

NHẬP MÔN LẬP TRÌNH

Tác giả

Nguyễn Văn Sơn

Nguyễn Bá Trung

Lê Thanh Tùng

Tháng 06/2015

MỤC LỤC

LỜI NÓI ĐẦU.....	6
BÀI 1: CÁC KHÁI NIỆM CƠ BẢN.....	7
I. Máy tính và cấu tạo máy tính.....	7
I.1. Máy tính là gì?	7
I.2. Cấu tạo máy tính:	8
II. Các khái niệm cơ bản	9
II.1. Lập trình?	9
II.2. Ngôn ngữ lập trình (programming language)?.....	9
II.3. Biên dịch (compile)?.....	10
II.3. Thông dịch (Interpreting)?	10
II.4. Trình biên dịch (compiler)	11
II.5. Ngôn ngữ máy (machine language)	11
III. Ngôn ngữ lập trình C	11
III.1. Tập ký tự (charater set) trong C	11
III.2. Từ khóa trong C (keyword).....	12
III.3. Tên (identifier):.....	12
III.4. Một số chương trình đơn giản trong C	13
IV. Bài tập tự làm	16
BÀI 2: GIẢI THUẬT - MÃ GIẢ - LƯU ĐỒ.....	22
I. Giải thuật.....	22
II. Mô tả giải thuật.....	22
II.1. Mã giả (Pseudocode):.....	22
II.2. Lưu đồ (Flowchart)	25
III. Bài tập tự làm	27
BÀI 3: BIỂU THỨC VÀ CÁC PHÉP TOÁN.....	33
I. Biểu thức, toán hạng, toán tử	33
I.1. Khái niệm:.....	33
I.2. Ví dụ minh họa:	33
II. Các phép toán:	34
II.1. Phép toán số học:.....	34
II.2. Phép toán quan hệ (so sánh):.....	36
II.3. Phép toán logic:.....	36

II.4. Phép toán tăng giảm:	37
II.5. Phép toán trên bit:	37
II.5. Phép toán điều kiện:	38
II.5. Ví dụ minh họa:	39
III. Bài tập tự làm	39
BÀI 4: KIỂU DỮ LIỆU – HẲNG – BIẾN – NHẬP XUẤT.....	46
I. Kiểu dữ liệu	46
I.1. Khái niệm:.....	46
II. Biến (Variable):.....	47
III. Hằng (Constant).....	48
III.1. Các loại hằng:	48
III.2. Khai báo hằng:.....	49
IV. Kiểu enum	50
V. Nhập xuất (Input - Output).....	51
V.1. Nhập xuất ký tự với conio.h.....	51
V.2. Nhập xuất ký tự với stdio.h.....	52
V.3. Nhập xuất có định dạng với stdio.h	52
V.4. Các ký tự định dạng.....	53
VI. Một số bài tập minh họa	53
VII. Bài tập tự làm	55
Câu hỏi trắc nghiệm	56
BÀI 5: CẤU TRÚC ĐIỀU KHIỂN	59
I. Cấu trúc if	59
I.1. Cú pháp:	59
I.2. Lưu đồ:	60
I.3.Ví dụ minh họa:	61
II. Cấu trúc if ... else...	65
II.1. Cú pháp:	65
II.2. Lưu đồ:	67
II.3.Ví dụ minh họa:	68
III. Cấu trúc if ... else...lồng nhau	69
III.1. Cú pháp:	69
III.2.Ví dụ minh họa:	70
IV. Cấu trúc switch ... case	72

IV.1. Cú pháp:	72
IV.2. Ví dụ minh họa:	73
BÀI 6: CẤU TRÚC VÒNG LẶP	78
I. Vòng lặp for	78
I.1. Cú pháp:	78
I.2. Lưu đồ:	79
I.3 Bài tập minh họa	79
II. Vòng lặp while	83
II.1. Cú pháp:	83
II.2 Lưu đồ	84
II.3 Bài tập minh họa	84
III. Vòng lặp do - while.....	89
III.1. Cú pháp:	89
III.2. Lưu đồ	90
III.3 Bài tập minh họa	90
IV. Lệnh break – continue.....	93
V. Bài tập tự làm	95
BÀI 7: HÀM VÀ CẤU TRÚC CHƯƠNG TRÌNH	100
I. Hàm	100
I.1. Khái niệm:.....	100
I.2. Xây dựng hàm:	100
I.3. Sử dụng hàm:	102
I.4. Truyền tham số:	103
I.5. Hàm trả về giá trị và hàm không trả về giá trị:.....	103
I.5. Khai báo hàm nguyên mẫu:.....	104
II. Hàm đệ qui	107
II.1. Khái niệm đệ qui và hàm đệ qui.....	107
II.2. Xây dựng hàm đệ qui	107
II.3. Một số ví dụ về hàm đệ qui.....	107
III. Cấu trúc chương trình	109
III.1. Cấu trúc chung một chương trình	109
III.2. Tầm vực biến.....	110
III.3. Khối lệnh.....	110
IV. Phân rã bài toán	111

