N* +1 ;

Bước 4. *S* ← 1/ (*a* + *N* ) rồi quay lại bước 2; Bước 5. Đưa S ra màn hình, rồi kết thúc

Như vậy, việc lặp với số lần chưa biết trước sẽ chỉ kết thúc khi một điều kiện cho trước được thỏa mãn.

Để mô tả cấu trúc lặp như vậy C++ dùng câu lệnh while… có dạng:

***while (điều kiện)***

***{câu lệnh;***

***Câu lệnh;***

***……***

***Câu lệnh;***

***}***

Trong đó:

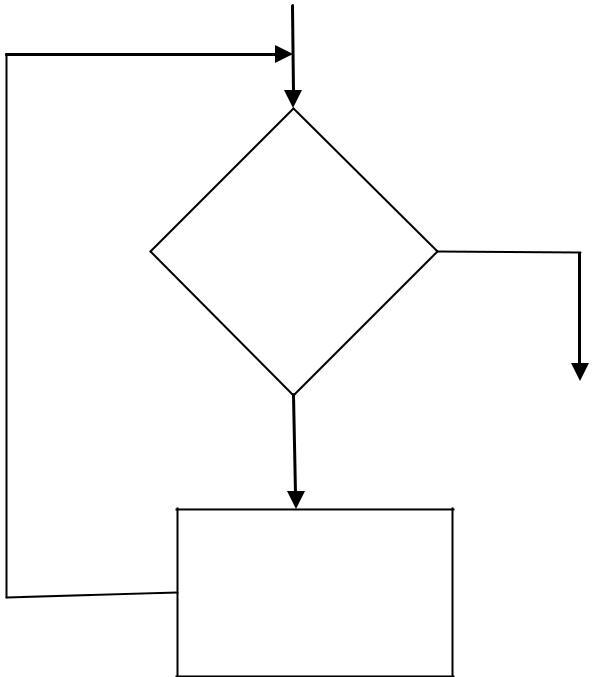
* Điều kiện là biểu thức logic;
* Câu lệnh là một câu lệnh đơn hoặc ghép

Việc thực hiện lệnh while được thể hiện trên sơ đồ ở hình 7



*Đặng Trung Kiên* *Trang 65*

*Ngôn ngữ lập trình C++*



|  |  |
| --- | --- |
| Điều | Sai |
|  |
| kiện |  |

Đúng

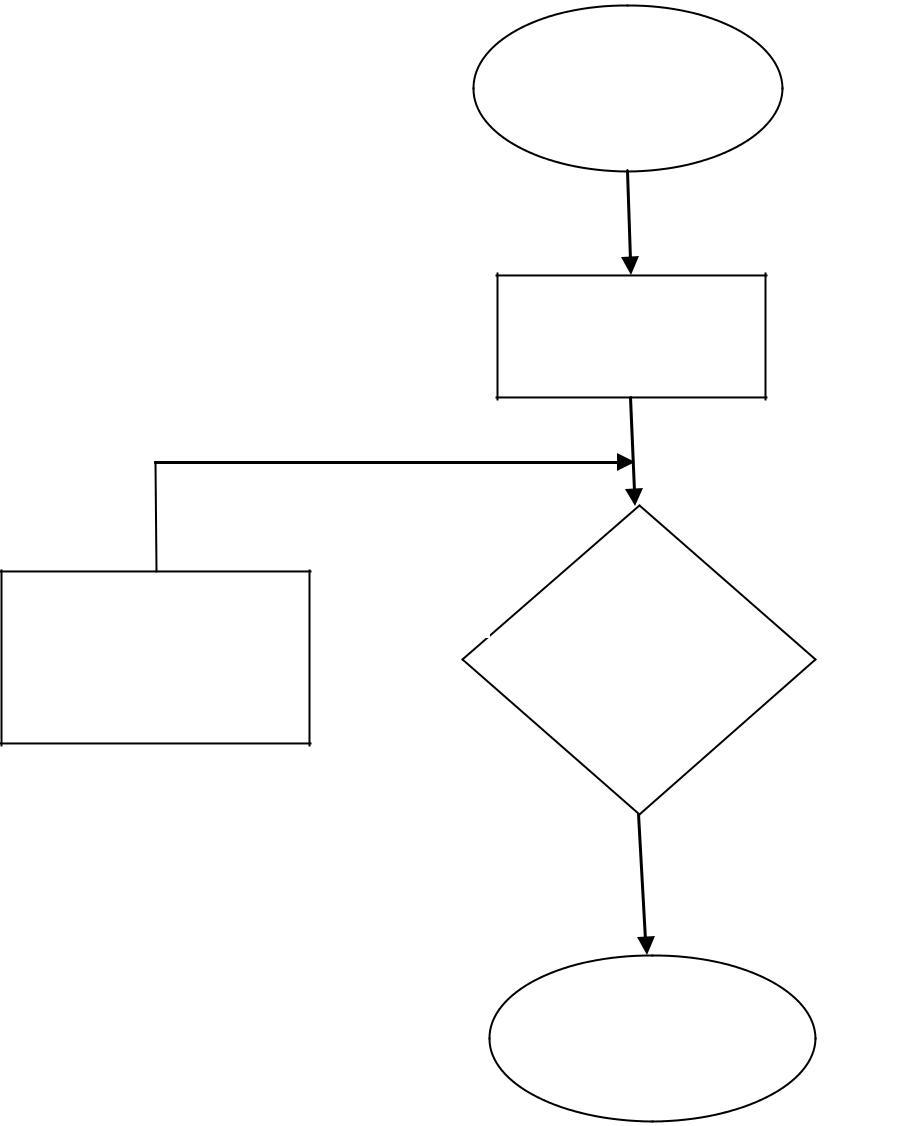
Câu lệnh

Ví dụ: Sau đây là sơ đồ khối và chương trình cài đặt thuật toán Tong\_2



*Đặng Trung Kiên* *Trang 66*

*Ngôn ngữ lập trình C++*



Nhập a

S<-1/a;

N<-0

Sai

N<-N+1; 1/(a+N)

S<-S+1/(a+N) <0.0001

?

Đúng

Đưa ra S

rồi kết thúc

Hình : Sơ đồ khối của thuạt toán Tong\_2



*Đặng Trung Kiên* *Trang 67*

*Ngôn ngữ lập trình C++*



#include <bits/stdc++.h>

using namespace std;

double s;

int a,n;

int main()

{ cout<<"Hay nhap gia tri a vao"; cin>>a;

s=1.0/a;n=0;

while (!(1/(a+n)<0.0001))

{

n=n+1;

s=s+1.0/(a+n);

}

cout<<fixed<<setprecision(4)<<"tong s la"<<s; return 0;

}



*Đặng Trung Kiên* *Trang 68*

*Ngôn ngữ lập trình C++*



**Bài tập**

Gợi mở tư duy: Nhìn bài toán đó dưới con mắt của **sự lặp lại**. Tìm kiếm một công việc, một hành động nào đó lặp lại nhiều lần, và rút ra kết luận, công thức tổng quát, mối liên hệ giữa hiện tại và quá khứ.

**Bài tập 1**. Một người gửi tiết kiệm có kì hạn với số tiền A đồng, với lãisuất 0.3% mỗi tháng. Hỏi sau N tháng, người đó sẽ nhận được bao nhiêu tiền? Lãi suất sẽ được cộng vào vốn sau mỗi tháng. Với A, N được nhập từ bàn phím.

**Bài tập 2**. Một người gửi tiết kiệm không kì hạn với số tiền A đồng vớilãi suất 0.3% mỗi tháng. Hỏi sau bao nhiêu tháng, người đó rút hết tiền thì sẽ nhận được số tiền ít nhất là B đồng. Biết rằng với việc gửi tiết kiệm không kì hạn thì lãi suất không được cộng vào vốn. Với A, B nhập từ bàn phím

**Bài 3**. Lập trình tính

1. *n*

*a*)*Y* =∑*n*=1 *n* +1



|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *a*)*e*( *n*)=1+ |  | 1 | + | 1 | + .... + | 1 | + ....cho đến khi | 1 | < 2.10−6 thì không cộng |
| 1! | | 2! | *n*! | *n*! |
|  |  |  |  |  |

vào và dừng lại

**Bài 4**. Lập trình nhập từ bàn phím số nguyên n(0<*n*≤32767) . Hãy xácđịnh và đưa ra màn hình số chữ số 0 có nghĩa của n.

**Bài 5**. Lập trình nhập số thực x và số nguyên dương n từ bàn phím. Tìmsố nguyên k nhỏ nhất thỏa mãn điều kiện 3*k* > *n* . Đưa ra màn hình số k và giá trị 3*k*

**Bài 6**. Nhập vào một số nguyên dương N0≤*N*≤109. Phân tích N thànhcác thừa số nguyên tố.



*Đặng Trung Kiên* *Trang 69*

*Ngôn ngữ lập trình C++*



**CHƢƠNG IV. KIỂU DỮ LIỆU CÓ CẤU TRÚC**

**BÀI 11. KIỂU MẢNG**

***1. Kiểu mảng một chiều***

Mảng một chiều là dãy hữu hạn các phần tử cùng kiểu. Mảng được đặt tên và mỗi phần tử của nó có một chỉ số. Để mô tả mảng một chiều cần xác định kiểu của các phần tử và cách đánh số các phần tử của nó.

Để người lập trình có thể xây dựng và sử dụng kiểu mảng một chiều, các ngôn ngữ lập trình có quy tắc, cách thức cho phép xác định:

* + Tên kiểu mảng một chiều
  + Số lượng phần tử
  + Kiểu dữ liệu của phần tử
  + Cách khai báo biến mảng
  + Cách tham chiếu đến phần tử

1. ***Khai báo mảng một chiều***

***a) Cú pháp***

***Kiểu dữ liệu từng phần tử của mảng tên mảng[số lượng phần tử]; Ví dụ int a[5];***

**Ví dụ trên khai báo một mảng gồm 5 phần tử, chỉ số của mảng đƣợc đánh số từ 0 tới 4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** |
|  |  |  |  |  |
| **a[0]** | **a[1]** | **a[2]** | **a[3]** | **a[4]** |
|  |  |  |  |  |

1. **Khởi tạo mảng**
   * **Khởi tạo toàn bộ các phần tử của mảng có giá trị bằng 0**

***Kiểu dữ liệu tên mảng[số lượng phần tử]={}***

Ví dụ: int a[5]={}

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **Chỉ số** |  |
|  |  |  |  |  |  |  |  |
|  | **0** | **0** | **0** | **0** | **0** | **Giá trị** |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  | *Đặng Trung Kiên* | |  |  |  | *Trang 70* |  |

*Ngôn ngữ lập trình C++*



∙ **Gán một vài giá trị ban đầu cho mảng** ∙ Ví dụ: int a[5]={3,2,8,9,7}

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 |  | 2 |  | 3 | 4 |  | **Chỉ số** |
|  |  |  |  |  |  |  |  |  |
| 3 | 2 |  | 8 |  | 9 | 7 |  | **Giá trị** |
|  |  |  |  |  |  |  |  |  |
| **3. Một số ví dụ** | |  |  |  |  |  |  |  |
| **Ví dụ 1.** Tìm phần tử lớn nhất của dãy số nguyên | | | | | |  |  |  |
| Input: Số | nguyên | dương | | N (N≤250) và dãy | | N | số nguyên dương | |

*a*1, *a*2,....,*an* , mỗi số đều không vượt quá 500.

Ouput: Chỉ số và giá trị của phần tử lớn nhất trong dãy số đã cho (nếu có nhiều phần tử lớn nhất chỉ cần đưa ra mộ trong số chúng)

Bước 1. Nhập số nguyên dương N (N≤250) và dãy N số nguyên dương

*a*1, *a*2,....,*an* ,

Bước 2. max ← *a*1 , *i* ← 2

Bước 3. Nếu i>N thì đưa ra giá trị max rồi kết thúc; Bước 4.

Bước 4.1 Nếu *ai* > max thì max ← *ai*

Bước 4.2: *i* ← *i* +1 rồi quay lại bước 3;

Chương trình dưới đây thực hiện việc duyệt tuần tự các phần tử để tìm ra số lớn nhất của dãy số nguyên

**Chú ý đối với Toán học hay trong các đề bài, chỉ số của phần tử thƣờng đƣợc đánh số từ 1. Còn trong C++ chỉ số phần tử đƣợc đánh số từ 0. Nên khi lập trình với mảng cần để ý tới điều này.**

#include <iostream>

using namespace std;

const long nmax=250;

long n,i,max1,csmax,a[nmax];

int main()

{

cout <<"Nhap so luong phan tu cua day so"; cin>>n;

for(long i=0;i<=n-1;i++)

{

cout<<"nhap phan tu thu"<<i+1<<"="; cin>>a[i];

}

max1=a[0];

csmax=1;

for(long i=1;i<=n-1;i++)



*Đặng Trung Kiên* *Trang 71*

*Ngôn ngữ lập trình C++*



if (a[i]>max1) {max1=a[i];csmax=i;}; cout<<"Gia tri cua phan tu max"<<max1; cout<<"chi so cua phan tu max"<<csmax; return 0;

}

**Ví dụ 2. Sắp xếp dãy số nguyên bằng thuật toán tráo đổi**

Input: Số nguyên dương N ( *N* ≤ 250) và dãy A gồm N số nguyên dương *a*1, *a*2,...,*an* , mỗi số đều không vượt quá 500.

