BAN HỌC TẬP KHOA CÔNG NGHỆ PHẦN MỀM
CHUỖI TRAINING GIỮA HỌC KÌ 1 NĂM HỌC 2021 – 2022

ĐOÀN KHOA
CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP

Sharing is learning

Ban học tập

Khoa Công Nghệ Phần Mềm
Trường ĐH Công Nghệ Thông Tin
ĐHQG Hồ Chí Minh

Email / Group

bht.cnpm.uit@gmail.com
fb.com/groups/bht.cnpm.uit

Training

Nhập môn lập trình

Thời gian training: 19h30 ngày 09/2/2022

Phòng: 2tg33du (Microsoft Team)

Trainer: Trần Chu Hùng Sơn - MMTT2020

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

Nội dung Training

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

I. **Thuật toán và lưu đồ**
II. Phép toán/Kiểu dữ liệu
III. Cấu trúc điều khiển
IV. Hàm/Đệ quy
V. Lỗi thường gặp
VI. Giải đề

Sharing is learning

Thuật toán và lưu đồ

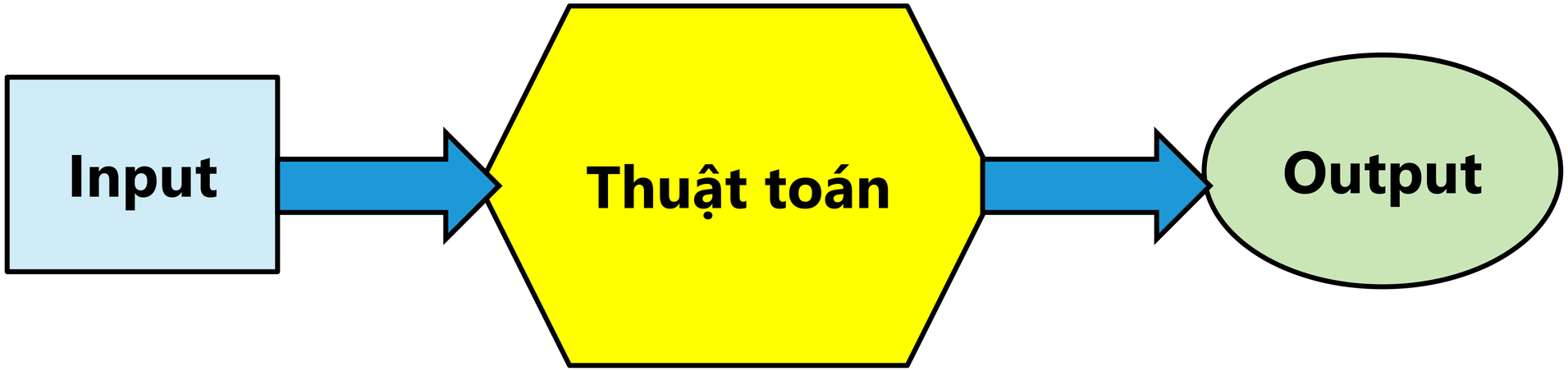
ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Thuật toán

1.1 Thuật toán là gì

Thuật toán là **tập hợp** (dãy) **hữu hạn** các **chỉ thị** (hành động) được **định nghĩa rõ ràng** nhằm **giải quyết một bài toán cụ thể** nào đó.

Thuật toán để giải một bài toán là một dãy hữu hạn các thao tác được sắp xếp theo một trình tự xác định sao cho sau khi thực hiện dãy thao tác đó, từ Input của bài toán, ta nhận được Output cần tìm.



ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Thuật toán

1.2 Các tiêu chuẩn của thuật toán

Tính chính xác/đúng:
- Quá trình tính toán hay các thao tác máy tính thực hiện là chính xác.
- Khi kết thúc, giải thuật phải cung cấp kết quả đúng đắn.

Tính phổ dụng/tổng quát:
- Có thể áp dụng cho một lớp các bài toán có đầu vào tương tự nhau.

Tính kết thúc/hữu hạn:
- Thuật toán phải dừng sau một số bước hữu hạn.

Thuật toán và lưu đồ

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Thuật toán

1.2 Các tiêu chuẩn của thuật toán

Tính rõ ràng/hiệu quả:

- Các câu lệnh minh bạch được sắp xếp theo thứ tự nhất định.

Tính khách quan/xác định:

- Được viết bởi nhiều người trên máy tính nhưng kết quả phải như nhau.
- Trong cùng một điều kiện hai bộ xử lý cùng thực hiện, thuật toán phải cho những kết quả giống nhau.

Thuật toán và lưu đồ

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Thuật toán

1.3 Phương pháp biểu diễn thuật toán

Sử dụng ngôn ngữ tự nhiên

Sử dụng ngôn ngữ thường ngày để liệt kê các bước của thuật toán.
Không có một quy tắc cố định nào trong việc thể hiện thuật toán.

Ví dụ: Giải phương trình ax+b=0

1. Nhập 2 số thực a và b.
2. Nếu a = 0 thì
    2.1. Nếu b = 0 thì
        2.1.1. Phương trình vô số nghiệm
        2.1.2. Kết thúc thuật toán.
    2.2. Ngược lại
        2.2.1. Phương trình vô nghiệm.
        2.2.2. Kết thúc thuật toán.
3. Ngược lại
    3.1. Phương trình có nghiệm.
    3.2. Giá trị của nghiệm đó là x = -b/a
    3.3. Kết thúc thuật toán.

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Thuật toán

1.3 Phương pháp biểu diễn thuật toán

Sử dụng mã giả Ngôn ngữ tựa ngôn ngữ lập trình chẳng hạn như Pascal, C...

Ví dụ: Giải phương trình ax+b=0

```
If a = 0 Then
Begin
      If b = 0 Then
            Xuất “Phương trình vô số nghiệm”
      Else
            Xuất “Phương trình vô nghiệm”
End
Else
      Xuất “Phương trình có nghiệm x = -b/a”
```

Thuật toán và lưu đồ

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Thuật toán

1.3 Phương pháp biểu diễn thuật toán

Sử dụng lưu đồ

Là một công cụ trực quan để diễn đạt các thuật toán.

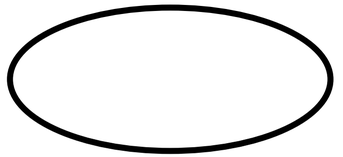
Biểu diễn thuật toán bằng lưu đồ sẽ giúp người đọc theo dõi được sự phân cấp các trường hợp và quá trình xử lý của thuật toán.

Thuật toán và lưu đồ

1. Thuật toán

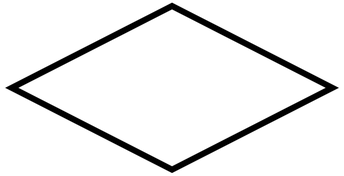
1.3 Phương pháp biểu diễn thuật toán

Sử dụng lưu đồ



Khối giới hạn
Chỉ thị bắt đầu và kết thúc.

Khối vào ra (input/output)
Nhập/Xuất dữ liệu.



Khối lựa chọn (decision)
Tùy điều kiện sẽ rẽ nhánh.

Khối thao tác (process)
Ghi thao tác cần thực hiện.

Đường đi (route)
Chỉ hướng thao tác tiếp theo.

Thuật toán và lưu đồ

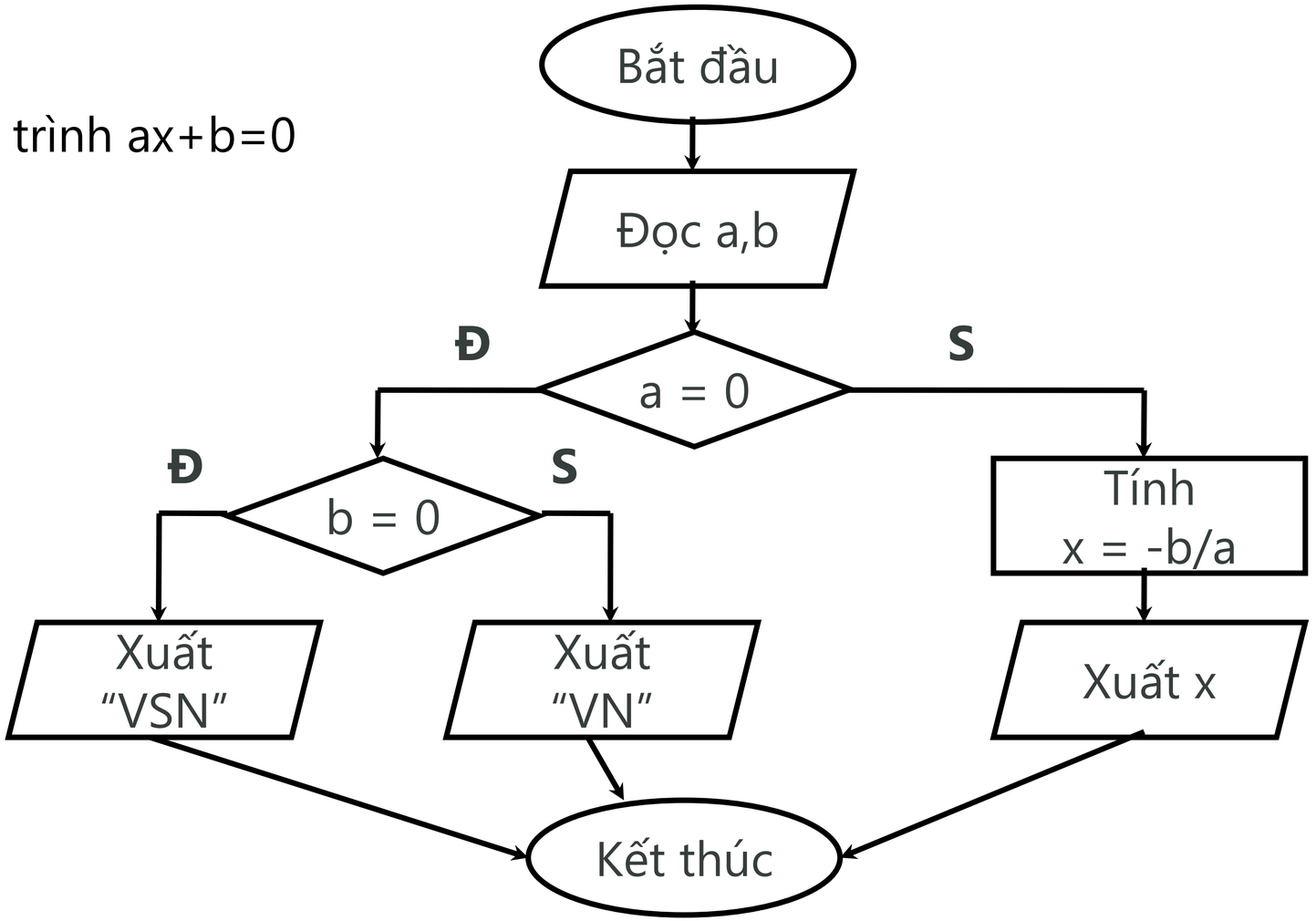
ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Thuật toán