V. Bài tập tự làm	114
BÀI 8: MẢNG - ARRAY	120
I. Mảng một chiều	120
I.1. Khái niệm.....	120
I.2. Khai báo.....	120
I.3. Truy xuất các phần tử của mảng	121
I.4. Các thao tác trên mảng 1 chiều.....	121
I.5. Bài tập minh họa	127
I.6. Bài tập tự làm	129
II. Mảng nhiều chiều.....	130
II.1. Khái niệm.....	130
II.2. Khai báo mảng 2 chiều:	130
II.3. Các thao tác trên mảng 2 chiều.....	131
II.4. Bài tập minh họa	132
II.5. Bài tập tự làm	133
III. Mảng là một tham số truyền vào hàm.....	134
III.1. Mảng một chiều là tham số truyền vào hàm	134
III.2. Mảng hai chiều là tham số truyền vào hàm	134
III.3. Bài tập minh họa	134
III.4. Bài tập tự làm	136
IV. Câu hỏi trắc nghiệm	137
BÀI 9: CHUỖI - STRING	141
I. Khái niệm về chuỗi	141
II. Khai báo chuỗi.....	141
III. Nhập xuất chuỗi	141
IV. Truy xuất từng ký tự của chuỗi	142
V. Một số hàm xử lý chuỗi trong C	143
VI. Một số ví dụ minh họa về xử lý chuỗi	146
VII. Bài tập.....	150
BÀI 10: CON TRỎ – POINTER.....	154
I. Khái niệm con trỏ	154
I.1. Địa chỉ (address)	154
I.2. Con trỏ (pointer)	154
II. Con trỏ và mảng	156

III.	Khởi tạo con trỏ.....	157
IV.	Các phép toán trên con trỏ	157
V.	Bộ nhớ động (Dynamic Memory).....	158
VI.	Bài tập minh họa	160
VII.	Bài tập tự làm	162
BÀI 11: STRUCT – KIỂU CẤU TRÚC.....		165
I.	Khái niệm kiểu cấu trúc.....	165
II.	Khai báo kiểu cấu trúc (struct)	165
III.	Truy xuất các thành phần của struct	166
IV.	Mảng struct	167
V.	Các ví dụ về struct	168
VI.	Bài tập.....	174
BÀI 12: LẬP TRÌNH VỚI TẬP TIN – FILE.....		179
I.	Khái niệm file.....	179
II.	Cấu trúc FILE.....	180
III.	Thao tác trên file text – Đọc File – Ghi File	180
IV.	Bài tập minh họa đọc-ghi file text	184
V.	Các thao tác trên file nhị phân	185
VI.	Bài tập minh họa đọc ghi file nhị phân.....	186
Cho thông tin nhân viên < mã số (int) , tên (50 ký tự) , lương (double)>		186
VII.	Bài tập tự làm	188
BÀI TẬP LÀM THÊM		195
TÀI LIỆU THAM KHẢO		202

LỜI NÓI ĐẦU

Các bạn sinh viên thân mến! Khi các bạn cầm trên tay và bắt đầu đọc những hàng chữ đầu tiên của cuốn sách này, chúng tôi hiểu các bạn là ai và đang muốn gì? Hầu hết sinh viên đều ít nhiều gặp khó khăn khi bắt đầu học lập trình, và kết quả trong những năm qua cho thấy rằng chỉ khoảng một phần ba sinh viên có kết quả từ khá trở lên khi học môn này. Với kinh nghiệm nhiều năm trong nghề dạy học và đặc biệt là dạy lập trình cho các bạn sinh viên mới chập chững làm quen với những dòng code lạ lẫm và khô cứng, chúng tôi đã biên soạn cuốn sách “Nhập môn lập trình” với mục đích giúp các bạn học môn này một cách hiệu quả nhất.