Ouput: Dãy số A đã được sắp xếp thành dãy không giảm

**Chú ý đối với Toán học hay trong các đề bài, chỉ số của phần tử thƣờng đƣợc đánh số từ 1. Còn trong C++ chỉ số phần tử đƣợc đánh số từ 0. Nên khi lập trình với mảng cần để ý tới điều này.**

#include <iostream>

#include<iomanip>

using namespace std;

const long nmax=250;

long n,i,max1,csmax,trunggian,a[nmax];

int main()

{

cout <<"Nhap so luong phan tu cua day so"; cin>>n;

for(long i=0;i<=n-1;i++)

{

cout<<"phan tu thu"<<i+1<<"="; cin>>a[i];

}

for(long j=n-1;j>=1;j--)

for(long i=0;i<=j-1;i++)

if(a[i]>a[j])

{trunggian=a[i];

a[i]=a[j];

a[j]=trunggian;

}

cout<<"day so duoc sap xep la:\n"; cout<<fixed;

for(long i=0;i<=n-1;i++) cout<<setw(5)<<a[i];//danh 5 vi tri

cho a[i]

return 0;

}

Một cách khác không cần để ý tới việc chỉ số mảng đánh từ 0 trong C++, đó là ta khai báo số lượng phần tử mảng trong C++ rộng ra, và ta quan



*Đặng Trung Kiên* *Trang 72*

*Ngôn ngữ lập trình C++*



niệm chỉ số mảng đánh từ 0 như bình thường, có điều bỏ qua phần tử

a[0]. Như đoạn code sau:

#include <iostream>

#include<iomanip>

using namespace std;

const long nmax=250;

long n,i,max1,csmax,trunggian,**a[nmax+1];//**251 phần tử

int main()

{

cout <<"Nhap so luong phan tu cua day so"; cin>>n;

for(long i=1;i<=n;i++)

{

cout<<"phan tu thu"<<i<<"="; cin>>a[i];

}

for(long j=n;j>=2;j--)

for(long i=1;i<=j-1;i++)

if(a[i]>a[j])

{trunggian=a[i];

a[i]=a[j];

a[j]=trunggian;

}

cout<<"day so duoc sap xep la:\n"; cout<<fixed;

for(long i=1;i<=n;i++)

cout<<setw(5)<<a[i];//danh 5 vi tri

cho a[i]

return 0;

}

**Ví dụ 3. Tìm kiếm nhị phân**

**Input**: Dãy A là dãy tăng gồm N ( *N* ≤ 250) số nguyên dương *a*1, *a*2,...,*an* ,và số nguyên k.

Ouput: Chỉ số i mà ai=k hoặc thông báo: ―khong tim thay‖ nếu không có số hạng nào của dãy A có giá trị bằng k

**Ouput:** Dãy số A đã được sắp xếp thành dãy không giảm

Để giải bài toán này, chúng ta sẽ sử dụng thuật toán tìm kiếm nhị phân được trình bày trong sách giáo khoa Tin học lớp 10.

*Bước 1. Nhập N, các số hạng a*1, *a*2,....,*an và khóa k*

*Bước 2 dau* ←1,*cuoi* ← *n*



*Đặng Trung Kiên* *Trang 73*

*Ngôn ngữ lập trình C++*



*dau* + *cuoi*

*Bước 3 giua* ←[ ]



*Bước 4. Nếu agiua* = *k thì thông báo chỉ số giua, rồi kết thúc; Bước 5. Nếu agiua* > *k thì đặt cuoi=giua-1 rồi chuyến đến bước 7;*

*Bước 6. dau* ← *giua* +1

*Bước 7. Nếu dau>cuoi thì thông báo dãy A không có số hạng có giá trị*

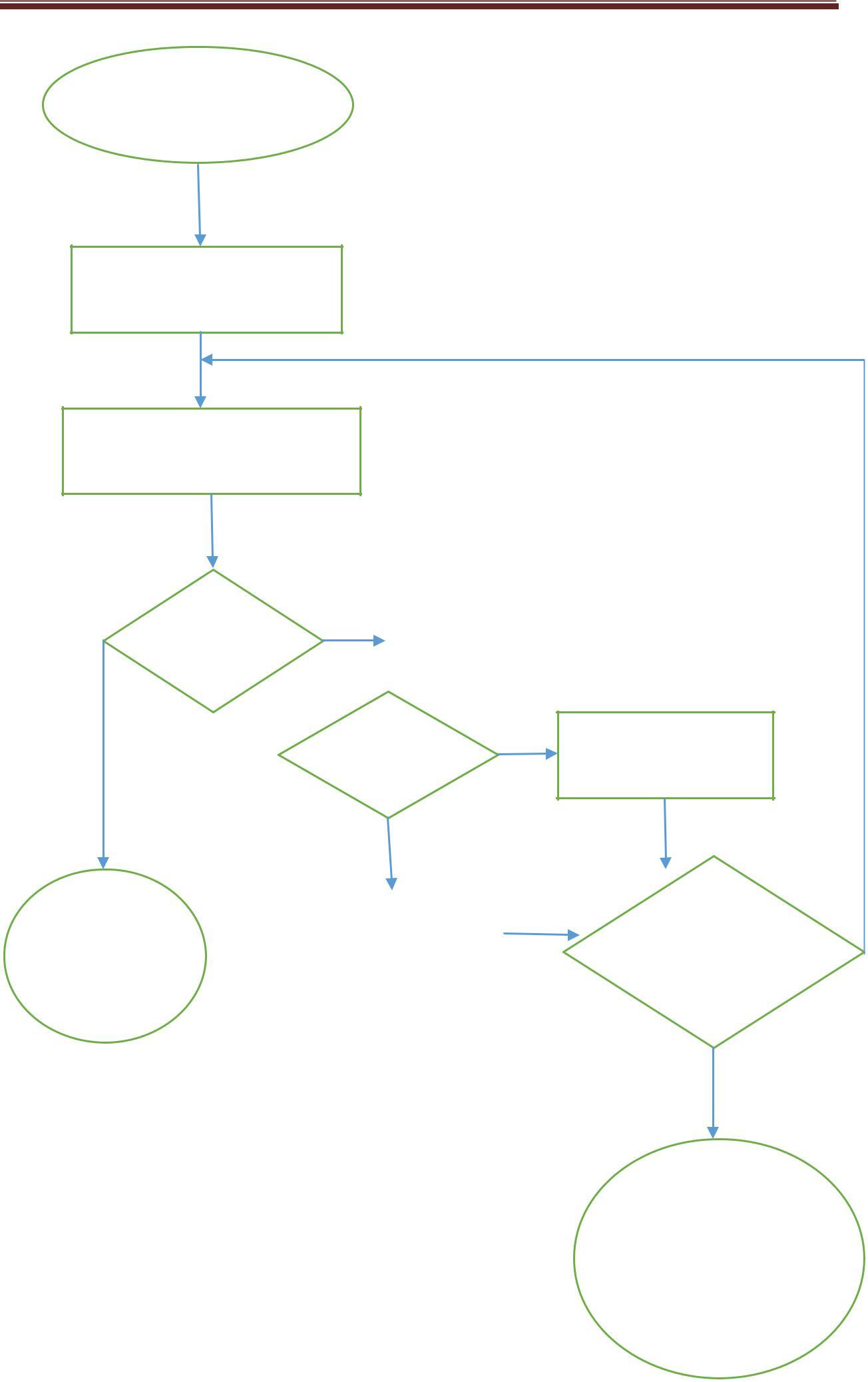
*bằng k, rồi kết thúc;*

*Bước 8: Quay lại bước 3.*



*Đặng Trung Kiên* *Trang 74*

*Ngôn ngữ lập trình C++*



Nhập N và

a1,a2,…ak;k

*Dau* ←1;*Cuoi* ← *N*

*giua* ←[( *dau* + *cuoi*) / 2]

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | Sai | | |  |  | |
| Đúng | *agiua* = *k*? | |  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |
|  |  |  |  |  |  |  | |
|  |  |  |  |  |  | Đúng | |
|  |  |  |  |
|  |  |  |  | | | |  | |
|  |  |  | *agiua* > *k* | | | | | |
|  |  |  | Sai | | | | | |
|  |  |  |  | | |  |  |  |
| Đưa ra |  | *Dau* ← *giua* +1 | | | | | |  |
| Giua rồi |  |  |  |  |  |  |  |  |
| kết thúc |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |



*Cuoi* ← *Giua* −1

|  |  |
| --- | --- |
| *Dau* > *Cuoi* ? | Sai |
|  |

Đúng

Thông báo dãy

A không có số

hạng có giá trị

bằng k rồi kết

thúc

*Đặng Trung Kiên* *Trang 75*

*Ngôn ngữ lập trình C++*



#include <iostream>

#include<iomanip>

using namespace std;

const long nmax=250;

long n,i,k,dau, cuoi,giua,a[nmax];

bool tim\_thay;

int main()

{

cout <<"Nhap so luong phan tu cua day so "; cin>>n;

cout<<"nhap cac phan tu cua day so tang \n"; for(long i=0;i<=n-1;i++) {

cout<<"phan tu thu"<<i+1<<"=";

cin>>a[i];

}

cout<<"nhap gia tri k= \n";

cin>>k;

dau=0;cuoi=n-1;tim\_thay=false;

while((dau<=cuoi)&& tim\_thay==false)

{

giua=(dau+cuoi)/2;

if(a[giua]==k) tim\_thay=true;

else

if (a[giua]>k) cuoi=giua-1;

else dau=giua+1;

}

if (tim\_thay==true) cout<<"chi can tim duoc la"<<giua+1;

else cout<<"khong tim thay"; return 0;

}

Trong phần tìm trên, sở dĩ ta phải đưa ra giua+1 vì lý do, chỉ số mảng trong C++ được đánh số từ 0, nên bao giờ thứ tự của nó cũng nhỏ hơn chỉ số mảng theo quan niệm Toán học đánh số từ 1.

**Có một cách khác tránh nhầm lẫn là ta vẫn quan niệm chỉ số mảng nhƣ Toán học, nhƣng lúc khai báo mảng ta khai báo số lƣợng phần tử mảng rộng hơn nhƣ đoạn code sau:**

#include <iostream>

#include<iomanip>

using namespace std;

const long nmax=250;

long n,i,k,dau, cuoi,giua,**a[nmax+1];//251 phần tử** bool tim\_thay;

int main()

{

cout <<"Nhap so luong phan tu cua day so "; cin>>n;

cout<<"nhap cac phan tu cua day so tang \n"; for(long i=1;i<=n;i++)



*Đặng Trung Kiên* *Trang 76*

*Ngôn ngữ lập trình C++*



{

cout<<"phan tu thu"<<i<<"=";

cin>>a[i];

}

cout<<"nhap gia tri k= \n";

cin>>k;

dau=1;cuoi=n;tim\_thay=false;

while((dau<=cuoi)&& tim\_thay==false)

{

giua=(dau+cuoi)/2;

if(a[giua]==k) tim\_thay=true;

else

if (a[giua]>k) cuoi=giua-1;

else dau=giua+1;

}

if (tim\_thay==true)cout<<"chi so tim duoc la"<<giua; else cout<<"khong tim thay";

return 0;

}

***4. Khai báo mảng nhiều chiều***

**a) Khai báo mảng hai chiều**

Ví dụ int a[5][6]

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| a[hàng][cột] | **0** | **1** | **2** | **3** | **4** | **5** | **6** |
|  |  |  |  |  |  |  |  |
| **0** | 15 | 20 | 26 | 3 | 7 | 8 | 9 |
|  |  |  |  |  |  |  |  |
| **1** | 25 | 3 | 7 | 8 | 9 | 9 | 9 |
|  |  |  |  |  |  |  |  |
| **2** | 18 | 17 | 23 | 22 | 27 | 76 | 78 |
|  |  |  |  |  |  |  |  |
| **3** | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|  |  |  |  |  |  |  |  |
| **4** | 3 | 5 | 4 | 6 | 7 | 8 | 9 |
|  |  |  |  |  |  |  |  |
| **5** | 1 | 2 | 87 | 67 | 89 | 8 | 9 |
|  |  |  |  |  |  |  |  |
| a[0][0] =15; |  |  |  |  |  |  |  |

a[0][1]=20;

a[2][5]=76;

#define WIDTH 5

#define HEIGHT 3

int jimmy [HEIGHT][WIDTH];

int n,m;



*Đặng Trung Kiên* *Trang 77*

*Ngôn ngữ lập trình C++*



int main ()

{

for (n=0; n<HEIGHT; n++)

for (m=0; m<WIDTH; m++)

{

jimmy[n][m]=(n+1)\*(m+1);

}

}



*Đặng Trung Kiên* *Trang 78*

*Ngôn ngữ lập trình C++*



***Bài tập về mảng***

**Bài 1.** Viết chương trình nhập từ bàn phím số nguyên dương N(*N*≤109)và dãy A gồm N số nguyên *A*1 , *A*2 ,....*AN* có giá trị tuyệt đối không lớn

hơn 1000. Hãy cho biết dãy A có phải là cấp số cộng hay không và thông báo kết quả ra màn hình?