1.3 Phương pháp biểu diễn thuật toán

Sử dụng lưu đồ

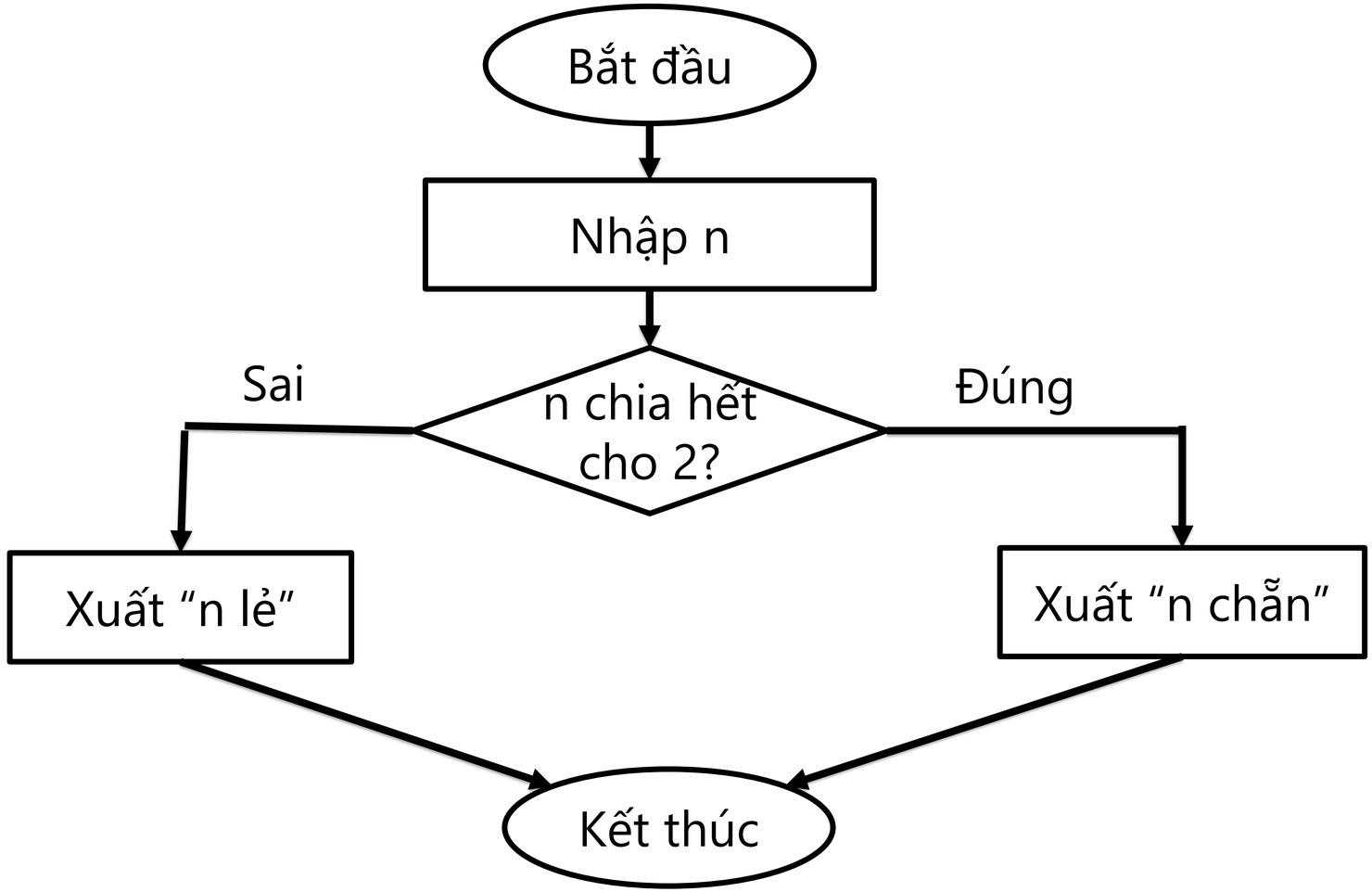
Ví dụ: Giải phương trình ax+b=0



Thuật toán và lưu đồ

2. Sử dụng lưu đồ biểu diễn thuật toán

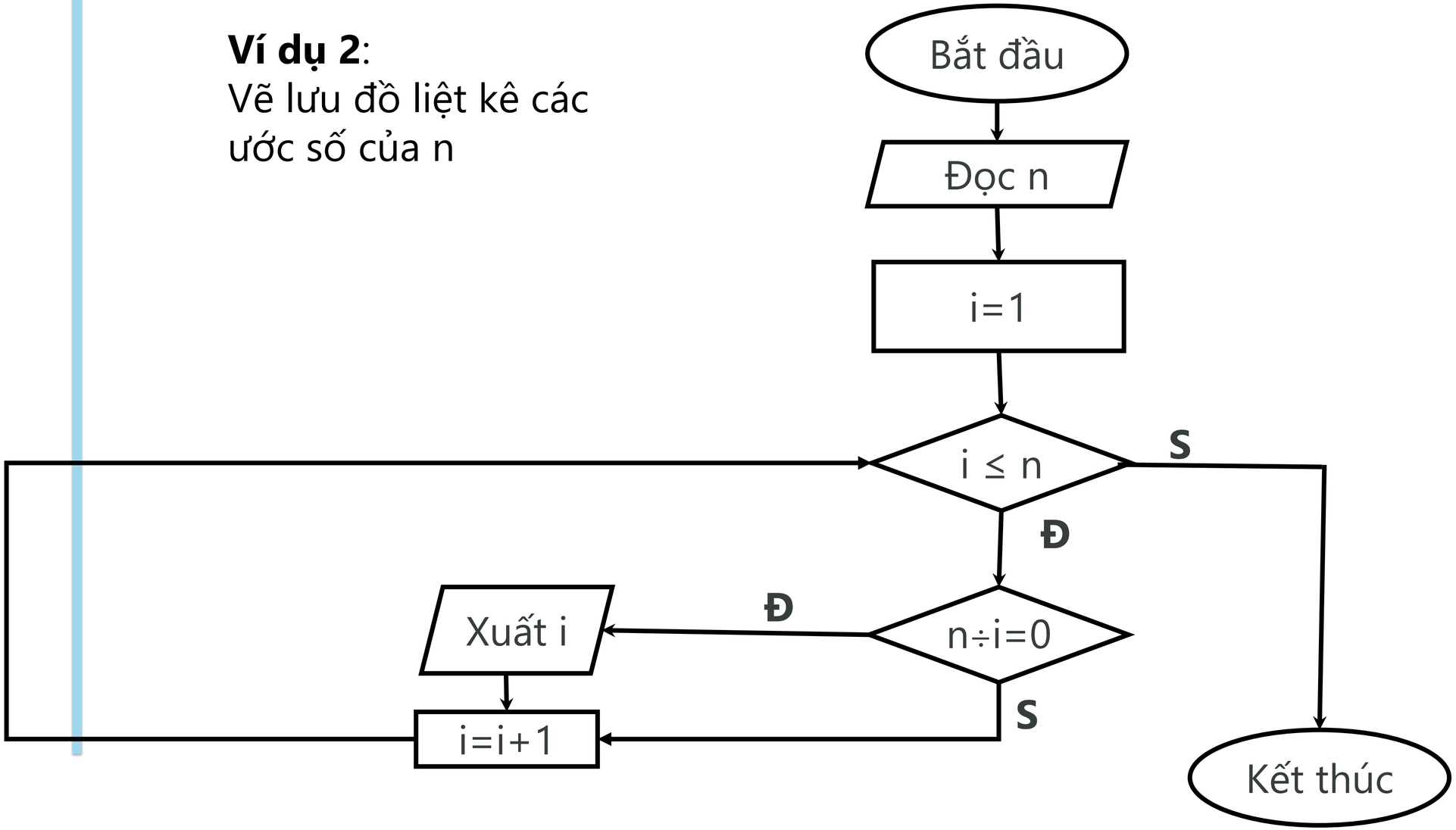
Ví dụ 1: Kiểm tra tính chẵn lẻ của một số nguyên



Thuật toán và lưu đồ

2. Sử dụng lưu đồ biểu diễn thuật toán

Ví dụ 2:
Vẽ lưu đồ liệt kê các ước số của n



2. Sử dụng lưu đồ biểu diễn thuật toán

Không ai là hoàn hảo!

Sau khi hoàn thì lưu đồ thì nhiều lúc có những sai sót.

Vì thế, cần phải kiểm tra tính đúng đắn của thuật toán bằng cách nhập thử dữ liệu và kiểm tra kết quả.

Nên dùng nhiều dữ liệu khác nhau để kiểm tra. Chú ý kiểm tra những trường hợp đặc biệt.

Ví dụ: đối với thuật toán giải phương trình bậc nhất, cần kiểm tra những trường hợp a bằng 0, a khác 0…..