Cuốn sách này được biên soạn theo từng bài, mỗi bài học được trình bày một cách súc tích về một nội dung cụ thể bao gồm phần lý thuyết, ví dụ minh họa, bài tập tự làm và phần trắc nghiệm. Tất cả có 12 bài học trải dài kiến thức từ khái niệm cơ bản cho đến các kiến thức nền tảng trong lập trình. Mỗi bài học có tính độc lập về mặt nội dung nhưng lại có tính thừa kế từ bài trước đến bài sau giúp sinh viên dễ dàng thu nạp và tích lũy kiến thức sau mỗi bài học.

Các ví dụ minh họa trong cuốn sách này được viết bằng cả hai ngôn ngữ lập trình C và Java. Mỗi bài là mô tả cách áp dụng kiến thức lý thuyết trong mỗi tình huống bài toán cụ thể, vì vậy sinh viên không nên chỉ đọc qua mà hãy gõ lại những bài tập này và chạy thử để hiểu một cách chắc chắn trước khi làm các bài tập. Và cũng đừng quên hoàn tất các câu trả lời trắc nghiệm trước khi học bài mới.

Mặc dù đã có nhiều cố gắng khi biên soạn cuốn sách này, nhưng không thể nào tránh được những sai sót, chúng tôi mong nhận được sự đóng góp ý kiến của tất cả các bạn để cuốn sách ngày càng hoàn thiện hơn. Chúc tất cả các bạn học giỏi và thành công với lĩnh vực mà mình đã chọn.

Nhóm tác giả:

Nguyễn Văn Sơn

Nguyễn Bá Trung

Lê Thanh Tùng

BÀI 1: CÁC KHÁI NIỆM CƠ BẢN

I. Máy tính và cấu tạo máy tính

I.1. Máy tính là gì?

Máy tính (computer) hay còn gọi là máy vi tính là một dụng cụ điện tử, nó có thể:

- Chạy được các chương trình cài đặt sẵn
- Lưu trữ dữ liệu
- Xử lý dữ liệu thành thông tin hữu ích
- Có khả năng lập trình



Hình 1.1 Máy tính để bàn (desktop computer)



Hình 1.2 Máy tính xách tay (laptop)



Hình 1.3 Máy Ipad nhỏ gọn có tính năng gần như máy tính



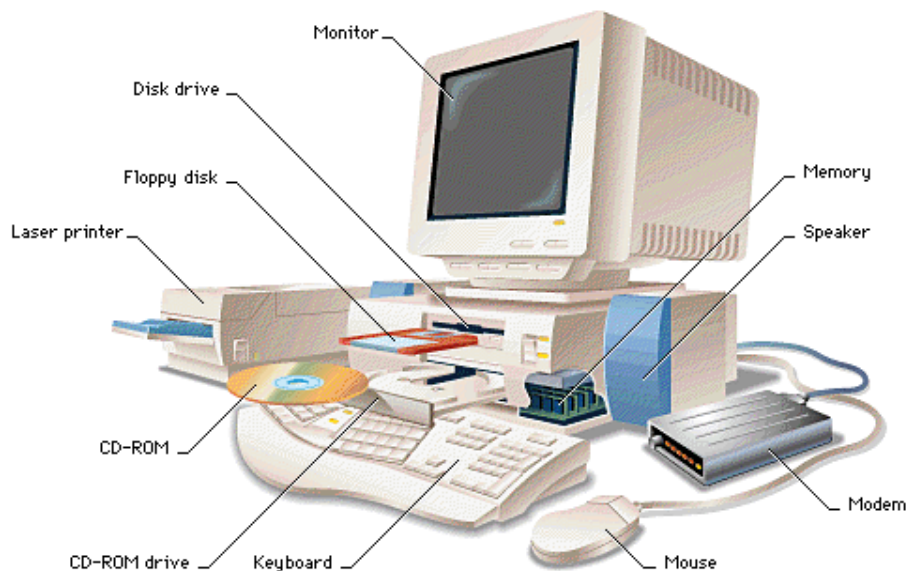
Hình 1.4: Điện thoại thông minh

Các loại điện thoại thông minh (smartphone) ngoài tính năng chính là gọi điện, nhắn tin nó còn có khả năng khác như lướt web, check mail, nghe nhạc, chơi game, chạy các phần mềm ứng dụng rất hữu ích,...

I.2. Cấu tạo máy tính:

Máy tính có 2 thành phần chính: phần cứng (hardware) và phần mềm (software).