**Bài 2.** Viết chương trình nhập từ bàn phím số nguyên dương

N ( *N* ≤109 ) và dãy A gồm N số nguyên *A*1 , *A*2 ,....*AN* có giá trị tuyệt đối

không lớn hơn 1000. Hãy đưa ra những thông tin sau:

1. Số lượng số chẵn và số lẻ trong dãy.
2. Số lượng số nguyên tố trong dãy. **Bài 3**. Lập trình:
   * Nhập từ bàn phím hai số nguyên M và N −32768 ≤ *M* , *N* ≤ 32767
   * Tính và đưa ra màn hình số lần xuất hiện của các chữ số có nghĩa của tổng M+N trong dạng biểu diễn thập phân. Kết quả cần đưa ra

dưới dạng:

SO LAN XUAT HIEN CHU SO <chữ số>:<số lượng> Theo thứ tự tăng dần của các chữ số

Ví dụ, với M=309 và N=21, kết quả đưa ra màn hình sẽ là:

SO LAN XUAT HIEN CHU SO 0: 1

SO LAN XUAT HIEN CHU SO 3: 2

**Bài 4.**

Nhập vào một dãy n số nguyên A[1],A[2],...,A[n] . Đọc ra màn hình các thông tin sau :

1. Tổng các phần tử của dãy.
2. Số lượng các số hạng dương và tổng của các số hạng dương.
3. Số lượng các số hạng âm và tổng của các số hạng âm.
4. Trung bình cộng của cả dãy. Trung bình cộng các phần tử dương của mảng. Trung bình cộng các phần tử âm của mảng.
5. Chỉ số của số hạng dương đầu tiên của dãy.
6. Chỉ số của số hạng âm đầu tiên của dãy.
7. Chỉ số của số hạng dương cuối cùng của dãy.
8. Chỉ số của số hạng âm cuối cùng của dãy.
9. Số hạng lớn nhất của dãy và chỉ số của nó.
10. Số hạng nhỏ nhất của dãy và chỉ số của nó.
11. Số hạng âm lớn nhất của dãy và chỉ số của nó.
12. Số hạng dương nhỏ nhất của dãy và chỉ số của nó.
13. Giá trị lớn thứ nhì của dãy và các chỉ số của các số hạng đạt giá trị lớn nhì.
14. Giá trị nhỏ thứ nhì của dãy và các chỉ số của các số hạng đạt giá trị nhỏ nhì.



*Đặng Trung Kiên* *Trang 79*

*Ngôn ngữ lập trình C++*



1. Số lượng các số dương liên tiếp nhiều nhất.
2. Số lượng các số hạng dương liên tiếp có tổng lớn nhất.
3. Số lượng các số hạng âm liên tiếp nhiều nhất.
4. Số lượng các số hạng âm liên tiếp có tổng lớn nhất.
5. Số lượng các số hạng liên tiếp đan dấu nhiều nhất ( dãy số hạng liên tiếp được gọi là đan dấu nếu tích hai số liên tiếp âm )
6. Đoạn con dương liên tiếp có nhiều các số hạng nhất (nếu có nhiều đoạn con thoả mãn thì đọc ra màn hình : số đoạn con thoả mãn và các đoạn con đó )
7. Đoạn con gồm nhiều nhất các số hạng liên tiếp của dãy lập thành một cấp số cộng nhất (nếu có nhiều đoạn con thoả mãn thì đọc ra màn hình : số đoạn con thoả mãn và các đoạn con đó )
8. Đoạn con có các số hạng dương liên tiếp có tổng lớn nhất (nếu có nhiều đoạn con thoả mãn thì đọc ra màn hình : số đoạn con thoả mãn và các đoạn con đó )
9. Đoạn con có các số hạng âm liên tiếp nhiều nhất (nếu có nhiều đoạn con thoả mãn thì đọc ra màn hình : số đoạn con thoả mãn và các đoạn con đó )
10. Đoạn con có số hạng âm liên tiếp có tổng lớn nhất (nếu có nhiều đoạn con thoả mãn thì đọc ra màn hình : số đoạn con thoả mãn và các đoạn con đó )
11. Đoạn con có các số hạng liên tiếp đan dấu nhiều nhất(nếu có nhiều đoạn con thoả mãn thì đọc ra màn hình : số đoạn con thoả mãn và các đoạn con đó )
12. Đoạn con có các phần tử không tăng nhiều nhất. nhất (nếu có nhiều đoạn con thoả mãn thì đọc ra màn hình : số đoạn con thoả mãn và các đoạn con đó )
13. Đoạn con có các phần tử giảm nhiều nhất. nhất (nếu có nhiều đoạn con thoả mãn thì đọc ra màn hình : số đoạn con thoả mãn và các đoạn con đó )
14. Số lượng các phần tử bằng giá trị X ( nguyên ) cho trước.
15. Số lượng các phần tử khác giá trị X ( nguyên ) cho trước.
16. Số lượng các phần tử >= giá trị X ( nguyên ) cho trước.
17. Chuyển các số hạng dương của mảng lên đầu mảng và in mảng ra màn hình.
18. Tìm số phần tử là dương và là số nguyên tố của mảng và vị trí của nó trong mảng
19. Sắp xếp tăng dần mảng đã cho (a[i]<=a[i+1] )
20. Sắp xếp giảm dần mảng đã cho (a[i]>=a[i+1] )
21. Chèn một số nguyên m (m nhập vào từ bàn phím ) vào cuối dãy
22. Chèn một số nguyên m (m nhập vào từ bàn phím ) vào đầu dãy
23. Chèn một số nguyên m ( m nhập vào từ bàn phím ) vào vị trí k .



*Đặng Trung Kiên* *Trang 80*

*Ngôn ngữ lập trình C++*



1. Chèn một số nguyên m ( m nhập vào từ bàn phím ) vào vị trí thích hợp .

VD: m =5 dãy đã cho sau khi sắp xếp lại là : 1 2 3 4 6

Thì dãy sau khi chèn là: 1 2 3 4 5 6



*Đặng Trung Kiên* *Trang 81*

*Ngôn ngữ lập trình C++*



**BÀI 12. KIỂU XÂU**

Dữ liệu trong các bài toán không chỉ thuộc kiểu số mà cả kiểu phi số - dạng ký tự Dữ liệu kiểu xâu là dãy các kí tự Ví dụ: Các xâu ký tự đơn giản

―Tin hoc‖ ―Toan hoc‖ ―THANH CONG‖ ―Lop 10 Tin rat quyet tam hoc gioi mon Tin‖

Xâu là dãy các kí tự trong bộ mã ASCII, mỗi kí tự được gọi là một phần tử của xâu. Số lượng kí tự trong một xâu được gọi là độ dài của xâu. Xâu có độ dài bằng 0 được gọi là xâu rỗng.

Các ngôn ngữ lập trình đều có quy tắc, cách thức cho phép xác định:

* Tên kiểu xâu
* Cách khai báo biến kiểu xâu
* Số lượng kí tự của xâu
* Các phép toán thao tác với xâu
* Cách tham chiếu tới phần tử của xâu

Có thể xem xâu là mảng một chiều mà mỗi phần tử là một kí tự. Các kí tự của xâu được đánh số thứ tự, bắt đầu từ 0.

Tương tự mảng, tham chiếu tới phần tử của xâu được xác định bởi tên biến xâu và chỉ số đặt trong cặp dấu ngoặc [ và ].

Ví dụ, giả sử có biến Hoten lưu trữ giá trị hằng xâu ―Dang Trung Kien‖ thì Hoten[6] cho ta kí tự ‘r‘ là kí tự thứ sáu của biến xâu Hoten

Dưới đây trình bày cách khai báo dữ liệu kiểu xâu, các thao tác xử lý xâu, một số ví dụ sử dụng kiểu xâu trong C++

1. ***Khai báo và nhập dữ liệu cho xâu***

a) Khai báo

**Trƣớc hết ta cần phải khai báo thƣ viện chứa các hàm liên**

**quan đến xâu #include<string>**

**Để khai báo dữ liệu kiểu xâu ta sử dụng tên dành riêng string**



*Đặng Trung Kiên* *Trang 82*

*Ngôn ngữ lập trình C++*



**Biến kiểu xâu đƣợc khai báo nhƣ sau:**

***string <tên biến xâu>***

Muốn biết độ dài lớn nhất của xâu, ta dùng cú pháp

***tên biến xâu.max\_size();***

Độ dài lớn nhất của xâu có thể không cố định, phụ thuộc vào giới hạn của thư viên cung cấp:

#include <iostream>

#include<string>

using namespace std;

int main()

{

string s;

cout<<s.max\_size();

return 0;

}

b) Nhập xâu

**Cách 1. Nhập nhƣ dữ liệu thông thƣờng:**

cin>>tên biến xâu;

Ví dụ: Nếu s là một biến xâu

cin>>s;

Nhưng câu lệnh cin khi gặp các kí tự trắng (như dấu cách, kí tự xuống dòng, ….) sẽ coi là kết thúc việc nhập dữ liệu.

cin không cho phép nhập dấu trắng

Xét chương trình sau đây:

#include <iostream>

#include <string>

using namespace std;

int main(){

string str;



*Đặng Trung Kiên* *Trang 83*

*Ngôn ngữ lập trình C++*



cin>> str;

cout << str << endl;

return 0;

}

Nếu ta nhập vào đoạn văn bản sau: **“Tin hoc”** thì xâu **str** chỉ ghi nhận đoạn đầu **“Tin”** vì dấu cách là một khoảng trắng mà cin coi các dấu trắng là ký tự báo hiệu kết thúc việc nhập dữ liệu. Để khắc phục điều này ***C++*** cung cấp một số cách thức để nhập toàn bộ xâu ký tự. Nhưng trước hết ta phải xem xét một vấn đề về ***bộ đệm***.

1. **Nhập lệnh bằng getline(cin,tên biến xâu) Cú pháp: getline(cin,tên biến xâu)**

**Ví dụ: getline(cin,s); Bộ đệm (buffer)**

Khi ta nhập dữ liệu vào bàn phím thì dữ liệu không được đọc ngay vào biến mà được đẩy lên trên ***bộ đệm****(buffer)*. Dữ liệu sẽ tồn tại trong bộ đệm cho tới khi một lệnh nào đó gửi yêu cầu đến bộ đệm ***“xin phép” được*** ***load dữ liệu về***. Ví dụ khi chương trình gặp câu lệnh:

***cin >> x; // với x là một biến kiểu int***

thì nó sẽ mò lên ***buffer*** để load dữ liệu về, nếu như trên bộ đệm có số ***100*** thì nó sẽ đọc ***100*** vào biến **x** mà không đợi ta nhập gì cả, còn nếu bộ đệm trống hoặc có những dữ liệu không phải số nguyên (ví dụ mã phím**Enter** của lần nhập trước) thì nó mới dừng lại đợi ta nhập dữ liệu vào. Như vậy ta hoàn toàn không cần để ý nhiều việc sử dụng **cin >>** để nhập các dữ liệu số (nguyên hoặc dấu chấm động – floating point), nhưng để ***nhập dữ liệu ký tự*** thì lại hoàn phải hết sức chú ý. Trong đoạn chương trình test nhập xâu ký tự bên trên ta nhận thấy biến str chỉ lưu trữ

|  |  |
| --- | --- |
| phần **“Tin”**, phần còn lại thì vẫn nằm trên | buffer để cho lần nhập |
|  |  |
|  |  |
| *Đặng Trung Kiên* | *Trang 84* |

*Ngôn ngữ lập trình C++*



sau. ***C++***cung cấp **getline** cho phép nhập toàn bộ xâu ký tự kể cả khoảng trắng. Cú pháp sử dụng hàm getline như sau.

***getline(cin, str, delimiter);***

Câu lệnh trên sẽ thực hiện đọc toàn bộ xâu nhập từ bàn phím vào biến **str**, cho tới khi bắt gặp **ký tự kết thúc** *(delimiter)* hoặc **EOF** *(end-of-file)*. Nếu không viết **delimiter** thì mặc định là ký tự xuống dòng – **„\n‟**. Xét chương trình sau:

#include <iostream>

#include <string>

using namespace std;

int main(){

string str;

getline(cin,str);

cout << str << endl;

return 0;

}

Nếu ta nhập vào bàn phím xâu **“Tin hoc”** rồi nhấn **Enter** thì kết quả thu được trên màn hình sẽ là trọn vẹn xâu **“Tin hoc”**. Điều gì xảy ra với ký tự **Enter**, nó có nằm lại trên bộ đệm không? Câu trả lời là **getline** đã đọc mã của ký tự **Enter** nhưng không gắn nó vào trong xâu **str** và cũng không để lại nó trên bộ đệm mà hủy nó đi.