Nội dung Training

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

I. Lưu đồ thuật toán

II. Phép toán/Kiểu dữ liệu

III. Cấu trúc điều khiển

IV. Hàm/Đệ quy

V. Lỗi thường gặp

VI. Giải đề

Sharing is learning

Kiểu dữ liệu và phép toán

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Kiểu dữ liệu cơ bản

1.1 Kiểu số nguyên

Kiểu dữ liệu	Kích thước	Phạm vi
short	2 bytes	[-32.768, 32.767]
unsigned short	2 bytes	[0, 65.535]
int	4 bytes	[-2.147.483.648, 2.147.483.647]
unsigned	4 bytes	[0, 4.294.967.295]
long	4 bytes	[-2.147.483.648, 2.147.483.647]
unsigned long	4 bytes	[0, 4.294.967.295]
long long	8 bytes	[-9.223.372.036.854.775.807, 9.223.372.036.854.775.807]
unsigned long long	8 bytes	[0, 18.446.744.073.709.551.615]

Kiểu dữ liệu và phép toán

1. Kiểu dữ liệu cơ bản

1.2 Kiểu số thực

Kiểu dữ liệu	Kích thước	Phạm vi
float	4 bytes	[3.4E-38, 3.4E+38](~7 chữ số)
double	8 bytes	[1.7E-308,1.7E+308](~15 chữ số)
long double	8 bytes	[1.7E-308,1.7E+308](~15 chữ số)

Kiểu số nguyên và kiểu số thực!

C++ phân biệt kiểu dữ liệu rất rõ ràng.

Ví dụ:

12 là kiểu số nguyên.

12.0 là kiểu số thực.

Mặc dù trong toán học thì hai số này là như nhau.

Kiểu dữ liệu và phép toán

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Kiểu dữ liệu cơ bản

1.3 Kiểu kí tự

Kiểu dữ liệu	Kích thước	Giá trị
char	1 byte	[-128, 127] hoặc [0, 255]
unsigned char	1 byte	[0, 255]

Lưu ý:
Các kí tự được viết trong cặp dấu nháy đơn. Ví dụ: **'a'**, **'b'**, **'*'**,...
Một số kí tự đặc biệt được viết sau dấu ****.

- **'\''** kí tự '
- **'\"'** kí tự "
- **'\n'** kí tự xuống dòng
- ....

Mỗi kí tự ứng với một giá trị số nguyên trong bảng mã **ASCII.**

Kiểu dữ liệu và phép toán

1. Kiểu dữ liệu cơ bản

1.3 Kiểu kí tự

Một số kí tự thường dùng trong mã ASCII

Ký tự	ASCII
0, 1, …, 9	48, 49, …, 57
A, B, …, Z	65, 66, …, 90
a, b, …, z	97, 98, …, 122
Enter, ESC, Space	13, 27, 32
“, +, -, *, /	34, 43, 45, 42, 47
<, =, >, @	60, 61, 62, 64

Kiểu dữ liệu và phép toán

1. Kiểu dữ liệu cơ bản

1.4 Kiểu logic

Kiểu dữ liệu	Kích thước	Giá trị
bool	1 bytes	**false**: giá trị 0 **true**: giá trị khác 0

Nhận xét:
Kiểu logic chỉ có 2 giá trị là **true** và **false**.
Giá trị **true** ứng với số nguyên **0**.
Giá trị **false** ứng với một số nguyên **khác 0**.

Kiểu dữ liệu và phép toán

2. Các phép toán

2.1 Toán tử số học

Toán tử	Miêu tả	Ví dụ
+	Cộng hai toán hạng	10 + 20 kết quả là 30
-	Trừ hai toán hạng	10 - 20 kết quả là -10
*	Nhân hai toán hạng	10 * 20 kết quả là 200
/	Phép chia	20 / 10 kết quả là 2
%	Phép chia lấy số dư	20 % 10 kết quả là 0

Lưu ý:
Đối với số nguyên, toán tử **/** sẽ thực hiện phép chia lấy nguyên. Ví dụ: **3/2** kết quả là **1**.
Đối với số thực, toán tử **/** sẽ thực hiện phép chia bình thường. Ví dụ: **3.0/2.0** kết quả là **1.5**.
Trong biểu thức nếu có một toán hạng là số thực thì kết quả là số thực.
Ví dụ: **3.0/2** kết quả là **1.5**.

Kiểu dữ liệu và phép toán

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Các phép toán

2.1 Toán tử số học

Code:

```
#include <iostream>
using namespace std;
int main () {
    cout << "3 + 2 = " << 3 + 2 << endl;
    cout << "3 - 2 = " << 3 - 2 << endl;
    cout << "3 * 2 = " << 3 * 2 << endl;
    cout << "3 / 2 = " << 3 / 2 << endl;
    cout << "3.0 / 2 = " << 3.0 / 2 << endl;

    return 0;
}
```

Output:

```
3 + 2 = 5
3 – 2 = 1
3 * 2 = 6
3 / 2 = 1
3.0 / 2 = 1.5
```

Kiểu dữ liệu và phép toán

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Các phép toán

2.2 Toán tử quan hệ

Toán tử	Miêu tả	Ví dụ
==	Kiểm tra nếu 2 toán hạng bằng nhau hay không. Nếu bằng thì kết quả là true.	10 == 20 là false
!=	Kiểm tra 2 toán hạng có giá trị khác nhau hay không. Nếu không bằng thì kết quả là true.	10 != 20 là true
>	Kiểm tra toán hạng bên trái có giá trị lớn hơn toán hạng bên phải hay không. Nếu lớn hơn thì kết quả là true.	10 > 20 là false
<	Kiểm tra toán hạng bên trái nhỏ hơn toán hạng bên phải hay không. Nếu nhỏ hơn thì kết quả là true.	10 < 20 là true
>=	Kiểm tra toán hạng bên trái lớn hơn hoặc bằng toán hạng bên phải hay không. Nếu đúng thì kết quả là true.	10 >= 20 là false
<=	Kiểm tra toán hạng bên trái nhỏ hơn hoặc bằng toán hạng bên phải hay không. Nếu đúng thì kết quả là true.	10 <= 20 là true

Kiểu dữ liệu và phép toán

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Các phép toán

2.2 Toán tử quan hệ

Code:

```
#include <iostream>
using namespace std;
int main () {
    cout << "10 == 20: " << (10 == 20) << endl;
    cout << "10 != 20: " << (10 != 20) << endl;
    cout << "10 > 20 : " << (10 > 20) << endl;
    cout << "10 < 20 : " << (10 < 20) << endl;
    cout << "10 >= 20: " << (10 >= 20) << endl;
    cout << "10 <= 20: " << (10 <= 20) << endl;
    return 0;
}
```

Output:

```
10 == 20: 0
10 != 20: 1
10 > 20 : 0
10 < 20 : 1
10 >= 20: 0
10 <= 20: 1
```

Lưu ý:
Kết quả được xuất ra là **0** và **1**.
0 ứng với giá trị **false**.
1 ứng với giá trị **true**.

Kiểu dữ liệu và phép toán

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Các phép toán

2.3 Toán tử logic

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

Kiểu dữ liệu và phép toán

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Các phép toán

2.3 Toán tử logic

Code:

```
#include <iostream>
using namespace std;
int main () {
    cout << "true && false: " << (true && false) << endl;
    cout << "true || false: " << (true || false) << endl;
    cout << "!true: " << !true << endl;
    cout << "!false: " << !false << endl;
    return 0;
}
```

Output:

```
true && false: 0
true || false: 1
!true: 0
!false: 1
```

Lưu ý:
Có thể dùng toán tử này đối với số nguyên.
Số **0** ứng với giá trị **false**.
Số **khác 0** ứng với giá trị **true**.

Ví dụ:
0 && 1 kết quả là **false**.
0 || 1 kết quả là **true**.
0 && true kết quả là **false**.

Kiểu dữ liệu và phép toán

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Các phép toán

2.4 Toán tử thao tác trên bit

p	q	p & q (AND)	p ^ q (XOR)	p | q (OR)	~p (NOT)
0	0	0	0	0	1
0	1	0	1	1	1
1	1	1	0	1	0
1	0	0	1	1	0

Toán tử dịch bit sang trái **<<**
3 = 0011
3 **<<** 1 = 0110 = 6
3 **<<** 2 = 1100 = 12

Toán tử dịch bit sang phải **>>**
12 = 1100
12 **>>** 1 = 0110 = 6
12 **>>** 2 = 0011 = 3

Nội dung Training

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

I. Lưu đồ thuật toán

II. Phép toán/Kiểu dữ liệu

III. Cấu trúc điều khiển

IV. Hàm/Đệ quy

V. Lỗi thường gặp

VI. Giải đề

Sharing is learning

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Dẫn nhập

Ví dụ thực tế:

Vào mỗi buổi sáng, mình thực hiện những công việc như sau:

- Thức dậy
- Vệ sinh cá nhân
- Ăn sáng
- Nếu có tiết sáng thì đi học, còn không thì ngủ tiếp

Ba việc đầu tiên được thực hiện lần lượt.

Việc cuối cùng được thực hiện dựa vào một **điều kiện**.

- Nếu điều kiện **đúng** thì thực hiện việc 1.
- Nếu điều kiện **sai** thì thực hiện việc 2.

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Dẫn nhập

Trong lập trình cũng tương tự như vậy.

Đôi khi chúng ta cần xét đến điều kiện nào đó để thực hiện những câu lệnh khác nhau.

Đó chính là cấu trúc rẽ nhánh.