- Phần cứng (hardware): Bao gồm các linh kiện cấu thành máy tính như: bộ vi xử lý (CPU), bo mạch chủ (mainboard), bộ nhớ tạm (ram), màn hình (monitor), đĩa cứng (harddisk), bàn phím (key board), con chuột (mouse), thùng máy (case),...



Hình 1.5: Cấu tạo máy tính

(<http://tip4pc.com/cau-cao-may-tinh/>)

- Phần mềm (software): bao gồm các hệ điều hành và các phần mềm được cài đặt vào máy tính.

→ Ngoài 2 phần cơ bản trên, thông thường máy tính còn kết nối với các thiết bị ngoại vi như: máy in (printer), modem, webcam, loa (speaker), máy quét (scanner),...

II. Các khái niệm cơ bản

II.1. Lập trình?

Lập trình (programming): là tạo ra một chương trình bằng một ngôn ngữ lập trình để máy tính thực hiện một công việc nào đó.

Lập trình là kỹ thuật:

- Sử dụng một ngôn ngữ lập trình
- Cài đặt các giải thuật (thuật toán)
- Tạo ra chương trình máy tính

Ví dụ: sử dụng ngôn ngữ lập trình C để tạo ra chương trình Hello:

```
#include "stdio.h"
void main()
{
    printf("Hello!");
}
```

II.2. Ngôn ngữ lập trình (programming language)?

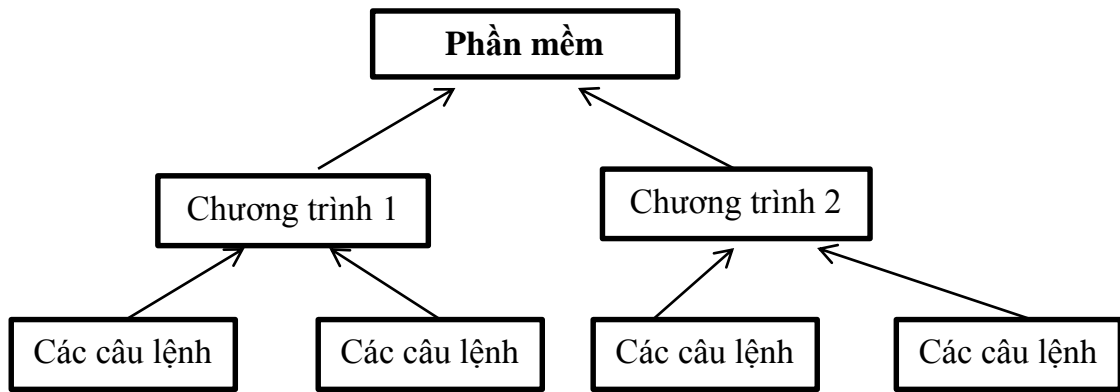
Ngôn ngữ lập trình là một hệ thống được ký hiệu hóa để miêu tả những tính toán (qua máy tính) trong một dạng mà cả con người và máy đều có thể đọc và hiểu được.

Một ngôn ngữ lập trình phải thỏa mãn được hai điều kiện cơ bản là:

- Nó phải dễ hiểu và dễ sử dụng đối với người lập trình, để con người có thể dùng nó giải quyết các bài toán.
- Nó phải miêu tả một cách đầy đủ và rõ ràng các tiến trình, để có thể chạy được trên các máy tính.

Câu lệnh (instruction) là đơn vị cơ bản của một ngôn ngữ lập trình.