**Hiện tƣợng trôi lệnh**

Xét chương trình sau:

#include <iostream>

#include <string>

using namespace std;



*Đặng Trung Kiên* *Trang 85*

*Ngôn ngữ lập trình C++*



int main(){

int num;

string str;

cout << "Input an integer a= "; cin >> num;

cout << num << endl;

cout << "Input a string str= "; getline(cin,str); cout << str << endl;

cout << "End program" << endl;

return 0;

}

Bạn chạy thử chương trình trên sẽ thấy ngay. Sau khi nhập dữ liệu cho biến **num**, chương trình không dừng lại cho ta nhập dữ liệu cho **str**. Mà in ngay ra thông báo **“End program”**. Nguyên nhân là do sau khi nhập dữ liệu cho biến **num** ta gõ phím **Enter**. Mã của **Enter** được lưu trong bộ đệm (chính là ký tự xuống dòng **„\n‟**) và do đó khi chương trình gặp câu lệnh:

***getline(cin,str);***

nó sẽ đọc ngay ký tự này và nhận thấy đây là ký tự kết thúc nên nó sẽ loại bỏ ký tự này ra khỏi bộ đệm mà không đọc vào xâu str. Sau đó nó sẽ chạy thẳng đến câu lệnh tiếp theo là:

cout << "End program" << endl;



*Đặng Trung Kiên* *Trang 86*

*Ngôn ngữ lập trình C++*



Người ta thường gọi đó là hiện tượng ***“trôi lệnh”***. Để khắc phục hiện tượng trôi lệnh này thì trước mỗi lệnh **getline** ta nên đặt câu lệnh:

***fflush(stdin);***

Câu lệnh này có tác dụng xóa bộ đệm và do đó ta có thể yên tâm là sẽ không bị trôi lệnh nữa.

#include <iostream>

#include <string>

#include<stdio.h>

using namespace std;

int main(){

int num;

string str;

cout << "Input an integer a= "; cin >> num;

cout << num << endl;

cout << "Input a string str= ";

fflush(stdin);

getline(cin,str);



*Đặng Trung Kiên* *Trang 87*

*Ngôn ngữ lập trình C++*



cout << str << endl;

cout << "End program" << endl;

return 0;

}

***2. Các thao tác xử lý xâu***

1. Phép ghép xâu, kí hiệu là dấu cộng (+) Ví dụ ―Tin‖ + ― hoc‖=‖Tin hoc‖
2. Các phép so sánh: nhỏ hơn (<), lớn hơn (>), bằng (==), nhỏ hơn hoặc bằng (<=), lớn hơn hoặc bằng (>=) có thứ tự ưu tiên thực hiện thấp hơn phép ghép xâu và thực hiện việc so sánh hai xâu theo các quy tắc sau:
   * Xâu A được gọi là lớn hơn xâu B nếu như kí tự đầu tiên khác nhau giữa chúng kể từ trái sang trong xâu A có mã ASCII lớn hơn
   * Nếu A và B là các xâu có độ dài khác nhau và A là đoạn đầu của B thì A là nhỏ hơn B

Ví dụ

―May tinh‖ <‖May tinh cua toi‖

* Hai xâu được coi là bằng nhau nếu như chúng giống nhau hoàn

toàn

Ví dụ: ―TIN HOC‖ = ―TIN HOC‖

**Để xử lý các xâu, có thể dùng các hàm chuẩn dƣới đây:**

**1. Xóa phần tử trong xâu erase(pos,len)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Tên hàm** | **Tiếng Anh** | **Chức năng** | **Ghi chú** |  |
|  |  |  |  |  |  |
|  | **erase(pos,len)** | **Erase là** | **Xóa một phần của** | **Nếu vị trí cần xóa** |  |
|  |  | **xóa** | **xâu** | **vƣợt quá vị trí của** |  |
|  |  | **Pos là** | **Pos-vị trí ký tự đầu** | **xâu, một lỗi sai có** |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  | *Đặng Trung Kiên* | |  | *Trang 88* |  |

*Ngôn ngữ lập trình C++*



|  |  |  |  |
| --- | --- | --- | --- |
|  | **position vị** | **tiên đƣợc xóa** | **thể xảy ra** |
|  | **trí** | **Len- số lƣợng các kí** | **Nếu số lƣợng các** |
|  | **Len là** | **tự cần xóa** | **kí tự cần xóa vƣợt** |
|  | **length độ** | **Tóm lại: Xóa đi len** | **quá chiều dài có** |
|  | **dài** | **kí tự kể từ vị trí thứ** | **thể xóa của xâu,** |
|  |  | **pos của xâu (vị trị** | **thì chỉ xóa chiều** |
|  |  | **các phần tử đánh số** | **dài có thể của xâu.** |
|  |  | **từ 0)** |  |
|  |  |  |  |
| Ví dụ | |  |  |

* string::erase

#include <iostream> #include <string> Using namespace std; int main ()

{

string str ("This is an example sentence."); cout << str << '\n';

str.erase (10,8); cout << str << '\n'; str.erase (str.begin()+9); cout << str << '\n';

str.erase (str.begin()+5, str.end()-9); cout << str << '\n';

return 0;

}

Ouput

This is an example sentence. This is an sentence.

This is a sentence. This sentence.

**2. Chèn một xâu vào xâu khác** insert(pos,str)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Tên hàm** | **Tiếng Anh** | **Chức năng** | **Ghi chú** |  |
|  |  |  |  |  |  |
|  | **insert(pos,str)** | **Insert là** | **Chèn xâu str vào** | **Nếu vị trí cần** |  |
|  |  | **chèn** | **xâu đã cho ngay** | **chèn vƣợt quá vị** |  |
|  |  | **Pos là** | **trƣớc vị trí pos của** | **trí có thể của xâu,** |  |
|  |  | **position vị** | **xâu đó** | **một lỗi sai có thể** |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  | *Đặng Trung Kiên* | |  | *Trang 89* |  |

*Ngôn ngữ lập trình C++*



|  |  |
| --- | --- |
| **trí** | **xảy ra** |

**Str là viết**

**tắt tên**

**biến xâu**

**string**

* inserting into a string #include <iostream> #include <string>

int main ()

{

std::string str="to be question";

std::string str2="the ";

std::string str3="or not to be";

std::string::iterator it;

// used in the same order as described above:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| str.insert(6,str2); |  | // to be (the )question | | | | |  |
| str.insert(6,str3,3,4); | |  | // | to | be | (not )the | question |
| str.insert(10,"that | is cool",8); | | // | to | be | not (that | is )the |

question

str.insert(10,"to be "); // to be not (to be )that is the question str.insert(15,1,':'); // to be not to be(:) that is the question it = str.insert(str.begin()+5,',');// to be(,) not to be: that is

the question

str.insert (str.end(),3,'.'); // to be, not to be: that is the question(...)

str.insert (it+2,str3.begin(),str3.begin()+3); // (or )

std::cout << str << '\n';

return 0;

}



*Đặng Trung Kiên* *Trang 90*

*Ngôn ngữ lập trình C++*



Ví dụ: string s;

1. **Gán một nội dung mới tới một xâu (copy nội dung mới tới một xâu)**

|  |  |  |
| --- | --- | --- |
| **Tên hàm** | **Chức năng** | **Ghi chú** |
|  |  |  |
| **assign(str,subpos,sublen)** | **Str-xâu đƣợc** | **Nếu subpos (vị trí copy** |
|  | **copy** | **này lớn hơn độ dài của** |
|  | **Subpos-vị trí** | **xâu str, một lỗi sai sẽ xảy** |
|  | **copy** | **ra)** |
|  | **Sublen-chiều** |  |
|  | **dài cần copy** |  |
|  | **Xâu đƣợc gán** |  |
|  | **nội dung mới** |  |
|  | **từ xâu str,** |  |
|  | **copy sublen kí** |  |
|  | **tự từ vị trí** |  |
|  | **subpos** |  |
|  |  |  |
| **Ví dụ** |  |  |

* string::assign #include <iostream> #include <string>

int main ()

{

std::string str;

std::string base="The quick brown fox jumps over a lazy dog."; // used in the same order as described above:

str.assign(base);

std::cout << str << '\n';

|  |  |
| --- | --- |
| str.assign(base,10,9); |  |
| std::cout << str << '\n'; | // "brown fox" |
| str.assign("pangrams are cool",7); |  |
| std::cout << str << '\n'; | // "pangram" |
| str.assign("c-string"); |  |
| std::cout << str << '\n'; | // "c-string" |
| str.assign(10,'\*'); |  |
| std::cout << str << '\n'; | // "\*\*\*\*\*\*\*\*\*\*" |
| str.assign<int>(10,0x2D); |  |
|  |  |
|  |  |
| *Đặng Trung Kiên* | *Trang 91* |

*Ngôn ngữ lập trình C++*



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | std::cout << str << '\n'; | | // "----------" | |  |  |
|  |  | str.assign(base.begin()+16,base.end()-12); | | | | | |
|  |  | std::cout << str << '\n'; | | // "fox jumps over" | | | |
|  |  | return 0; |  |  |  |  |  |
| } | | |  |  |  |  |  |
|  | **Ouput** | |  |  |  |  |  |
|  | The quick brown fox jumps over a lazy dog. | | | |  |  |  |
|  | brown fox | |  |  |  |  |  |
|  | pangram | |  |  |  |  |  |
|  | c-string | |  |  |  |  |  |
| \*\*\*\*\*\*\*\*\*\* | | |  |  |  |  |  |
| ---------- | | |  |  |  |  |  |
|  | fox jumps over | |  |  |  |  |  |
|  | **4. Tìm nội dung trong một xâu** | | | |  |  |  |
|  |  | **Tên hàm** |  | **Chức năng** | **Ghi chú** | | |
|  |  |  |  | |  |  |  |
|  | **find(str,pos)** | | **Tìm xâu str** | | **Nếu không tìm thấy, kết** | | |
|  |  |  | **trong xâu đã** | | **quả trả về -1, hay một số** | | |
|  |  |  | **cho, kể từ vị** | | **nguyên gọi là** | | |
|  |  |  | **trí thứ pos của** | | **string::npos (số nguyên** | | |
|  |  |  | **xâu đã cho** | | **này mặc định là -1)** | | |
|  |  |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | // string::find |  |  |  |  |  |
|  |  | #include <iostream> | // std::cout | |  |  |  |
|  |  | #include <string> | // std::string | |  |  |  |
|  |  | using namespace std; |  |  |  |  |  |
|  |  |  |  |  |  |  |  |



int main ()



{



string str ("There are two needles in this haystack with needles.");



string str2 ("needle");



* different member versions of find in the same order as above: size\_t found = str.find(str2);

if (found!=string::npos)

cout << "first 'needle' found at: " << found << '\n'; found=str.find("needles are small",found+1,6);

if (found!=string::npos)

cout << "second 'needle' found at: " << found << '\n'; found=str.find("haystack");

if (found!=string::npos)

cout << "'haystack' also found at: " << found << '\n'; found=str.find('.');

if (found!=string::npos)

cout << "Period found at: " << found << '\n';

* let's replace the first needle: str.replace(str.find(str2),str2.length(),"preposition");



cout << str << '\n';



return 0;



}



*Đặng Trung Kiên* *Trang 92*

*Ngôn ngữ lập trình C++*



Ouput

first 'needle' found at: 14

second 'needle' found at: 44

'haystack' also found at: 30

Period found at: 51

There are two prepositions in this haystack with needles.

**5. Tạo một xâu mới là xâu con của một xâu đã cho**

substr(pos,len)

|  |  |  |
| --- | --- | --- |
| **Tên hàm** | **Chức năng** | **Ghi chú** |
|  |  |  |
| **substr(pos,len)** | **Tạo một xâu con mới** | **Nếu vị trí thứ pos của** |
|  | **của xâu đã cho, bằng** | **xâu đó lớn hơn vị trí có** |
|  | **cách copy len kí tự kể từ** | **thể có của xâu đã cho, thì** |
|  | **vị trí thứ pos của xâu đó** | **một lỗi sai sẽ xảy ra** |
|  |  | **Nếu độ dài lớn hơn độ** |
|  |  | **dài của xâu đó, thì chỉ** |
|  |  | **copy độ dài hiện tại của** |
|  |  | **xâu đó** |
|  |  |  |

* string::substr #include <iostream> #include <string> using namespace std; int main ()

{

string str="We think in generalities, but we live in details.";

* + (quoting Alfred N.

Whitehead)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| string str2 = str.substr | | (3,5); | // "think" | |  |  |
| size\_t | pos = str.find("live"); | | // | position | of "live" | in str |
| string | str3 = str.substr | (pos); | // | get from | "live" to | the end |

cout << str2 << ' ' << str3 << '\n';

return 0;

}

Ouput

think live in details.