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh if

2.1 Câu lệnh if đầy đủ (if else)

Cú pháp:

```
if (<điều kiện>)
    <câu lệnh 1>;
else
    <câu lệnh 2>;
```

Trong đó:

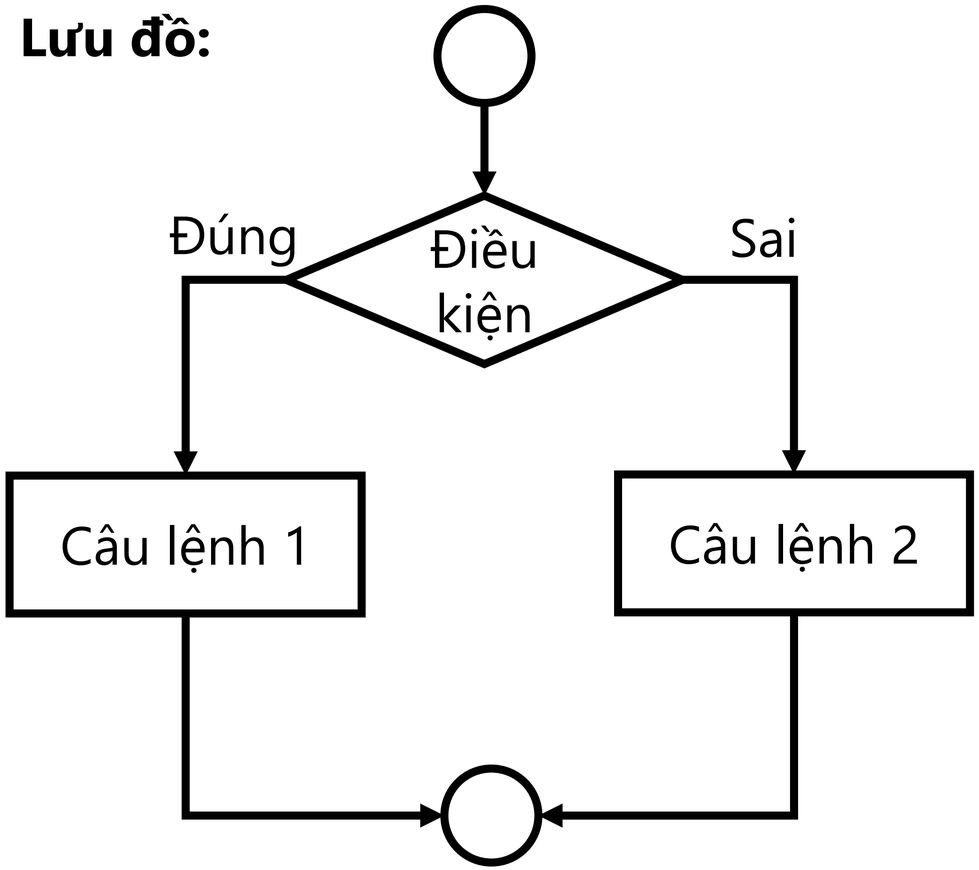
- <điều kiện> thường là biểu thức quan hệ, trả về kết quả **true** hoặc **false**
- <câu lệnh 1> là câu lệnh được thực thi nếu điều kiện **đúng (true)**.
- <câu lệnh 2> là câu lệnh được thực thi nếu điều kiện **sai (false)**.

Cấu trúc rẽ nhánh

2. Câu lệnh if

2.1 Câu lệnh if đầy đủ (if else)

Lưu đồ:




```
if (<điều kiện>)
    <câu lệnh 1>;
else
    <câu lệnh 2>;
```

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh if

2.1 Câu lệnh if đầy đủ (if else)

Ví dụ:
Viết một chương trình để **đánh giá 1 bài thi**, bài thi đó **đậu** khi **điểm từ 5 trở lên**, **ngược lại** thì **rớt.**
Yêu cầu: Cho người dùng nhập vào điểm bài thi. Thông báo kết quả bài thì đó đậu hay rớt.

Giải quyết vấn đề:
Nhập điểm bài thi.
Kiểm tra điểm đó có từ 5 trở lên hay không.
- Nếu đúng thì thông báo bài thi đậu.
- Ngược là thì thông báo bài thi rớt.

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh if

2.1 Câu lệnh if đầy đủ (if else)

Code:

```
#include <iostream>
using namespace std;
int main () {
    float diemBaiThi = 0;
    cout << "Nhap diem bai thi: ";
    cin >> diemBaiThi;
    if (diemBaiThi >= 5)
        cout << "Bai thi dau!" << endl;
    else
        cout << "Bai thi rot!" << endl;
    return 0;
}
```

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh if

2.2 Câu lệnh if thiếu (không có else)

Cú pháp:

```
if (<điều kiện>)
    <câu lệnh>;
```

Trong đó:

- <điều kiện> thường là biểu thức quan hệ, trả về kết quả **true** hoặc **false**
- <câu lệnh> là câu lệnh được thực thi nếu điều kiện **đúng (true)**.

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh if

2.2 Câu lệnh if thiếu (không có else)

Ví dụ:
Viết chương trình tính điểm trung bình của 2 môn học và xuất ra điểm trung bình, nếu điểm trung bình từ 9 trở lên thì xuất thêm thông báo đủ điều kiện nhận học bổng.

Giải quyết vấn đề:
Nhập điểm từng môn.
Tính điểm trung bình.
Xuất điểm trung bình.
Kiểm tra điểm trung bình có từ 9 trở lên hay không. Nếu đúng thì xuất thêm thông báo theo đề bài.

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh if

2.2 Câu lệnh if thiếu (không có else)

Code:

```
#include <iostream>
using namespace std;
int main () {
    float diemMon1 = 0, diemMon2 = 0;
    cout << "Nhap diem mon 1: ";
    cin >> diemMon1;
    cout << "Nhap diem mon 2: ";
    cin >> diemMon2;
    float diemTrungBinh = (diemMon1 + diemMon2)/2;
    cout << "Diem trung binh la: " << diemTrungBinh << endl;
    if (diemTrungBinh >= 9)
        cout << "Du dieu kien nhan hoc bong!" << endl;
    return 0;
}
```

Cấu trúc rẽ nhánh

2. Câu lệnh if

2.4 Toán tử điều kiện **? :**

Cú pháp:

```
<điều kiện>?<giá trị 1>:<giá trị 2>
```

Nếu <điều kiện> đúng thì trả về <giá trị 1>, ngược lại thì trả về <giá trị 2>

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh if

2.4 Toán tử điều kiện **? :**

Code:

```
#include <iostream>
using namespace std;
int main() {
    int a, b;
    cin >> a >> b;
    int max = (a>b)?a:b;
    cout << max << endl;
    return 0;
}
```

Có thể thấy rằng toán tử này có thể thay thế cho lệnh if. Tuy nhiên chỉ nên sử dụng trong những trường hợp đơn giản. Đối với trường hợp phức tạp hơn thì nên dùng if để code dễ nhìn.

```
#include <iostream>
using namespace std;
int main() {
    int a, b;
    cin >> a >> b;
    int max;
    if (a > b)
        max = a;
    else
        max = b;
    cout << max << endl;
    return 0;
}
```

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh switch case

Cú pháp:

```
switch (<x>) {
    case <giá trị 1>:
        <câu lệnh 1>;
        break;
    case <giá trị 2>:
        <câu lệnh 2>;
        break;
    ……
    case <giá trị n>:
        <câu lệnh n>;
        break;
    default:
        [câu lệnh mặc định];
}
```

Trong đó:

- <x> có thể là một biến hoặc một biểu thức.
- <giá trị 1>, <giá trị 2>…. <giá trị n> là các hằng số.
- <câu lệnh 1>, <cậu lệnh 2>…..<câu lệnh n> là các câu lệnh được thực hiện khi <x> bằng các giá trị tương ứng.
- [câu lệnh mặc định] là câu lệnh được thực hiện khi <x> không bằng giá trị nào trong các giá trị trên. Phần **default** có thể có hoặc không.

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh switch case

Nhận xét:
Câu lệnh **switch** so sánh một biến (hoặc biểu thức) với một danh sách các giá trị số nguyên hoặc các hằng kí tự. Mỗi giá trị trong danh sách chính là một **case** (trường hợp) của **switch**. Trong **switch** còn có thể có một **default** (mặc định). Trong mỗi **case** chứa các khối lệnh tương ứng.

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh switch case

Ví dụ:
Nhập vào một số nguyên. Hiển thị cách đọc số đó nếu số đó có giá trị từ 0 đến 9. Nếu không thì xuất "Khong doc duoc". Ví dụ: input: 1, output: mot

Nhận xét:
Có thể dùng các lệnh **if else** để chia thành 10 trường hợp. Hoặc có thể dùng lệnh **switch case** cho cả 10 trường hợp.

Cấu trúc rẽ nhánh

2. Câu lệnh switch case

Code dùng if else:

```
#include <iostream>
using namespace std;
int main () {
    int n;
    cin >> n;
    if (n == 0)
        cout << "khong" << endl;
    else if (n == 1)
        cout << "mot" << endl;
    else if (n == 2)
        cout << "hai" << endl;
    else if (n == 3)
        cout << "hai" << endl;
    else if (n == 4)
        cout << "bon" << endl;
    else if (n == 5)
        cout << "nam" << endl;
    else if (n == 6)
        cout << "sau" << endl;
    else if (n == 7)
        cout << "bay" << endl;
    else if (n == 8)
        cout << "tam" << endl;
    else if (n == 9)
        cout << "chin" << endl;
    else
        cout << "Khong doc duoc" << endl;
    return 0;
}
```

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh switch case

Code dùng switch case:

```
#include <iostream>
using namespace std;
int main () {
    int n; cin >> n;
    switch (n) {
    case 0:
        cout << "khong" << endl; break;
    case 1:
        cout << "mot" << endl; break;
    case 2:
        cout << "hai" << endl; break;
    case 3:
        cout << "ba" << endl; break;
    case 4:
        cout << "bon" << endl; break;
    case 5:
        cout << "nam" << endl; break;
    case 6:
        cout << "sau" << endl; break;
    case 7:
        cout << "bay" << endl; break;
    case 8:
        cout << "tam" << endl; break;
    case 9:
        cout << "chin" << endl; break;
    default:
        cout << "Khong doc duoc" << endl;
    }
    return 0;
}
```

2. Câu lệnh switch case

Nhận xét:
Việc dùng **switch** sẽ giúp code dễ đọc, dễ điều chỉnh hơn so với **if**.
.

Dùng break trong switch:
Lệnh **break** có thể có hoặc không. Nếu không được sử dụng thì không kết thúc cấu trúc **switch case** khi đã thực hiện hết khối lệnh của **case** đó. Thay vào đó, nó sẽ thực hiện tiếp các khối lệnh tiếp theo cho đến khi gặp lệnh **break** hoặc dấu **}** cuối cùng của cấu trúc **switch case**. Vì vậy, chúng ta có thể sử dụng 1 khối lệnh cho nhiều trường hợp khác nhau.

Cấu trúc rẽ nhánh

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Câu lệnh switch case

Ví dụ:
Viết chương trình nhập vào 1 số nguyên thể hiện 1 tháng trong năm, hiển thị số ngày trong tháng đó. Nếu là tháng 2 thì xuất “28 hoac 29”. Nếu số nguyên không phải từ 1 đến 12 thì xuất “Khong co thang nhu vay”.

Cấu trúc rẽ nhánh

2. Câu lệnh switch case

Code:

```
#include <iostream>
using namespace std;
int main () {
    int n;
    cin >> n;
    switch (n) {
    case 1:
    case 3:
    case 5:
    case 7:
    case 8:
    case 10:
    case 12:
        cout << "31" << endl;
        break;
    case 4:
    case 6:
    case 9:
    case 11:
        cout << "30";
        break;
    case 2:
        cout << "28 hoac 29";
        break;
    default:
        cout << "Khong co thang nhu vay" << endl;
        break;
    }
    return 0;
}
```

Cấu trúc lặp

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

1. Dẫn nhập

Trong cuộc sống có rất nhiều trường hợp cần phải lặp đi lặp lại 1 việc làm dựa trên một điều kiện nào đó.

Ví dụ:
Bạn A đi học mà quên học bài nên bị thầy bắt chép 10 lần bài phạt.
Bạn A bắt đầu chép lần thứ nhất, chép lần thứ hai….
Việc chép bài phạt đó lặp lại khi số lần chép nhỏ hơn hoặc bằng 10.

Cấu trúc lặp trong ví dụ trên gồm 2 phần cơ bản như sau:

- Việc làm: chép bài.
- Điều kiện: số lần chép nhỏ hơn hoặc bằng 10.

Trong lập trình cũng cần lặp lại những câu lệnh giống nhau hoặc tương tự nhau dựa vào điều kiện nào đó.

Cấu trúc lặp

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Vòng lặp while

Cú pháp:

```
while (<điều kiện>)
    <câu lệnh>;
```

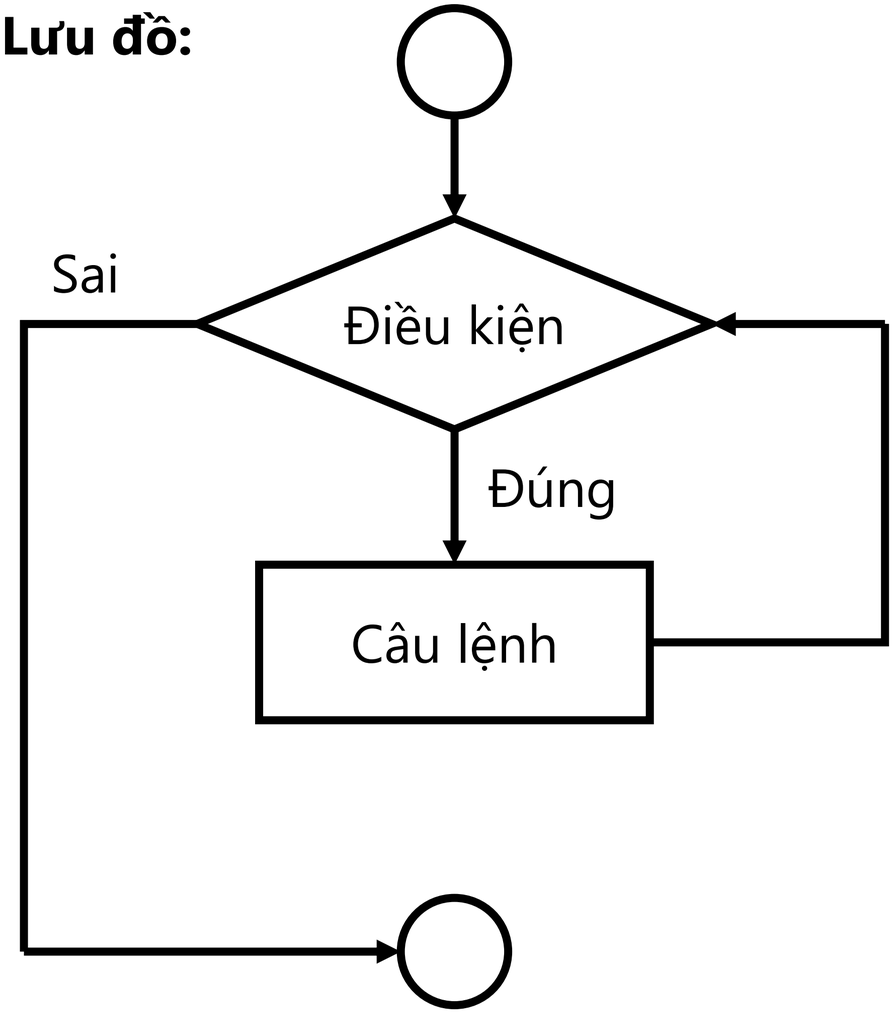
Trong đó:

- <điều kiện> thường là biểu thức quan hệ, trả về kết quả **true** hoặc **false**.
- <câu lệnh> là câu lệnh được lặp lại cho đến khi điều kiện **sai (false)**.

Cấu trúc lặp

2. Vòng lặp while

Lưu đồ:




```
while (<điều kiện>)
    <câu lệnh>;
```

Cách hoạt động của vòng lặp while:
Đầu tiên là kiểm tra **điều kiện**. Nếu điều kiện đúng thì thực hiện **câu lệnh**, nếu điều kiện sai thì thoát khỏi vòng lặp. Sau khi thực hiện **câu lệnh** thì quay lại kiểm tra **điều kiện**.

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Vòng lặp while

Ví dụ:
Xuất các số nguyên từ 1 đến 100 ra màn hình. Mỗi số cách nhau một khoảng trắng.

Ý tưởng :
Dùng vòng lặp while để lặp lại các lệnh xuất.
Tạo một biến để đếm từ 1 đến 100. Trong mỗi lần lặp thì xuất biến đó ra và tăng biến đó lên 1 đơn vị.
Điều kiện để vòng lặp thực hiện là biến đếm nhỏ hơn hoặc bằng 100.

Cấu trúc lặp

2. Vòng lặp while

Ví dụ:
Xuất các số nguyên từ 1 đến 100 ra màn hình. Mỗi số cách nhau một khoảng trắng.

Code:

```
#include <iostream>
using namespace std;
int main () {
    int dem = 1;
    while (dem <= 100) {
        cout << dem << " ";
        dem++;
    }
    return 0;
}
```

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

2. Vòng lặp while

Ví dụ:
Viết chương trình nhập vào số nguyên N. Tính tổng các chữ số trong N.

Ý tưởng :
Lấy N chia lấy dư cho 10 để lấy chữ số hàng đơn vị và cộng thêm vào tổng.
Lấy N chia cho 10 để cắt bỏ chữ số hàng đơn vị.
Thực hiện hai việc trên cho đến khi N bằng 0.

Cấu trúc lặp

2. Vòng lặp while

Ví dụ:
Viết chương trình nhập vào số nguyên N. Tính tổng các chữ số trong N.

Code:

```
#include <iostream>
using namespace std;
int main () {
    int N = 0;
    cin >> N;
    int sum = 0;
    while (N > 0) {
        sum += (N%10);
        N /= 10;
    }
    cout << sum << endl;
    return 0;
}
```

Cấu trúc lặp

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

3. Vòng lặp for

Cú pháp:

```
for ([khởi tạo]; [điều kiện]; [bước nhảy])
    <câu lệnh>;
```

Trong đó:

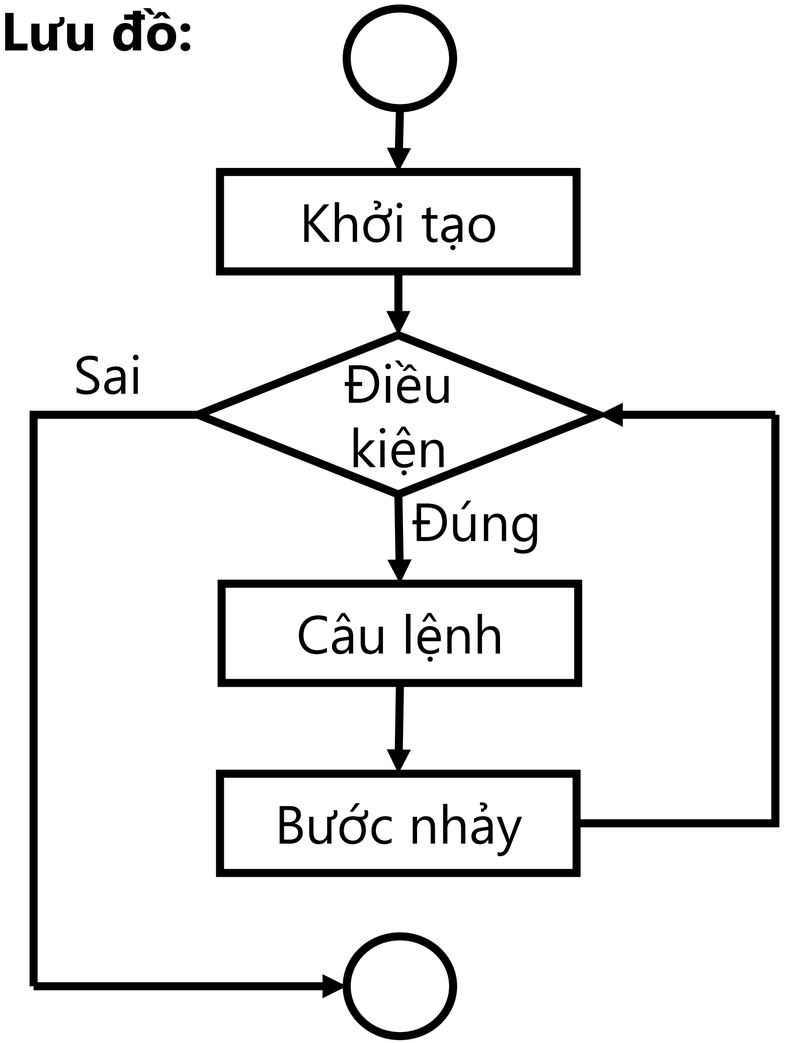
- [khởi tạo] là phần để khởi tạo biến đếm. Biến đếm có thể khai báo ngay trong phần khởi tạo. Phần này chỉ chạy một lần duy nhất khi bắt đầu vòng lặp.
- [điều kiện] là điều kiện của vòng lặp. Khi điều kiện sai thì kết thúc vòng lặp.
- [bước nhảy] phần này được thực thi ở cuối vòng lặp. Thường dùng để tăng hoặc giảm giá trị biến đếm.
- <câu lệnh> được thực hiện khi điều kiện đúng.

Các phần [khởi tạo], [điều kiện], [bước nhảy] có thể có nhiều biến, nhiều biểu thức. Những phần này có thể có hoặc không.