*Đặng Trung Kiên* *Trang 93*

*Ngôn ngữ lập trình C++*



**3. Cách chuyển từ xâu sang số, từ số sang xâu**

C++ 11 mới có hàm chuyển đổi từ xâu sang số, và từ số sang xâu dễ dàng như Pascal. Tuy nhiên C++ 11 chưa có khả năng chấm được trên Themis. Do vậy, ta phải đi một con đường vòng khi muốn biến đổi qua lại từ số sang xâu, từ xâu sang số

**a) Chuyển từ kí tự sang số**

Phép trừ giữa hai kí tự thực chất là phép trừ giữa hai số thứ tự trong bộ mã ASCII của hai kí tự đó

Ví dụ ‗A‘-‗a‘=-32 vì số thứ tự trong bộ mã ASCII của kí tự ‗A‘ là 65, trong khi số thứ tự trong bộ mã ASCII của kí tự ‗a‘ là 97, do vậy 65-97 chính bằng -32

Lợi dụng điều này, ta có thể cài đặt trên C++ như sau:

* Đối với một mảng các chữ cái, ta sử dụng một mảng gồm 26 phần tử được đánh số từ 0 tới 25, ví dụ a[25], vậy thì có thể coi a[0]=a[‗a‘-‗a‘];a[1]=a[‗b‘-‗a‘], a[2]=a[‗c‘-‗a‘]
* Một kí tự số muốn chuyển thành số ta chỉ việc trừ cho kí tự ‗0‘ Ví dụ ‗9‘-‗0‘=9, ‗5‘-‗0‘=5

Như vậy, có 2 cách chuyển từ kí tự sang số:

Cách 1. Trừ cho kí tự ‗0‘

Ví dụ: ‗9‘-‗0‘=9

Cách 2. Lấy luôn kí tự từ đi số 48, số 48 chính là số thứ tự trong bảng mã ASCII của ‗0‘

Ví dụ string s=‖9321‖

cout<<s[0]-48

Vậy thì ngược lại, chuyển từ chữ số sang dạng kí tự thì như thế nào? **b) Chuyển từ chữ số sang dạng kí tự**

Cách 1. Cộng số đó với kí tự ‗0‘, gán vào kí tự cần chuyển int main()

{ int x=2; char c=x+'0';

cout<<c<<'\n'; //dua ra gia tri la ky tu 2 cout<<c+3;//dua ra gia tri la so 53 return 0;

}

Cách 2. Cộng số đó với 48, gán vào kí tự cần chuyển

1. **Chuyển từ mảng các kí tự thành số (thƣ viện stdlib.h)**
   * atoi(str) chuyển đổi mảng các kí tự thành một số int
   * atol(str) chuyển đổi mảng các kí tự thành một số long
   * atoll(str) chuyển đổi mảng các kí tự thành một số long long
   * atof(str) chuyển đổi mảng các kí tự thành một số float

Ví dụ:



*Đặng Trung Kiên* *Trang 94*

*Ngôn ngữ lập trình C++*



int main()

{long long a;

char x[21]="1000000000000000000"; a=atoll(x);

cout<<a;

return 0;}

1. **Chuyển từ số sang mảng các kí tự (thƣ viện stdlib.h)**
   * itoa(value, str, cơ số) chuyển đổi số int thành mảng các kí tự
   * lltoa(value, str, cơ số) chuyển đổi số long long thành mảng các kí tự

int main()

{long long a=1234500000000000000; char x[21];

lltoa(a,x,10);

cout<<x; return 0;

}

1. **Chuyển từ xâu thành số trong C++**
   * Cách 1. Chuyển từ xâu thành mảng các kí tự, rồi từ mảng các kí tự đó dùng hàm để chuyển thành số

(Để chuyển xâu thành mảng các kí tự ta dùng hàm **s.c\_str(), với s**

**là biến xâu)**

* Cách 2. Dùng đến lớp stringstream

**Cách 1.**

#include<stdlib.h> using namespace std; int main()

{

long long a;

string s="1000000000000000000"; a=atoll(s.c\_str()); // chuyển từ xâu sang

số

cout<<a; return 0;

}

**Cách 2. Dùng đến lớp stringstream**

#include <iostream>

#include<string>

#include<stdlib.h>

#include<sstream>

using namespace std;

int main()

{ stringstream ss;



*Đặng Trung Kiên* *Trang 95*

*Ngôn ngữ lập trình C++*



string s="1000000000000000000"; long long x;

ss<<s;//Lấy dữ liệu từ xâu s, đưa ra bộ nhớ đệm stream

ss>>x;//Đưa dữ liệu vào biến x từ bộ nhớ đệm stream

cout<<x;

return 0;

}

Ouput lúc này x sẽ nhận giá trị là 1000000000000000000.

**Cách này rất nhanh, tuy nhiên có nhƣợc điểm là sử dụng việc biến ss lại lần thứ hai để biến đổi từ xâu sang số lại không đƣợc. Ví dụ:**

#include<stdlib.h>

#include<sstream>

using namespace std;

int main()

{ stringstream ss; string s="10000000"; string s1="345"; long long x,y; ss<<s;

ss>>x;

cout<<x<<'\n';

ss<<s1;

ss>>y;

cout<<y;

return 0;

}

Lúc này y không bằng 345 như chúng ta mong đợi, do bộ nhớ đệm của ss còn chứa dữ liệu ban đầu của xâu s. Do vậy ta phải có động tác làm sạch bộ nhớ đệm này như sau, dùng cả hai lệnh ss.str(―‖) và thiết lập lại trạng thái của stream ở vị trí bắt đầu bộ nhớ đệm, chứ không ở vị trí kết thúc sau lệnh ss>>x, do đó ta dùng ss.clear()

Nếu vẫn cảm thấy khó hiểu, có thể tham khảo <http://stackoverflow.com/questions/2848087/how-to-clear-stringstream> <http://forums.codeguru.com/showthread.php?292719-How-to-clear-out-a-stringstream>

#include<stdlib.h>

#include<sstream>

using namespace std;

int main()



*Đặng Trung Kiên* *Trang 96*

*Ngôn ngữ lập trình C++*



{ stringstream ss; string s="10000000"; string s1="345"; long long x,y; ss<<s;

ss>>x;

cout<<x<<'\n';

ss.str("");

ss.clear();

ss<<s1;

ss>>y;

cout<<y;

return 0;

}

1. **Chuyển từ số sang xâu trong C++**
   * Cách 1. Chuyển từ số sang mảng các kí tự (dùng hàm), rồi chuyển mảng các kí tự này thành xâu lúc khai báo biến xâu
   * Cách 2. Dùng đến lớp stringstream

#include <iostream>

#include<string>

#include<stdlib.h>

using namespace std;

int main()

{ long long a=1000000000000000000; char x[21]; // mang cac ki tu lltoa(a,x,10);//bien doi so thanh mang

cac ki tu

string s(x);//khoi tao va chuyen mang ki tu thanh xau

cout<<s;

return 0;

}

**Cách 2. Dùng lớp stringstream**

* **Chuyển từ số sang xâu, dùng thƣ viện <sstream>**

#include<sstream> using namespace std; int main()

{ stringstream ss; string s;

long long n = 1230000000000000000;

1. << n; ss >> s;



*Đặng Trung Kiên* *Trang 97*

*Ngôn ngữ lập trình C++*



cout<<s;

return 0;

}

**Cách này rất nhanh, tuy nhiên có nhƣợc điểm là sử dụng việc biến ss lại lần thứ hai để biến đổi từ số sang xâu lại không đƣợc. Ví dụ:** #include<sstream>

#include<string>

#include<iostream>

using namespace std;

int main()

{ stringstream ss; string s,s1;

long long n = 1230000000000000000; ss<<n;

ss>>s;

cout<<s<<'\n';

long long n1=345;

ss<<n1;

ss>>s1;

cout<<s1;

return 0;

}

* đây xâu s1 không bằng ―345‖ như mong đợi, do bộ nhớ đệm của ss còn chứa dữ liệu ban đầu của n. Do vậy ta phải có động tác làm sạch bộ nhớ đệm này như sau, làm sạch bộ nhớ đệm stream, ss.str(―‖) và thiết lập lại trạng thái của stream ở vị trí bắt đầu bộ nhớ đệm, chứ không ở vị trí kết thúc sau lệnh ss>>s, do đó ta dùng ss.clear()

Nếu vẫn cảm thấy khó hiểu, có thể tham khảo <http://stackoverflow.com/questions/2848087/how-to-clear-stringstream> <http://forums.codeguru.com/showthread.php?292719-How-to-clear-out-a-stringstream>

#include<sstream>

#include<string>

#include<iostream> using namespace std; int main()

{ stringstream ss; string s,s1;

long long n = 1230000000000000000; ss<<n;

ss>>s;

cout<<s<<'\n';

ss.str("");



*Đặng Trung Kiên* *Trang 98*

*Ngôn ngữ lập trình C++*



ss.clear();

long long n1=345;

ss<<n1;

ss>>s1;

cout<<s1;

return 0;

}

Lúc này kết quả hiển thị 345 như mong đợi ở s1.



*Đặng Trung Kiên* *Trang 99*

*Ngôn ngữ lập trình C++*



**Bài tập về xâu**

**Lạ nhỉ, số lại biến thành xâu, xâu lại thành số?**

**Bài 1.** Hãy lập trình:

* Nhập một xâu S chỉ chứa các kí tự Latinh in thường và các kí tự số từ 0 đến 9.
* Đếm và đưa ra màn hình số kí tự khác nhau trong xâu S. Ví dụ, nếu S=‖abcabcaab‖, số kí tự khác nhau là 3
* Nhập xâu bất kỳ từ bàn phím
* Chuẩn hóa xâu theo các quy tắc sau:
  + Xóa các dấu cách ở đầu xâu nếu có
  + Xóa các dấu cách ở cuối xâu nếu có
  + Thay dãy nhiều dấu cách liên tiếp bằng một dấu cách
* Đưa kết quả đã chuẩn hóa ra màn hình

**Bài 3**. Xâu không chứa các kí tự phân cách như dấu chấm, dấu phẩy, dấucách được gọi là một từ. Câu là một xâu bao gồm một hoặc nhiều từ, nếu xâu có nhiều từ thì các từ được phân cách với nhau bởi một hoặc một số kí tự phân cách.

Ví dụ: Theo định nghĩa trên ta có: ―Informatics‖- câu có một từ ―Olympiad in Informatics‖-câu có ba từ;

―Co anh ban xa nha, co co ban nho cha.‖-câu có 10 từ Viết chương trình tính số lượng từ trong câu

**Bài 4**. Lập trình nhập số nguyên dương N từ bàn phím và đưa ra mànhình dạng biểu diễn của nó ở hệ đếm cơ số 16 (hecxa). Ví dụ, với N=446, kết quả đưa ra màn hình là 1BE.

**Bài 5.** Một số nguyên N ở hệ thập phân có chữ số cuối cùng là số chẵn thìsẽ chia hết cho 2, nếu tổng các chữ số của N chia hết cho 3 thì N sẽ chia hết cho 3. Cho số nguyên thập phân N không quá 60 chữ số. Hãy xác định xem N có chia hết cho 6 hay không và đưa ra màn hình kết quả kiểm tra ―CO‖ hoặc ―KHONG‖

**Bài 6.** Với số nguyên n cho trước ở hệ thập phân người ta có thể tạo ranhững số nguyên khác nhau bằng cách hoán vị các chữ số của n. Ví dụ, với n=133, ta có thể có các số 313 và 331. Hãy:

1. Xác định kiểu dữ liệu và biến cần dùng;
2. Viết đoạn chương trình xác định số lớn nhất có thể tạo được.



*Đặng Trung Kiên* *Trang 100*

*Ngôn ngữ lập trình C++*



**BÀI 13. KIỂU CẤU TRÚC TRONG C++ (STRUCT)**

(giống kiểu bản ghi trong Pascal)

1. Một cấu trúc dữ liệu là một nhóm các biến được nhóm lại với nhau dưới cùng một tên. Các biến có thể có các kiểu dữ liệu khác nhau. struct tên kiểu {

kiểu dữ liệu 1 tên biến 1; kiểu dữ liệu 2 tên biến 2; kiểu dữ liệu 3 tên biến 3;

.

.

} tên biến;

Trong đó: tên kiểu là tên cho kiểu của cấu trúc

Tên biến là tên của biến có kiểu dữ liệu là tên kiểu

Ví dụ

struct product {

int weight;

double price;

} ;

product apple;

product banana, melon;

Đoạn chương trình trên khai báo một kiểu sản phẩm, gồm các thành phần là cân nặng, giá cả. Biến Apple (táo) có kiểu là sản phẩm

**Ta truy vấn đến từng thành phần bằng dấu chấm:**

Ví dụ

apple.weight

apple.price

banana.weight

banana.price

melon.weight

melon.price

**2. Gán giá trị**

Có hai cách dùng để gán tía trị cho biến cấu trúc:

* Dùng lệnh gán trực tiếp

Nếu A và B là hai biến cấu trúc cùng kiểu, thì ta có có thể gán giá

trị của B cho A bằng câu lệnh A=B

* Gán giá trị cho từng thành phần: Có thể thực hiện bằng lệnh gán

hoặc nhập từ bàn phím

Ví dụ, một lớp gồm N (N≤60) học sinh. Cần quản lý học sinh với các thuộc tính như họ và tên, ngày sinh, địa chỉ điểm toán, điểm văn, xếp loại. Giả sử việc xếp loại được xác định như sau:



*Đặng Trung Kiên* *Trang 101*

*Ngôn ngữ lập trình C++*



* Nếu tổng điểm toán và văn lớn hơn hoặc bằng 18 thì xếp loại A.
* Nếu tổng điểm toán và văn lớn hơn hoặc bằng 14 và nhỏ hơn 18 thì xếp loại B.
* Nếu tổng điểm toán và văn lớn hơn hoặc bằng 10 và nhỏ hơn 14 thì xếp loại C
* Nếu tổng điểm toán và văn nhỏ hơn 10 thì xếp loại D.

Chú ý rằng, trong các thuộc tính cần quản lý, chỉ có năm thuộc tính đầu là độc lập, còn thuộc tính xếp loại được xác định dựa vào các điểm toán và văn. Để lưu trữ thông tin về học sinh, ta dùng kiểu bản ghi với sáu trường tương ứng với sáu thuộc tính cần quản lý.