Cấu trúc lặp

3. Vòng lặp for

Lưu đồ:




```
for ([khởi tạo]; [điều kiện]; [bước nhảy])
    <câu lệnh>;
```

Nhận xét

Có thể xem vòng lặp **for** là một trường hợp đặc biệt của vòng lặp **while**. Giá trị khởi tạo, điều kiện và bước nhảy được kết hợp vào trong vòng lặp **for** giúp code ngắn gọn hơn so với vòng lặp **while**.

Cấu trúc lặp

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

3. Vòng lặp for

Ví dụ 1:
Viết chương trình tính tổng các số nguyên dương chẵn nhỏ hơn N.
Với N nhập từ bàn phím.

Code:

```
#include <iostream>
using namespace std;
int main () {
    int N; cin >> N;
    int sum = 0;
    for (int i = 2; i < N; i++)
        if (i % 2 == 0)
            sum += i;
    cout << sum << endl;
    return 0;
}
```

```
#include <iostream>
using namespace std;
int main () {
    int N; cin >> N;
    int sum = 0;
    for (int i = 2; i < N; i+=2)
        sum += i;
    cout << sum << endl;
    return 0;
}
```

Cấu trúc lặp

4. Vòng lặp do while

Cú pháp:

```
do {
    <câu lệnh>;
}
while (<điều kiện>);
```

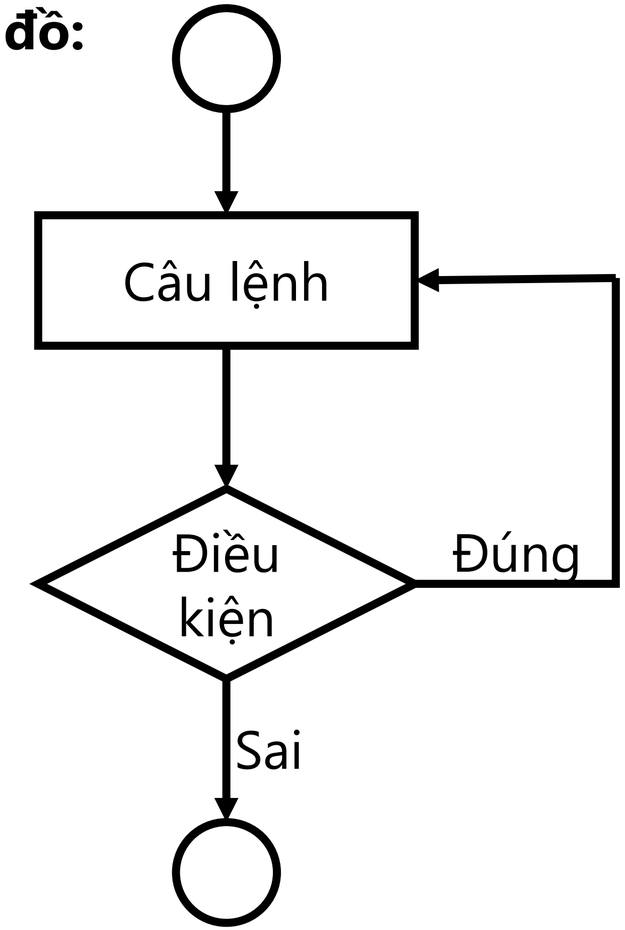
Trong đó:

- <câu lệnh> là lệnh được thực thi trong vòng lặp.
- <điều kiện> là điều kiện của vòng lặp.

Cấu trúc lặp

4. Vòng lặp do while

Lưu đồ:




```
do {
    <câu lệnh>;
}
while (<điều kiện>);
```

Nhận xét:

Vòng lặp **while** và **for** kiểm tra điều kiện trước rồi thực hiện câu lệnh.

Vòng lặp **do while** thực hiện câu lệnh rồi kiểm tra điều kiện. Vì thế, câu lệnh trong vòng lặp **do while** được thực hiện **ít nhất 1 lần**.

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

4. Vòng lặp do while

Ví dụ:
Viết chương trình nhập lần lượt 2 số nguyên a, b và tính a/b. Nếu nhập b bằng 0 thì cho người dùng nhập lại.

Ý tưởng:
Dùng vòng lặp do while để thực hiện việc nhập lại số nguyên b.
Điều kiện để nhập lại là b bằng 0.

Cấu trúc lặp

4. Vòng lặp do while

Code:

```
#include <iostream>
using namespace std;
int main () {
    int a = 0, b = 0;
    cout << "Nhap so nguyen a: "; cin >> a;
    do {
        cout << "Nhap so nguyen b (khac 0): ";
        cin >> b;
    }
    while (b == 0);
    cout << "a/b = " << float(a)/b << endl;
    return 0;
}
```

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

5. Lệnh break, continue

5.1 Lệnh break

Lệnh **break** dùng để thoát khỏi vòng lặp (cho dù điều kiện của vòng lặp vẫn đúng).

Ví dụ:

```
#include <iostream>
using namespace std;
int main () {
    for (int i = 0; i < 10; i++) {
        if (i == 5)
            break;
        cout << i << " ";
    }
    return 0;
}
```

Kết quả:

```
0 1 2 3 4
```

Cấu trúc lặp

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

5. Lệnh break, continue

5.2 Lệnh continue

Lệnh **continue** dùng để nhảy đến cuối khỏi vòng lặp và thực hiện lần lặp tiếp theo.

Ví dụ:

```
#include <iostream>
using namespace std;
int main () {
    for (int i = 0; i < 10; i++) {
        if (i == 5)
            continue;
        cout << i << " ";
    }
    return 0;
}
```

Kết quả:

```
0 1 2 3 4 6 7 8 9
```

Nội dung Training

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

I. Lưu đồ thuật toán
II. Phép toán/Kiểu dữ liệu
III. Cấu trúc điều khiển
IV. Hàm/Đệ quy
V. Lỗi thường gặp
VI. Giải đề

Sharing is learning

Hàm

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

- Khái niệm:
  - Một đoạn chương trình có tên, đầu vào và đầu ra
  - Có chức năng giải quyết một số vấn đề chuyên biệt cho chương trình chính
  - Được gọi với nhiều tham số khác nhau
  - Được sửa lỗi với nhu cầu tái sử dụng, sửa chữa, cải tiến

Hàm

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

Tên hàm

Tham số đầu vào

```
int tong(int a, int b)
{
    int s=a+b;
    return s;
}
```

Giá trị trả về

Hàm

Cú Pháp:

```
<kiểu trả về> <tên hàm>([<danh sách tham số>])
{
   <các câu lệnh>
   [return <giá trị>;]
}
```

Trong đó

- <kiểu trả về> : kiểu dữ liệu bất kỳ của C++ (char, int, long, float,...). Đặc biệt nếu hàm không trả về thì dùng kiểu void.
- <tên hàm>: theo quy tắc đặt tên định danh.
- <danh sách tham số> : tham số hình thức đầu vào giống khai báo biến, cách nhau bằng dấu ",".
- <giá trị> : trả về cho hàm qua lệnh return.

Nếu kiểu trả về là void thì không có lệnh này.

Hàm

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

Vị trí của hàm:

Trước hàm main

```
1   #include<bits/stdc
2   using namespace st
3
4
5   void Nhap(int a[], int &n, int k)
6   {
7       k=k+1;
8       cin>>n;
9       for (int i=0; i<n; ++i)
10      {
11          cin>>a[i];
12      }
13  }
14
15  int main()
16  {
17      int a[100],n,dem;
18      Nhap(a,n,dem);
19      return 0;
20  }
```

Sau hàm main

```
1   #includ
2   using nam ace std;
3
4   void Nhap(int [], int&, int);
5
6   int main()
7   {
8       int a[100],n,dem;
9       Nhap(a,n,dem);
10      return 0;
11  }
12
13  void Nhap(int a[], int &n, int k)
14  {
15      k=k+1;
16      cin>>n;
17      for (int i=0; i<n; ++i)
18      {
19          cin>>a[i];
20      }
21  }
```

Hàm

Vd về hàm

```
#include <iostream>

using namespace std;

int tong(int a, int b)
{
    int s=a+b;
    return s;
}

int main()
{
    int a,b;
    cin >> a >> b;
    cout << tong(a,b);
    return 0;
}
```

Đệ quy

Khái Niệm Đệ Quy

❖Khái niệm

Đệ quy là hàm được định nghĩa bằng chính nó.

❖Ví dụ

Tổng S(n) được tính thông qua tổng S(n- 1).

❖2 điều kiện quan trọng

- Tồn tại bước đệ quy.
- Điều kiện dừng.

Đệ quy

Cấu Trúc Hàm Đệ Quy

```
<Kiểu> <TênHàm>(TS)
{
   if (<ĐK dừng>)
   {
      …
      return <Giá trị>;
   }
   Lời gọi Hàm
   …
}
```

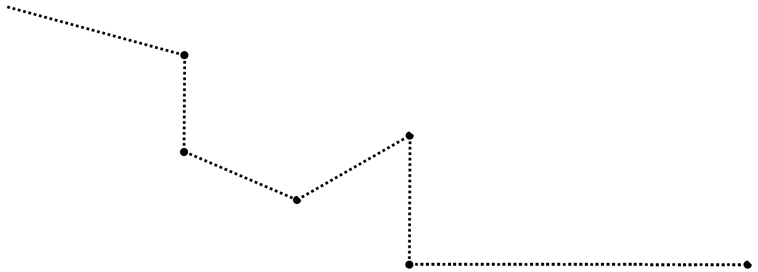
Phần dừng

- Phần khởi tính toán hoặc điểm kết thúc của thuật toán

Phần đệ quy

- Có sử dụng thuật toán đang được định nghĩa.