Dưới đây là chương trình nhập và từ bàn phím thông tin của từng học sinh trong lớp, thực hiện xếp loại và đưa ra màn hình kết quả xếp loại của học sinh:

#include <iostream>

#include<string>

#include<stdio.h>

using namespace std;

const long max1=60;

struct hocsinh

{

string hoten,ngaysinh,diachi; double toan,van; char xeploai;

};

hocsinh lop[max1];

long n,i;

int main()

{ cout<<"so luong hoc sinh trong lop n="; cin>>n;

fflush(stdin); for(long i=0;i<n;i++)

{

cout<<"Nhap so lieu ve hoc sinh thu"<<i+1<<":"<<'\n';

cout<<"ho va ten:"; getline(cin,lop[i].hoten); cout<<"Ngay sinh:"; getline(cin,lop[i].ngaysinh); cout<<"Dia chi:"; getline(cin,lop[i].diachi); cout<<"diem toan:"; cin>>lop[i].toan; cout<<"diem van:";



*Đặng Trung Kiên* *Trang 102*

*Ngôn ngữ lập trình C++*



cin>>lop[i].van; if(lop[i].toan+lop[i].van>=14 &&

lop[i].toan+lop[i].van<18)

lop[i].xeploai='B'; if(lop[i].toan+lop[i].van>=10 &&

lop[i].toan+lop[i].van<14)

lop[i].xeploai='C';

if(lop[i].toan+lop[i].van<10)

lop[i].xeploai='D';

fflush(stdin);

}

cout<<"Danh sach xep loai hoc sinh trong

lop"<<'\n';

for(i=0;i<n;i++)

{ cout<<lop[i].hoten<<'\n';

cout<<"xep loai:"<<lop[i].xeploai<<'\n';

}

return 0;

}



*Đặng Trung Kiên* *Trang 103*

*Ngôn ngữ lập trình C++*



**BÀI 14. KIỂU DỮ LIỆU TỆP *1. Vai trò của kiểu tệp***

Tất cả các dữ liệu thuộc các kiểu dữ liệu đã xét đều được lưu trữ ở bộ nhớ trong (RAM) và do đó dữ liệu sẽ bị mất khi tắt máy. Với một số bài toán có khối lượng dữ liệu lớn, có yêu cầu lưu trữ để xử lý nhiều lần, cần có kiểu dữ liệu tệp (file).

Kiểu dữ liệu tệp có những đặc điểm sau:

* + Dữ liệu kiểu tệp được lưu trữ lâu dài ở bộ nhớ ngoài (đĩa từ, CD,…) và không bị mất khi tắt nguồn điện
  + Lượng dữ liệu lưu trữ trên tệp có thể rất lớn và chỉ phụ thuộc vào dung lượng đĩa

1. ***Phân loại tệp***

Xét theo cách tổ chức dữ liệu, có thể phân tệp thành hai loại:

* Tệp văn bản: là tệp mà dữ liệu dược ghi dưới dạng các kí tự theo mã ASCII. Trong tệp văn bản, dãy ký tự kết thúc bởi ký tự xuống dòng hay ký tự kết thúc tệp tạo thành một dòng. Các dữ liệu dạng văn bản như sách, tài liệu, bài học, giáo án, các chương trình nguồn viết bằng ngôn ngữ bậc cao,… thường được lưu trữ dưới dạng tệp văn bản.
* Tệp có cấu trúc là tệp mà các thành phần của nó được tổ chức theo một cấu trúc nhất định. Tệp nhị phân là một trường hợp riêng của tệp có cấu trúc. Dữ liệu ảnh, âm thanh,… thường được lưu trữ dưới

dạng tệp có cấu trúc.

Xét theo cách thức truy cập, có thể phân tệp thành hai loại:

* Tệp truy cập tuần tự cho phép truy cập đến một dữ liệu nào đó trong tệp chỉ bằng cách bắt đầu từ đầu tệp và đi qua lần lượt tất cả các dữ liệu trước nó.



*Đặng Trung Kiên* *Trang 104*

*Ngôn ngữ lập trình C++*



* Tệp truy cập trực tiếp cho phép tham chiếu đến dữ liệu cần truy cập bằng cách xác định trực tiếp vị trí (thường là số hiệu) của dữ liệu đó

***Khác với mảng, số lượng phần tử của tệp không cần xác định trước Hai thao tác cơ bản đối với tệp là ghi dữ liệu vào tệp và đọc dữ liệu từ tệp***

***Thao tác đọc/ghi với tệp được thực hiện với từng phần tử của tệp***

Để có thể thao tác với kiểu tệp, người lập trình cần tìm hiểu cách thức mà ngôn ngữ lập trình cung cấp để:

* Khai báo biến tệp
* Mở tệp
* Đọc/ghi dữ liệu
* Đóng tệp



*Đặng Trung Kiên* *Trang 105*

*Ngôn ngữ lập trình C++*



**BÀI 15. THAO TÁC VỚI TỆP**

Trong mục này, ta xét cách khai báo, thao tác với tệp văn bản trong

C++

***1. Thao tác với tệp***

**Sử dụng thƣ viện #include<stdio.h>**

**a) Mở tệp để đọc**

***Cú pháp:*** ***freopen(<tên file>, “r”, stdin);***

Ví dụ: freopen(―xau.inp‖,‖r‖,stdin);

r là viết tắt của read

Thao tác này mở file xau.inp để đọc dữ liệu. File này phải tồn tại. **b) Mở tệp để ghi**

**Cú pháp:** ***freopen(<tên file>, “w”, stdout);***

Ví dụ: freopen(―xau.out‖, ―w‖, stdout);

w là viết tắt của write

Thao tác này mở file xau.out để tiến hành ghi dữ liệu **c) Đọc/ghi tệp văn bản**

Việc đọc tệp văn bản được thực hiện giống như nhập từ bàn phím. Việc ghi dữ liệu ra tệp văn bản giống như ghi ra màn hình. Dữ liệu trong tệp văn bản được chia thành các dòng. Câu lệnh dùng hàm đọc có dạng:

**cin>>biến>>biến>>…..>>biến;**

Câu lệnh dùng hàm ghi có dạng:

**Cout<<kết quả ra<<kết quả ra<<…<<kết quả ra;**

Muốn ghi thành nhiều dòng, ta có thể dùng endl hoặc ‗\n‘ ở cuối lệnh

**Cout<<kết quả ra<<kết quả ra<<…<<kết quả ra<<endl;**

**Cout<<kết quả ra<<kết quả ra<<…<<kết quả ra<<‟\n‟;**

**d) Đóng tệp**

Cú pháp:



*Đặng Trung Kiên* *Trang 106*

*Ngôn ngữ lập trình C++*



fclose(stdin) // với thao tác mở tệp để đọc flose(stdout) // với thao tác mở tệp để ghi



*Đặng Trung Kiên* *Trang 107*

*Ngôn ngữ lập trình C++*



**BÀI 16. VÍ DỤ LÀM VIỆC VỚI TỆP**

**Ví dụ 1.** Một trường trung học phổ thông tổ chức cho giáo viên và họcsinh của trường đi cắm trại, sinh hoạt ngoài trời ở vườn quốc gia Cúc Phương. Để lên lịch đến thăm khu trại các lớp, thầy hiệu trưởng cần biết khoảng cách từ trại của mình (ở vị trí có tọa độ (0,0)) đến trại của giáo viên chủ nhiệm. Mỗi lớp có một khu trại, vị trí trại của mỗi giáo viên chủ nhiệm đều có tọa độ nguyên (x,y) được ghi trong tệp văn bản TRAI.INP (như vậy, tệp TRAI.INP chứa liên tiếp các cặp số nguyên, các số cách nhau bởi dấu cách và không kết thúc bằng kí tự xuống dòng).

Chương trình sau sẽ đọc các cặp tọa độ từ tệp TRAI.INP, tính rồi ghi ra tệp TRAI.OUT khoảng cách (với độ chính xác hai chữ số sau dấu chấm thập phân) giữa trại của mỗi giáo viên chủ nhiệm và trại của thầy hiệu trưởng.

|  |  |
| --- | --- |
| TRAI.INP | TRAI.OUT |
|  |  |
| 3 4 | 5.00 |
| 6 8 | 10.00 |
| 1 1 | 1.41 |
|  |  |

Trước hết, ta cần tạo ra hai file TRAI.INP và TRAI.OUT bằng cách sau. Từ phần mềm Codeblock, ta vào File->New->Empty file (hoặc nhấn tổ hợp phím Ctrl+Shift+N) , một bảng hộp thoại hiện ra, trong ô File Name ta đặt tên file là TRAI.INP, trong ô ***save as type*** ta chọn All Files. Nhấn Save. Khi đó ta sẽ thấy tệp TRAI.INP trên màn hình. Nhập dữ liệu cho tệp này, save lại.

Tạo tệp TRAI.OUT cũng tương tự. Chạy thử chương trình dưới đây để xem dữ liệu có đưa ra tệp TRAI.OUT hay không.

#include<bits/stdc++.h>

using namespace std;

double d;



*Đặng Trung Kiên* *Trang 108*

*Ngôn ngữ lập trình C++*



long x,y;

int main()

{ freopen("TRAI.INP","r",stdin); //mo tep TRAI.INP de doc du lieu freopen("TRAI.OUT","w",stdout); //mo tep TRAI.OUT de ghi du lieu

while(cin>>x>>y)

{

d=sqrt(x\*x+y\*y);//d la khoang cach cout<<fixed<<setprecision(2); cout<<d<<'\n';

}

return 0;

}

while (cin>>x>>y) nghĩa là trong khi việc đọc dữ liệu ra hai biến x và y còn đúng. Cin>>x>>y sẽ trả về giá trị true nếu đọc dữ liệu đúng, trả về giá trị false nếu đọc dữ liệu sai (nghĩa là không còn dữ liệu để đọc)



*Đặng Trung Kiên* *Trang 109*

*Ngôn ngữ lập trình C++*



**CHƢƠNG VI. CHƢƠNG TRÌNH CON VÀ LẬP TRÌNH CÓ CẤU TRÚC**

**BÀI 17. CHƢƠNG TRÌNH CON VÀ PHÂN LOẠI** *1. Khái niệm chương trình con (hàm trong C++)*

Chương trình con là một dãy lệnh mô tả một số thao tác nhất định và có thể được thực hiện (được gọi) từ nhiều vị trí trong chương trình.

Cú pháp khai báo một hàm:

**a) Đối với hàm có trả về một giá trị**

Kiểu dữ liệu tên hàm (tham số 1, tham số 2,…) { các câu lệnh;

return giá trị trả về của hàm;

}

Ví dụ về hàm

* function example #include <iostream> using namespace std;

int addition (int a, int b)

{

int r;

r=a+b;

return r;

}

int main ()

{

int z;

z = addition (5,3);

cout << "The result is " << z;

}

Ví dụ 2:

* function example #include <iostream> using namespace std;

int subtraction (int a, int b)

{

int r;

r=a-b;

return r;



*Đặng Trung Kiên* *Trang 110*

*Ngôn ngữ lập trình C++*



}

int main ()

{

int x=5, y=3, z;

z = subtraction (7,2);

cout << "The first result is " << z << '\n'; cout << "The second result is " << subtraction

(7,2) << '\n';

cout << "The third result is " << subtraction (x,y) << '\n';

z= 4 + subtraction (x,y);

cout << "The fourth result is " << z << '\n';

}

1. **Đối với hàm không trả về một giá trị - hàm void**

**Cú pháp:**

Kiểu dữ liệu tên hàm (tham số 1, tham số 2,…)

{ các câu lệnh;

}

// void function example #include <iostream> using namespace std;

void printmessage ()

{

cout << "I'm a function!";

}

int main ()

{

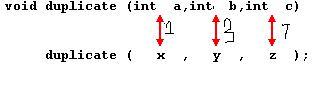
printmessage ();

}

*2. Tham trị và tham chiếu*

1. Truyền tham số bằng tham trị

Sự thay đổi của a,b,c trong hàm duplicate không kéo theo sự thay đổi của x,y,z, giá trị của x,y,z sẽ được truyền cho a,b,c tương ứng



*Đặng Trung Kiên* *Trang 111*

*Ngôn ngữ lập trình C++*



#include <iostream>

using namespace std;

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

{

a\*=2;

b\*=2;

c\*=2;

}

int main ()

{

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

duplicate (x, y, z);

cout << "x=" << x << ", y=" << y << ", z=" << z; return 0;

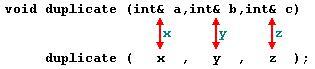
}

Output

x=1,y=3,z=7

1. Truyền tham số bằng tham chiếu (thêm dấu & trước tham số trong hàm)

Sự thay đổi của a,b,c trong hàm dubplicate kéo theo sự thay đổi của x,y,z tương ứng, x đồng



* passing parameters by reference #include <iostream>

using namespace std;

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

{

a\*=2;

b\*=2;

c\*=2;

}

int main ()

{

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

duplicate (x, y, z);

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



*Đặng Trung Kiên* *Trang 112*

*Ngôn ngữ lập trình C++*



return 0;