Đệ quy



Cấu trúc chương trình

```
<Kiểu> TênHàm(<TS>)
{
   if (<ĐK dừng>) {
      …
      return <Giá Trị>;
   }
   … TênHàm(<TS>); …
}
```

Ví dụ

Tính S(n) = 1 + 2 + … + n

Phân tích

S(n) = S(n – 1) + n

ĐK dừng: S(0) = 0

.: Chương trình :.

```
long Tong(int n)
{
   if (n ==0)
      return 0;
   return Tong(n–1) + n;
}
```

Đệ quy

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

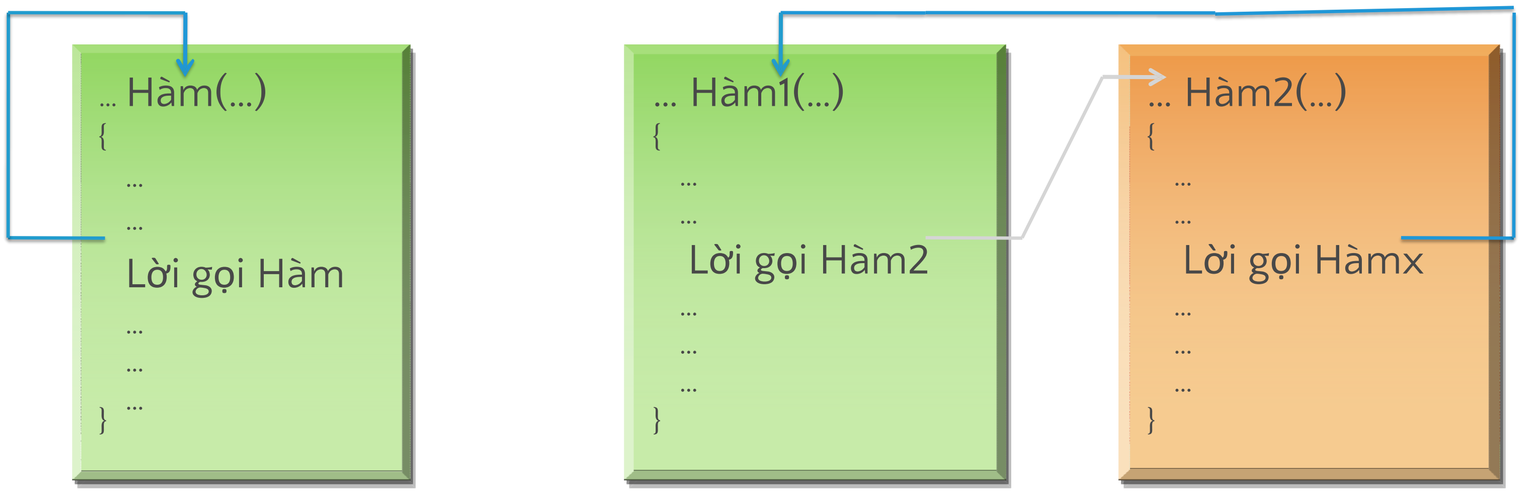
Ví Dụ

Tính số hạng thứ 4 của dãy Fibonacy

Ví dụ

```
Int F(int n)
If
{
        if (n==0) return 1;
        if (n==1) return 1;
Return F(n-1) + F(n-2);
}
```

Đệ quy



ĐQ trực tiếp

ĐQ gián tiếp

Đệ quy

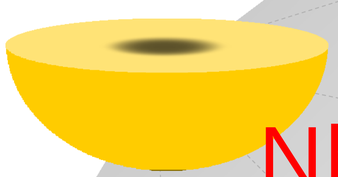
ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

Phân Loại

Tuyến tính

1

Trong thân hàm có duy nhất một lời gọi hàm gọi lại chính nó một cách tường minh.



Nhị phân

2

Trong thân hàm có hai lời gọi hàm gọi lại chính nó một cách tường minh.

Hỗ tương

3

Trong thân hàm này có lời gọi hàm tới hàm kia và bên trong thân hàm kia có lời gọi hàm tới hàm này.

Phi tuyến

4

Trong thân hàm có lời gọi hàm lại chính nó được đặt bên trong thân vòng lặp.

Đệ quy

Nhận Xét

Ưu điểm

Sáng sủa, dễ hiểu, nêu rõ bản chất vấn đề.

Tiết kiệm thời gian thực hiện mã nguồn.

Một số bài toán rất khó giải nếu không dùng đệ qui.

Khuyết điểm

Một số tính toán có thể bị lặp lại nhiều lần.

Một số bài toán không có lời giải đệ quy.

Nội dung Training

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

I. Lưu đồ thuật toán
II. Phép toán/Kiểu dữ liệu
III. Cấu trúc điều khiển
IV. Hàm/Đệ quy
V. Lỗi thường gặp
VI. Giải đề

Sharing is learning

Một số lỗi thường gặp

Lỗi cú pháp (lỗi chính tả - viết sai cú pháp)

- Thiếu dấu chấm phẩy hay ngoặc nhọn
- Sử dụng biến sai (chữ hoa chữ thường)
- Quên không khai báo kiểu dữ liệu
- Sử dụng tên biến trùng với từ khóa

```
#include <iostream>
using namespace std;
int main()
{
    cout << "Hello
world!"
    return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
    int bien = 5;
    cout << Bien;
    return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
    bien = 5;
    cout << bien;
    return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
    int try = 6;
    cout << try;
    return 0;
}
```

Một số lỗi thường gặp

Lỗi thực thi (lỗi liên quan đến tư duy giải thuật)

- Nhầm lẫn các dấu:
  - Dấu (=) và dấu (==)
  - Dấu (&), dấu (&&), dấu (||)
- Ép kiểu dữ liệu
- Vòng lặp vô hạn

```
#include <iostream>
using namespace std;
int main() {
    int a=2; b=3;
    if (a=b)
    {
        cout << a;
    }
}
```

```
#include <iostream>
using namespace std;
int main( )
{
        cout<<1/2;//0
        cout<<float(1/2); //0.5
}
```

```
#include <iostream>
using namespace std;
int main()
{
    int a=0; b=3;
    while (a<=b)
    {
        cout << a;
    }
}
```

Một số lỗi thường gặp

Lỗi logic (time limit)

- Miền chặn chưa chặt
- Vòng lặp quá dài
- Thuật toán chưa tối ưu

```
int gcd(int &x, int &y)
{
while (true){
      if ( x > y ) x = x - y;
      else if ( x < y ) y = y - x;
      else return x; }
}
```

```
int main()
{
int a, b,c=0,dem1=0,dem2=0;
cin>>a>>b;
for(int i=a;i<=b;i++)
{
int t = sqrt(i);
int e = sqrt(reverse(i));
 int f = reverse(i);
for(int j=2; j<=t; j++) if(i%j==0) {dem1++; break;}
for(int m=2; m<=e; m++) if(f%m==0) {dem2++; break;}
if(dem1==0&&dem2==0)c++; dem1=0; dem2=0;
continue; }
cout<<c;
return 0;
}
```

Nội dung Training

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

I. Lưu đồ thuật toán
II. Phép toán/Kiểu dữ liệu
III. Cấu trúc điều khiển
IV. Hàm/Đệ quy
V. Lỗi thường gặp
VI. Giải đề

Cấu trúc đề và đề thi minh họa

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

Cấu trúc đề

Câu 1: Có liên quan đến vòng lặp, cấu trúc điều khiển (khoảng 5đ)

+ Vẽ lưu đồ thuật toán

+ Viết chương trình C/C++

+ Tìm lỗi sai, tìm kết quả của chương trình.

Câu 2: Viết chương trình (xây dựng hàm, vòng lặp, cấu trúc điều khiển, ...) (khoảng 3đ)

Câu 3: Tương tự câu 2 nhưng mức độ sẽ khó hơn, thường sẽ là dạng vận dung cao sử dụng vòng lặp for, cấu trúc switch case, ... (khoảng 2đ)

+ Ở câu này thường đề thi sẽ yêu cầu chúng ta đặt kiểu dữ liệu phù hợp

+ Thuật toán sẽ phải tối ưu

Cấu trúc đề và đề thi minh họa

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

Đề thi minh họa

Câu 1: Nhập vào số nguyên dương n, hãy tính toán và in ra màn hình kết quả

S = 1 + 1*2 + 1*2*3 + … + 1*2*3*…*n.

a. Hãy vẽ lưu đồ thuật toán

b. Viết chương trình sử dụng ngôn ngữ C/C++

Câu 2: Nhập vào số nguyên dương n, hãy viết chương trình kiểm tra xem n có là số nguyên tố hay không? (nếu n là số nguyên thì thông báo ra màn hình TRUE và ngược lại thông báo ra màn hình FALSE)

Câu 3: Nhập vào hai số nguyên dương m, n hãy in ra tất cả các ước của n nhưng không là ước của m. Ví dụ m = 6, n = 9 thì kết quả in ra là 9 (mỗi giá trị cách nhau 1 khoảng trắng).

Cấu trúc đề và đề thi minh họa

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

Câu 1/ Tính tổng S = 1 + 1*2 + 1*2*3 + ... + 1*2*3*...*n

- Xác định input, output
  - Input: Số nguyên dương n.
  - Ouput: Giá trị của biểu thức.
- Nhận xét bài toán: Tn=T(n-1)*n
- Ý tưởng: Gọi tổng cần tìm là S ta có với số hạng thứ nhất là 1, số hạng thứ 2 là 1*2, số hạng thứ 3 là 1*2*3, ta sẽ lợi dụng vòng lặp để tính tổng S với điều kiện đã cho.

Bước 1: Đặt T=1; i=1;S=0;

Bước 2: Kiểm tra xem nếu i>n thì kết thúc việc tính toán còn không T=T*i và S=S+T sau đó tăng i lên 1 đơn vị và tiếp tục lặp lại bước này cho đến khi kết thúc thuật toán