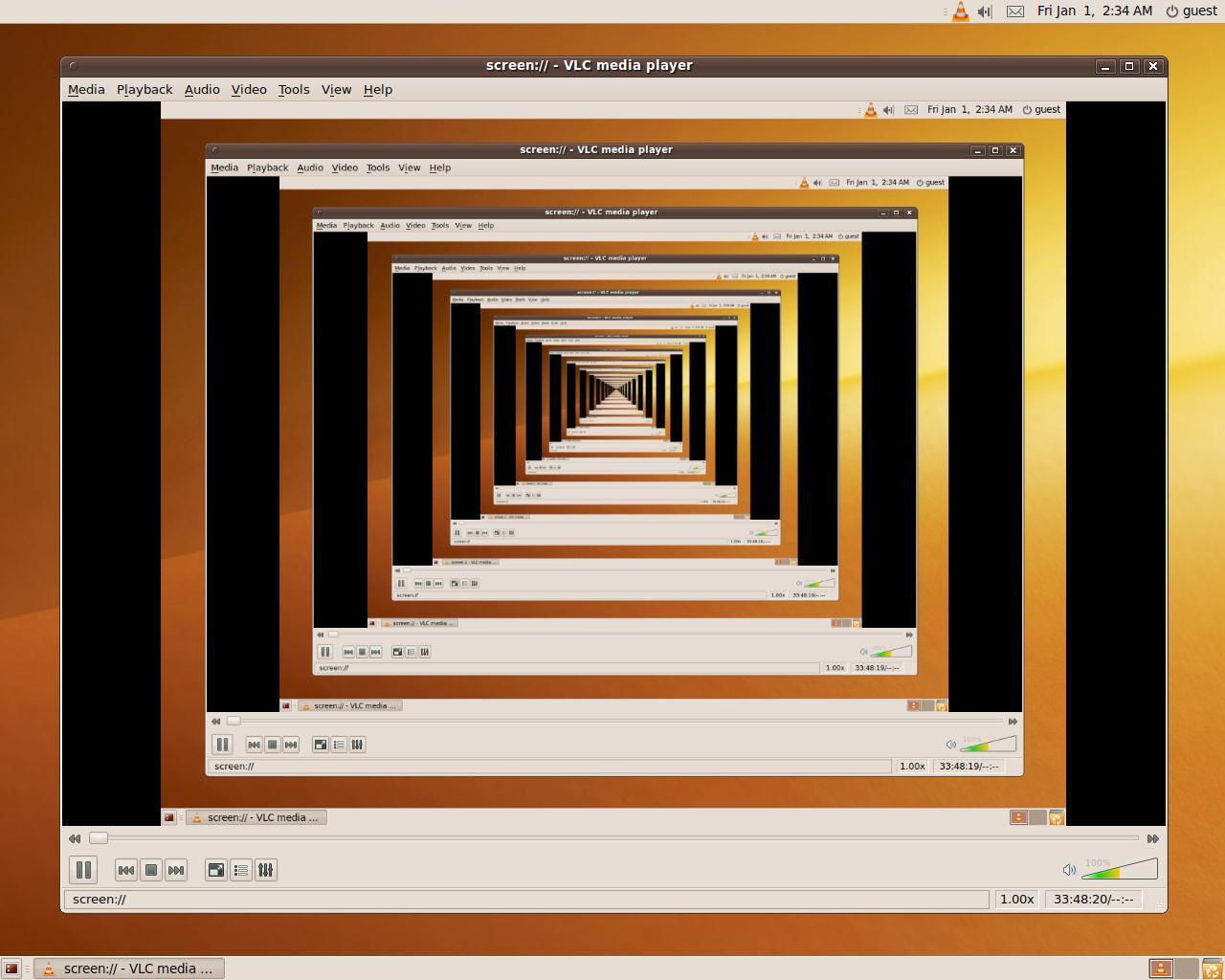
}

Output

x=2,y=6,z=14

***3. Đệ quy***

**Hiểu một cách đơn giản, đệ quy là một phƣơng pháp trong đó có sử dụng hàm sẽ gọi đến chính nó Hình ảnh về đệ quy**



**Ví dụ 1.**

**Tính** *n*!=*n*(*n*−1).....1

* factorial calculator #include <iostream> using namespace std;

long factorial (long a)

{

if (a > 1)



*Đặng Trung Kiên* *Trang 113*

*Ngôn ngữ lập trình C++*



return (a \* factorial (a-1)); else

return 1;

}

int main ()

{

long number = 9;

cout << number << "! = " << factorial (number); return 0;

}

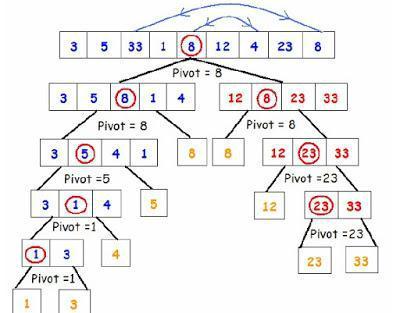
**Ví dụ 2. Thuật toán sắp xếp nhanh Quicksort** Xét dãy n phần tử a1, a2, a3, .... an.

**Bƣớc 1:** Chọn khóa pivot (chốt):a(Left + Righ)/ 2.

**Bƣớc 2:** Phân vùng. Những phần tử nhỏ hơn khóa thì nằm bên trái củakhóa, những phần tử lớn hơn khóa thì nằm bên phải của khóa và những phần tử bằng khóa có thể nằm bất cứ chỗ nào trên dãy.

**Bƣớc 3:** Sắp xếp cho cả hai phân vùng mới bên trái và bên phải.

* **Mô tả hoạt động của thuật toán Quick Sort:**



**Đánh giá thuật toán**

Quick sort - Thuật toán sắp xếp nhanh

Hiệu suất của quick sort O(n):

* Tốt nhất: n log n
* Tệ nhất: n^2



*Đặng Trung Kiên* *Trang 114*

*Ngôn ngữ lập trình C++*



Ưu điểm: Tuỳ cách chọn pivot mà tốc độ của thuật toán nhanh hay chậm

Nhược điểm: Code khá phức tạp

* **Cài đặt thuật toán**

**Thuật toán sắp xếp nhanh - Quick Sort #include <iostream.h> #include <conio.h>**

**#define max 100**

***// nhap mang***

**void NhapMang(int A[],int n)**

**{**

**for(int i=0; i<n; i++)**

**{**

**cout<<"nhap Phan tu thu A["<<i<<"] ="; cin>>A[i];**

**}**

**}**

***// in mang***

**void XuatMang(int A[],int n)**

**{**

**cout<<endl;**

**for(int i=0; i<n; i++)**

**cout<<A[i]<<"\t";**

**}**

***// doi cho 2 so***

**void Swap(int &a,int &b)**

**{**

**int temp = a;**

**a = b;**

**b = temp;**

**}**

***// thuat toan quick-sort***

**void QuickSort(int A[], int Left, int Right)**

**{**

**int i = Left, j = Right;**

**int pivot = A[(Left + Right) / 2]; */\* partition \*/***

**while (i <= j)**

**{**

**while (A[i] < pivot)**

**i++;**

**while (A[j] > pivot)**

**j--;**

**if (i <= j)**

**{**

**Swap(A[i],A[j]);**

**i++;**

**j--;**

**}**

**}**



*Đặng Trung Kiên* *Trang 115*

*Ngôn ngữ lập trình C++*



***/\* recursion \*/***

**if (Left < j)**

**QuickSort(A, Left, j);**

**if (i < Right)**

**QuickSort(A, i, Right);**

**}**

***// ham main***

**int** **main() {**

**int A[max], n;**

**cout<<"Nhap so phan tu:";**

**cin>>n;**

**NhapMang(A,n);**

**cout<<"Mang vua nhap la:";**

**XuatMang(A,n);**

**cout<<"Sap xep theo Quick Sort:"; QuickSort(A,0,n-1); XuatMang(A,n);**

**return 0;**

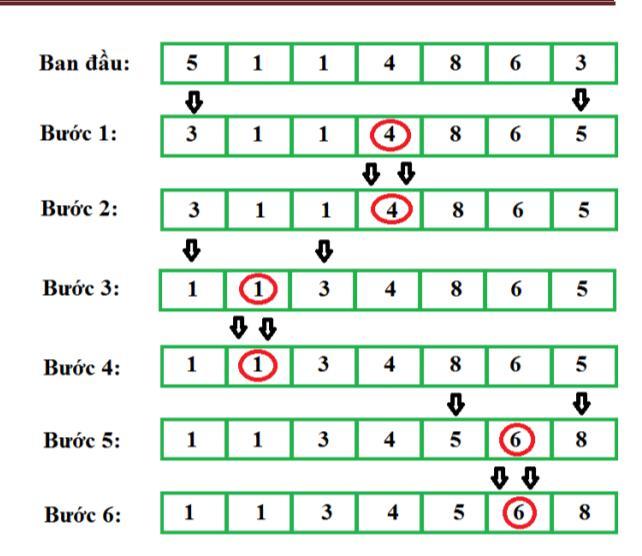
**}**

Cách sắp xếp như ***Hình 1***, trong đó phần khoanh tròn là **mốc**, mũi tên chỉ 2 phần tử vừa **hoán đổi** vị trí cho nhau:



*Đặng Trung Kiên* *Trang 116*

*Ngôn ngữ lập trình C++*



**Hình 1. Minh họa cách sắp xếp nhanh quicksort**



*Đặng Trung Kiên* *Trang 117*

*Ngôn ngữ lập trình C++*



**TÀI LIỆU THAM KHẢO**

1. Phạm Văn Ất - Giáo trình C++ và lập trình hướng đối tượng
2. Quách Tuấn Ngọc - Ngôn ngữ lập trình C++
3. <http://www.learncpp.com/>
4. <http://www.cplusplus.com/doc/tutorial/>
5. <http://www.tutorialspoint.com/cplusplus/>
6. Sách giáo khoa Tin học lớp 11
7. Sách bài tập Tin học lớp 11



*Đặng Trung Kiên* *Trang 118*

*Ngôn ngữ lập trình C++*



**PHỤ LỤC**

1. **Vận dụng chứng minh phản chứng của Toán học vào tƣ duy Tin học (Thuật toán đánh dấu phần tử)**

Bước 1. Giả sử mệnh đề đã cho là đúng

Bước 2. Nếu phát hiện ra một tính chất khiến cho mệnh đề đó sai, suy ra giả sử sai

Bước 3. Kết luận

**Ví dụ 1. Kiểm tra tính nguyên tố của một số nguyên dƣơng N**

**(** 1<*N*≤1018 **)**

Cách nghĩ của người mới học lập trình: Số nguyên tố mình học trong sách giáo khoa Toán là gì nhỉ? À, đó là số chia hết cho 1 và chính nó. Như vậy, với N>4, ta chỉ cần kiểm tra xem N có chia hết cho số nào từ 2 đến N-1 không. Nếu N chỉ cần chia hết cho bất kỳ số nào trong đoạn từ [2,N-1] thì ta kết luận nó không là số nguyên tố, nếu không thì ta kết luận nó là số nguyên tố. Tuyệt, vậy mình nghĩ ra cách làm rồi:

#include <iostream>

using namespace std;

int main()

{

long long n;

cin>>n;

if((n==2)||(n==3)) cout<<"nguyen to";

if(n>=4)

{

for(long long i=2;i<=n-1;i++) if(n%i==0) cout<<"khong nguyen to";

else cout<<"nguyen to";

}

return 0;



*Đặng Trung Kiên* *Trang 119*

*Ngôn ngữ lập trình C++*



}

Với tư duy như trên, nhìn qua chương trình có vẻ đúng với trường hợp N không nguyên tố. Nhưng hãy nhìn kỹ xem:

Nếu ta thử với N=6 thì chương trình sẽ đưa ra là:



Khong nguyen to

Khong nguyen to

Nguyen to

Nguyen to

Vậy chương trình trên sai ở đâu? Hãy thử chạy trong vòng lặp for N=6 > 4 rõ ràng rồi,

i=2, N chia hết cho 2, do vậy đưa ra thông báo N không là nguyên tố

i=3, N chia hết cho 3, do vậy đưa ra thông báo N không là nguyên tố

i=4, N không chia hết cho 4, do vậy đưa ra thông báo N là nguyên tố (và

đây chính là sai lầm của chương trình này, thử mỗi một trường hợp đã kết

luận luôn N là nguyên tố)

i=5, N không chia hết cho 5, do vậy đưa ra thông báo N là nguyên tố (và đây chính là sai lầm của chương trình này, thử mỗi trường hợp đã kết luận luôn N là nguyên tố)

Vậy bài này cần sửa như thế nào để tránh sai lầm. Ta sẽ dùng một biến OK, gọi là biến đánh dấu, OK lúc đầu bằng 1 (nghĩa là giả sử lúc đầu số N là nguyên tố). Nếu phát hiện ra một tính chất khiến số đó không là nguyên tố, thì OK sẽ bằng 0. (Đúng tư duy phản chứng của Toán nhé). Như vậy, sau khi chạy vòng lặp, nếu N chỉ cần chia hết cho bất kỳ số nào trong khoảng từ 2 đến N-1 thì OK sẽ chuyển thành 0. Cuối cùng, nếu OK vẫn giữ nguyên tính chất bằng 1 thì tôi kết luận N là nguyên tố, còn nếu OK bị thay đổi thành 0 thì kết luận N không là nguyên tố. Đây là chương trình minh hoạt thuật toán đánh dấu phần tử này:

#include <iostream>

using namespace std;



*Đặng Trung Kiên* *Trang 120*

*Ngôn ngữ lập trình C++*



int main()

{

long long n;

int ok=1;

cin>>n;

if(n>=4)

for(long long i=2;i<=n-1;i++)

if(n%i==0) ok=0; if(ok==1) cout<<"nguyen to"; else cout<<"khong nguyen to"; return 0;

}

Chương trình trên hoàn toàn đúng. Tuy nhiên, với N=6, chúng ta cùng nhau quan sát xem chuyện gì sẽ xảy ra: N=6 lớn hơn hoặc bằng 4

I=2, 6 chia hết cho 2 nên ok=0

I=3, 6 chia hết cho 3 nên ok=0

I=4, 6 không chia hết cho 4

I=5, 6 không chia hết cho 5

Cuối cùng do ok bằng 0 nên kết luận 6 không là nguyên tố.

* đây, tôi muốn dừng lại vòng lặp for ngay khi i=2, vì lúc đó ok=0 rồi, tôi muốn thoát khỏi vòng lặp ngay lúc đó, để chương trình đỡ mất thời gian chạy đến i=3, i=4,i=5. Khi đó tôi phải dùng đến câu lệnh break (thoát khỏi vòng lặp for hiện tại). Chương trình tôi viết lại khi đó như sau:

#include <iostream>

using namespace std;

int main()

{

long long n;



*Đặng Trung Kiên* *Trang 121*

*Ngôn ngữ lập trình C++*



int ok=1;

cin>>n;

if(n>=4)

for(long long i=2;i<=n-1;i++) if(n%i==0) {ok=0; break;}

if(ok==1) cout<<"nguyen to"; else cout<<"khong nguyen to"; return 0;

}

Khi đó, với N=6, chương trình trên sẽ chạy như sau:

N=6>=4, do đó vào vòng lặp

i=2, 6 chia hết cho 2, cho nên ok bằng 0, và thoát luôn khỏi vòng lặp do ok=0 nên kết luận n không là nguyên tố

Như vậy, ở đây ta thấy chương trình đã chạy nhanh hơn trước rồi

**Vậy chƣơng trình trên đã thật sự nhanh chƣa? Câu trả lời là chƣa.**

Vì lý do, máy tính có thể tính được 1 tỷ (109) phép tính trên một giây. Giả sử N là một số nguyên tố gần bằng 1018, như vậy, với việc chạy I từ 2 đến N-1, số phép tính kiểm tra xem N có chia hết cho I hay không sẽ lớn hơn 109 phép tính, do vậy sẽ vượt quá 1 giây. Liệu có cách nào mà máy tính chỉ chạy dưới 1 giây không? Đơn giản, ta sử dụng định lý của số học: ―mọi hợp số phải có ước nguyên tố nhỏ hơn hay bằng căn bậc hai của nó‖. Do vậy ta chỉ cần cho vòng lặp I chạy từ 1 tới căn bậc hai của N. Khi đó 1018 sẽ được căn bậc hai thành 109 phép tính, cách này sẽ nhanh hơn nhiều. Sau đây là minh họa cho chương trình:

#include <iostream>

#include<math.h>

using namespace std;

int main()

{



*Đặng Trung Kiên* *Trang 122*

*Ngôn ngữ lập trình C++*



long long n=1000000007;

int ok=1;

//cin>>n;

if(n>=4)

for(long long i=2;i<=sqrt(n);i++) if(n%i==0) {ok=0; break;}

if(ok==1) cout<<"nguyen to"; else cout<<"khong nguyen to"; return 0;

}

Vậy thuật toán trên đã thực sự là tối ưu chưa? Câu trả lời lần 2 vẫn là chưa. Vì sao? Các số nguyên tố chỉ có thể là các số lẻ trừ số 2, và do đó chúng không thể chia hết cho các số chẵn nên ta chỉ cần kiểm tra các ước số (giá trị của i) là số lẻ, do vậy thuật toán tiếp theo sẽ như sau:

#include <iostream>

#include<math.h>

using namespace std;

int main()

{

long long n=1000000007;

int ok=1;

//cin>>n;

if(n>2)

{

if(n%2==0){ok=0;}

else{for(long long i=3;i<=sqrt(n);i+=2)

if(n%i==0) {ok=0; break;}

}

}

if(ok==1) cout<<"nguyen to";



*Đặng Trung Kiên* *Trang 123*

*Ngôn ngữ lập trình C++*



else cout<<"khong nguyen to";

return 0;

}

Rõ ràng so với thuật toán trước đó, số giá trị của i cần kiểm tra giảm đi một nửa. Như vậy mấu chốt trong việc tăng tốc và cải tiến thuật toán nằm ở câu lệnh thay đổi giá trị của biến điều khiển i, ta chỉ cần giảm số giá trị của i cần kiểm tra là sẽ dẫn tới một thuật toán hiệu quả hơn. Vừa rồi ta đã loại bỏ được một nửa số giá trị của i bằng cách xét ước số 2 có thể có của n. Tiếp theo ta sẽ cải tiến thuật toán bằng cách xét ước số nguyên tố tiếp theo có thể có của n là số 3. Nếu n chia hết cho 2 hoặc 3 thì dễ dàng kết luận nó là hợp số (tất nhiên n phải khác hai giá trị đó). Ngược lại n sẽ có ước số có dạng 1+6i,5+6i , ta sẽ bắt đầu với giá trị i =

1. Nhưng nếu chỉ kiểm tra i sẽ thiếu mất ứng cử viên ở dạng 1+6i nên ta sẽ kiểm tra thêm giá trị i+2 (lưu ý: khi ta bắt đầu với i=5 thì ta có i+2 <=>

1+6i) cho đủ. Vậy thuật toán mới sẽ là như sau:

#include <iostream>

#include<math.h>

using namespace std;

int main()

{

long long n=1000000007;

int ok=1;

//cin>>n;

if(n>2)

{

if(n%2==0||n%3==0){ok=0;}

else{for(long long i=5;i<=sqrt(n);i+=6)

if(n%i==0||n%(i+2)==0) {ok=0; break;}

}

}

if(ok==1) cout<<"nguyen to";



*Đặng Trung Kiên* *Trang 124*

*Ngôn ngữ lập trình C++*



else cout<<"khong nguyen to";

return 0;

}

**2. Bài toán về kiểm tra tính nguyên tố của một số (cách tiếp cận khác)** Vẫn ví dụ 1. Kiểm tra tính nguyên tố của một số nguyên dương N

( 1 < *N* ≤1018 )

**Tƣ duy: Theo cách nghĩ từ thuật toán đếm số lƣợng phần tử thỏa**

**mãn một điều kiện cho trƣớc.**

**Cách tiếp cận thứ nhất:**

* Tính tổng S (hoặc đếm) các ước dương thực sự của N (ngoài 1 và chính N)
* Nếu S=0 thì kết luận N là số nguyên tố

#include <iostream>

using namespace std;

int main()

{ long long n,s; cin>>n; s=0;

for(long long i=2;i<=n/2;i++) if (n%i==0) s++;

if(s==0) cout<<"nguyen to"; else cout<<"khong nguyen

to";

return 0;

}

Nhận xét: Với cách tiếp cận này ta buộc phải duyệt hết tất cả các ước dương (có thể có) của N. Như vậy trong nhiều trường hợp sẽ bị duyệt những trường hợp không cần thiết, ví dụ n=50 thì đáng lẽ ta chỉ cần duyệt ước bằng 2 đã kết luận n không phải là số nguyên tố nhưng với cách này ta buộc phải duyệt đến hết ước bằng 25. Như vậy, ta đã lãng phí 24 lần duyệt không cần thiết.

**Cách tiếp cận thứ hai:**



* Duyệt các ước i (ước dương thực sự nếu có của n): từ 2 đến *N*



*Đặng Trung Kiên* *Trang 125*

*Ngôn ngữ lập trình C++*



* Điều kiện để còn tăng ước kiểm tra là i không là ước của N
* Nếu số lượng số không là ước của n vượt quá căn bậc hai của N trong vòng lặp điều kiện, điều đó chứng tỏ sau khi duyệt, không tìm thấy bất kỳ một ước nào, do đó ta kết luận N là số nguyên tố, nếu khác thì kết luận N là số nguyên

tố.

#include <iostream>

#include<math.h>

using namespace std;

int main()

{ long long n,s,i; cin>>n;

i=2;

while(i<=sqrt(n)&& n%i!=0) i++;

if(i>sqrt(n)) cout<<"nguyen to"; else cout<<"khong

nguyen to";

return 0;

}

Nhận xét: Cách tiếp cận này sẽ chưa hiệu quả với các số n lẻ. Vì các số lẻ nếu có ước thì ước của nó cũng là lẻ. Nhưng nếu tăng ước i lên 1 thì sẽ mất nhiều câu lệnh thừa. (giống như cách phần tích ở phần 1 của phụ lục)

Cách tiếp cận thứ 3 là cải tiến của cách tiếp cận thứ hai như sau:

* Kiểm tra n có là số chẵn không (trừ số 2): kết luận không là nguyên tố;
* Nếu n lẻ: Duyệt các ước lẻ (giống cách tiếp cận thứ hai)

#include <iostream>

#include<math.h>

using namespace std;

int main()

{ long long n,s,i; cin>>n;

if(n==2) cout<<"nguyen to"; else

{i=2;

while(i<=sqrt(n)&& n%i!=0) i=i+2;



*Đặng Trung Kiên* *Trang 126*

*Ngôn ngữ lập trình C++*



if(i>sqrt(n)) cout<<"nguyen to"; else cout<<"khong

nguyen to";

}

return 0;

}

Nhận xét: Cách tiếp cận này tuy cài đặt nhìn thì không gọn như cách tiếp cận thứ hai

nhưng về mặt hiệu quả thì hơn hẳn cách 2.

**3. Xử lý số nguyên lớn**

Phạm vi biểu diễn của kiểu số nguyên lớn nhất tầm từ là 1018 →1019 . Vậy đối với những số nguyên vượt quá khoảng đó thì có sao, hay ta còn gọi số nguyên lớn? Ta sẽ dùng cách nào. Khi đó ta có thể sử dụng mảng lưu các chữ số (hoặc một nhóm các chữ số) hoặc **dùng đến** **xâu** (tham khảo thêm tài liệu giáo khoa chuyên Tin tập 1-phần Biểu diễnsố nguyên lớn)

Giả sử muốn cộng hai phép tính A và B

A=10, B=20 thì C=A+B=30

Nhưng nếu A=10000000000000000000000000 (25 chữ số 0)

B= 10000000000000000000000000 (25 chữ số 0)

Khi đó kết quả C=A+B, với C được khai báo là long long sẽ xảy ra hiện tượng tràn số

Do vậy, ta phải khắc phục bằng cách khác. Ở đây tôi giới thiệu cách xử lý bằng xâu, một phần giúp các em ý nghĩa của việc dùng xâu trong các bài tập.

**Phép cộng hai số nguyên lớn**

Phép cộng hai số nguyên lớn được thực hiện từ phải qua trái và và phần nhớ được mang sang trái. Giờ ta phải thực biến các kí tự số thành các chữ số và thực hiện phép cộng từng chữ số và có nhớ

#include <iostream>

#include<stdio.h>

using namespace std;



*Đặng Trung Kiên* *Trang 127*

*Ngôn ngữ lập trình C++*



string cong(string a,string b)

{ long i,x,y,nho,tong; string c="";

nho=0; //bien nho ban dau bang 0

* + cong them ky tu 0 vao truoc xau a while (a.length()<b.length()) a="0"+a; //cong them ky tu 0 vao truoc xau b while(b.length()<a.length()) b="0"+b;
  + tien hanh cong hai so lon bang cach cong tu ky tu co nho

for (i=a.length()-1;i>=0;i--)

{

x=a[i]-'0'; y=b[i]-'0'; tong=x+y+nho; nho=tong/10; c=char(tong%10+'0')+c;

}

if (nho>0) c="1"+c; return c;

}

string hieu(string a,string b) { long i,x,y,nho,muon,hieu,s;

string c="";

muon=0; //bien muon ban dau bang 0

* cong them ky tu 0 vao truoc xau a while (a.length()<b.length()) a="0"+a; //cout<<a<<endl;

//cong them ky tu 0 vao truoc xau b



*Đặng Trung Kiên* *Trang 128*

*Ngôn ngữ lập trình C++*



while(b.length()<a.length()) b="0"+b; //cout<<b<<endl;

* tien hanh cong hai so lon bang cach cong tu ky tu co nho

for (i=a.length()-1;i>=0;i--)

{

x=a[i]-'0'; y=b[i]-'0'; s=x-y-muon; if (s<0)

{

s=s+10;

muon=1;

}

else muon=0; c=char(s+'0')+c;

}

while((c.length()>1)&&(c[0]=='0')) c.erase(0,1);

return c;

}

int main()

{ freopen("tru.inp","r",stdin); freopen("tru.out","w",stdout); string s1,s2;

cin>>s1>>s2;

cout<<hieu(s1,s2); return 0;

}



*Đặng Trung Kiên* *Trang 129*

*Ngôn ngữ lập trình C++*



Tương tự làm các bài tập như nhân hai số nguyên lớn, chia lấy dư của hai số nguyên lớn, chia lấy phần nguyên của hai số nguyên lớn (tham khảo tài liệu giáo khoa chuyên Tin) ( Cần đọc kỹ tài liệu giáo khoa chuyên Tin, code được, cần thiết cho việc xử lý số nguyên lớn trong nhiều bài tập)

Chú ý:

( *A* + *B*) mod *m* = ( *A* mod *m*) + ( *B* mod *m*)

( *A*.*B*) mod *m* = ( *A* mod *m*.)( *B* mod *m*)



*Đặng Trung Kiên* *Trang 130*