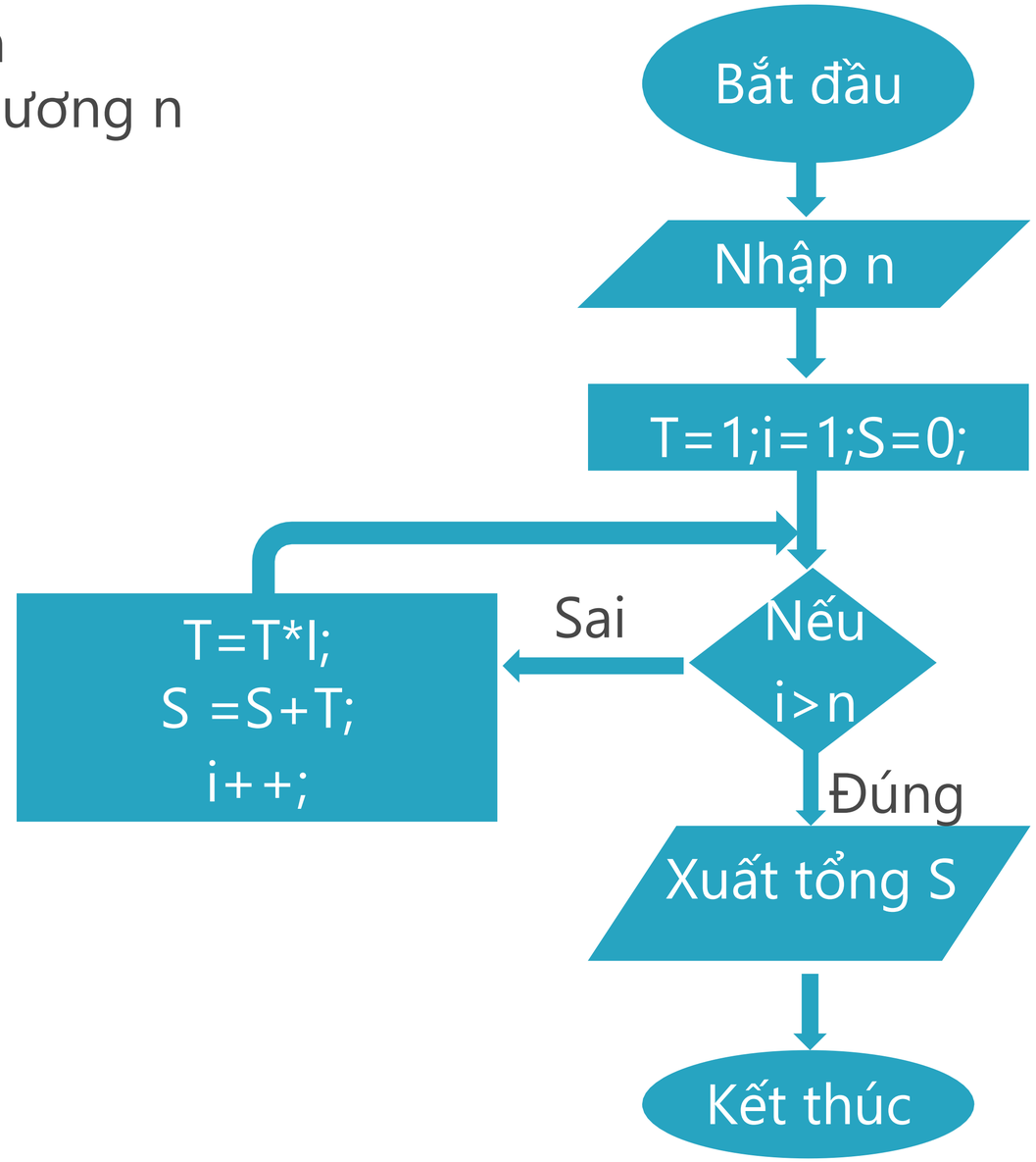
Cấu trúc đề và đề thi minh họa

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

a/ Vẽ lưu đồ thuật toán

Input: số nguyên dương n

ouput: tổng S



Cấu trúc đề và đề thi minh họa

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

b/ viết chương trình

```
#include <iostream>
using namespace std;
Int main ( )
{
        long long n,tong,S,i;
        cin>>n;
        i=1; S=1; tong=0;
        while (i<=n)
        {
                S*=i;
                tong+=S;
                 i++;
        }
        cout<<tong;
}
```

Cấu trúc đề và đề thi minh họa

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

Câu 2: Nhập vào số nguyên dương n, hãy viết chương trình kiểm tra xem n có là số nguyên tố hay không?

- Nhận xét: số nguyên tố là số chỉ có 2 ước là 1 và chính nó
- Ý tưởng: chúng ta sẽ kiểm tra tính chia hết của tất cả các số tự nhiên lớn hơn 1 và nhỏ hơn hoặc bằng căn bậc 2 của n. Chúng ta sẽ lợi dụng vòng lặp để làm tuần tự các thao tác kiểm tra tính chia hết của n cho 1 số tự nhiên mang tính chất như trên

Bước 1: Đặt i=2;

Bước 2: Kiểm tra nếu n chia hết cho i thì thông báo kết quả và kết thúc thuật toán còn không tăng i lên 1 đơn vị và tiếp tục thao tác này cho đến khi I lớn hơn căn của n.

Cấu trúc đề và đề thi minh họa

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

```
Int main ( )
{
        int n,i=2;
        cin>>n;
        If (n==0){cout << “FALSE”; return 0;}
        while (i*i<=n)
        {
              if (n%i==0)
                    {
                          cout<<“FALSE”;
                           return 0;
                     }
              i++;
         }
        cout<<“TRUE”;
}
```

Cấu trúc đề và đề thi minh họa

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

Câu 3: Nhập vào hai số nguyên dương m, n hãy in ra tất cả các ước của n nhưng không là ước của m. Ví dụ m = 6, n = 9 thì kết quả in ra là 9.

- Nhận xét: Ước của 1 số nguyên dương n là tập hợp tất cả các số tự nhiên sao n chia hết cho chúng.
- Ý tưởng: Cần tìm tất cả các số tự nhiên I bé hơn bằng n sao cho n chia hết i nhưng m không chia chia hết cho i. Ta dựa vào vòng lặp để tìm hết tất cả các số tự nhiên bé hơn bằng n thỏa mãn yêu cầu.
- Đặt i=2; ta kiểm tra nếu n chia hết cho i và m không chia hết cho i thì xuất giá trị i ra màn hình sau đó tăng i lên 1 đơn vị và tiếp tục lặp lại thao tác này cho đến khi i>n.

Cấu trúc đề và đề thi minh họa

BOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

```
#include    <iostream>
using  namespace  std;
Int main ( )
{
        long int n,i=2,m;
        cin>>n>>m;
        while (i<=n)
        {
                if (n%i==0 && m%i!=0)
                        {
                                cout<<i<<" ";
                        }
                i++;
        }
}
```

Cấu trúc đề và đề thi minh họa

ĐOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

Bài toán tổng quát nội dung:
Cho mảng a có n số nguyên (n<100).
Tính tổng $S = \sum_{i=1}^{n}(-1)^{n+1}|a_i| = |a_1| - |a_2| + |a_3| - \ldots (-1)^{n+1}|a_n|$

```
1  #include<iostream>
2  #include<cmath>
3  using namespace std;
4
5
6  void NhapMang(int [], int &);
7  int TinhTong(int [], int  );
8
9  int main()
10 {
11     int a[101],n;
12     NhapMang(a,n);
13     int S=TinhTong(a,n);
14     cout<<S;
15 }
16
```

```
17 void NhapMang(int a[], int& n)
18 {
19     cin>>n;
20     for (int i=1; i<=n; i++)
21         cin>>a[i];
22 }
23
24 int TinhTong(int a[], int n)
25 {
26     int S=0;
27     for (int i=1; i<=n; i++)
28         if (i%2==1)
29             S=S+abs(a[i]);
30         else
31             S=S-abs(a[i]);
32     return S;
33 }
```

Cấu trúc đề và đề thi minh họa

BOÀN KHOA CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP
Sharing is learning

BÀI THI GIỮA KỲ MÔN NHẬP MÔN LẬP TRÌNH – LỚP IT001.E11.ANTT

Thời gian: 90 phút

Qui định chung

- Không được sử dụng tài liệu, máy tính
- Thang điểm có tính điểm trình bày
- Sinh viên chỉ cần chọn làm 5 bài trong 6 bài

1. Viết chương trình tính sin(x) với độ chính xác 0.00001 theo công thức

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots + (-1)^n \frac{x^{2n+1}}{(2n+1)!}$$

2. Viết chương trình nhập vào hai giá trị chỉ thời gian (gồm giờ, phút, giây). Viết chương trình tính tổng của chúng

Ví dụ:

Nhập:

3 40 51
6 30 24

Xuất

10 11 15

3. Viết chương trình in ra bảng cửu chương từ 2 → 9 theo dạng

```
2*1= 2   3*1= 3   4*1= 4   5*1= 5   6*1= 6   7*1= 7   8*1= 8   9*1= 9
2*2= 4   3*2= 6   4*2= 8   5*2=10   6*2=12   7*2=14   8*2=16   9*2=18
2*3= 6   3*3= 9   4*3=12   5*3=15   6*3=18   7*3=21   8*3=24   9*3=27
2*4= 8   3*4=12   4*4=16   5*4=20   6*4=24   7*4=28   8*4=32   9*4=36
2*5=10   3*5=15   4*5=20   5*5=25   6*5=30   7*5=35   8*5=40   9*5=45
2*6=12   3*6=18   4*6=24   5*6=30   6*6=36   7*6=42   8*6=48   9*6=54
2*7=14   3*7=21   4*7=28   5*7=35   6*7=42   7*7=49   8*7=56   9*7=63
2*8=16   3*8=24   4*8=32   5*8=40   6*8=48   7*8=56   8*8=64   9*8=72
2*9=18   3*9=27   4*9=36   5*9=45   6*9=54   7*9=63   8*9=72   9*9=81
```

4. Viết chương trình nhập vào một số nguyên (int), sau đó in số đó ra theo hệ cơ số HEX (không dùng *printf* với *%x*)

5. Viết chương trình nhập vào số km đi được và tính tiền cước taxi biết:
   - 2 km đầu tiên là 12.000 VNĐ
   - mỗi 200m tiếp theo là 1.200 VNĐ
   - 30km tiếp theo mỗi km được tính là 4.000 VNĐ

6. Cần có tổng số 200.000 VNĐ từ các loại tiền giấy có mệnh giá 10.000 VNĐ, 20.000 VNĐ, 50.000 VNĐ. Viết chương trình liệt kê các phương án có thể.

BAN HỌC TẬP KHOA CÔNG NGHỆ PHẦN MỀM
CHUỖI TRAINING GIỮA HỌC KÌ 1 NĂM HỌC 2020 - 2021

ĐOÀN KHOA
CÔNG NGHỆ PHẦN MỀM
BAN HỌC TẬP

Sharing is learning

HẾT

CẢM ƠN CÁC BẠN ĐÃ THEO DÕI.
CHÚC CÁC BẠN CÓ KẾT QUẢ THI THẬT TỐT!

Ban học tập

Khoa Công Nghệ Phần Mềm
Trường ĐH Công Nghệ Thông Tin
ĐHQG Hồ Chí Minh

Email / Group

bht.cnpm.uit@gmail.com
fb.com/groups/bht.cnpm.uit

Link Góp ý và Điểm Danh:
https://tinyurl.com/TrainingBHTCNPM