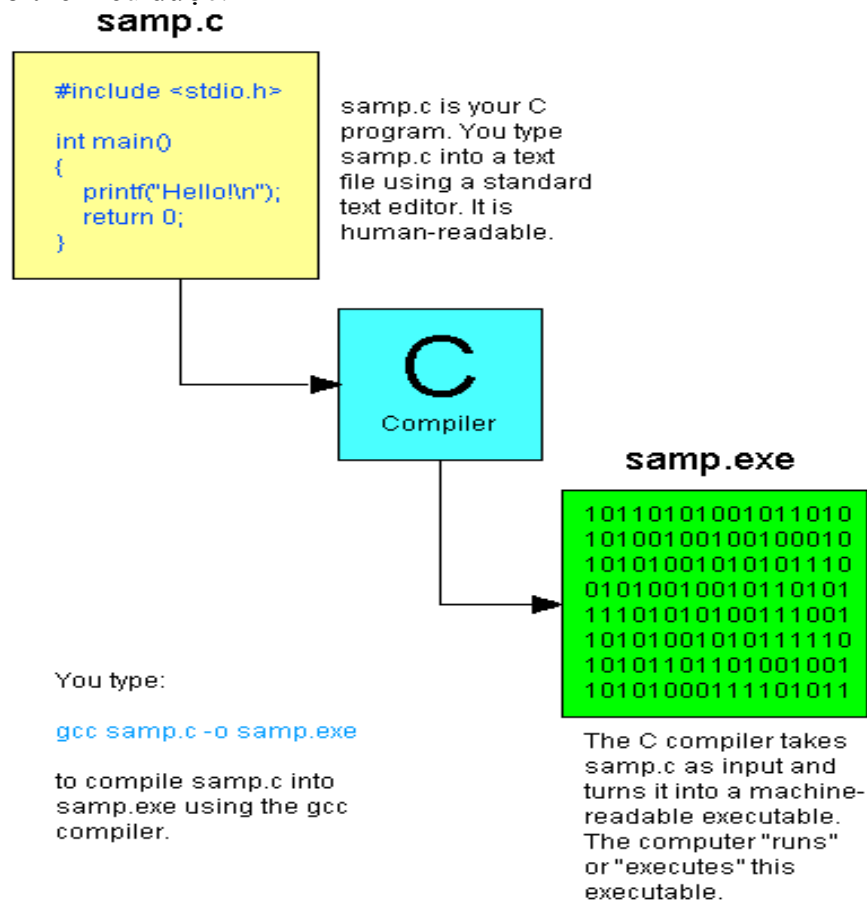
Mỗi chương trình (program) là tập hợp các câu lệnh giải quyết một bài toán cụ thể. Một nhóm lệnh → Một chương trình. Một nhóm các chương trình → Một phần mềm (software)



Hình 1.6: Sơ đồ một phần mềm

II.3. Biên dịch (compile)?

Là dịch một đoạn mã từ mã nguồn (Ngôn ngữ lập trình bậc cao) sang ngôn ngữ mà máy có thể hiểu được.



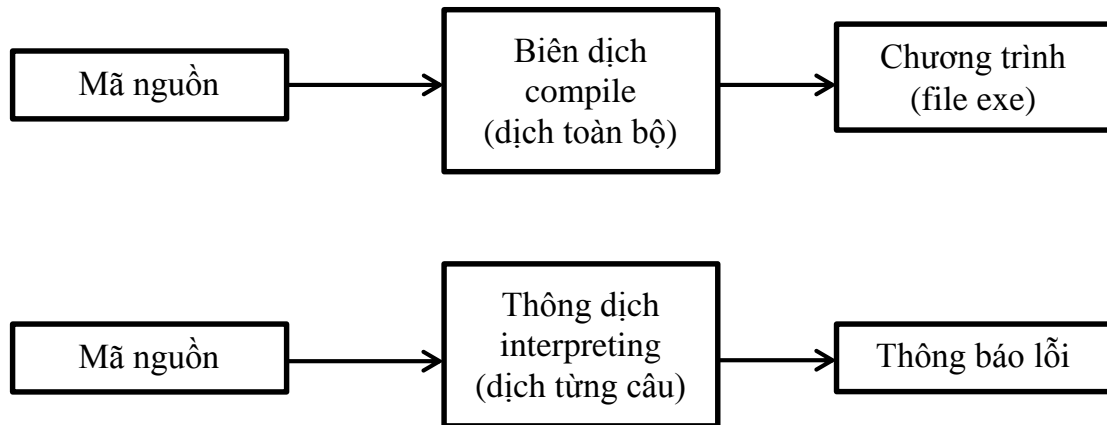
Trích từ: <http://computer.howstuffworks.com/c2.htm>

Hình 1.7: biên dịch (compile) chương trình

II.3. Thông dịch (Interpreting)?

Là dịch từng lệnh mã nguồn sang ngôn ngữ máy hiểu → Giúp người lập trình phát hiện các lỗi → Sửa các lỗi của chương trình.

Hình sau đây là sự khác nhau giữa biên dịch và thông dịch:



Hình 1.7: thông dịch (interpreting)

II.4. Trình biên dịch (compiler)

Là một chương trình dịch các đoạn mã từ mã nguồn (Ngôn ngữ lập trình bậc cao) sang ngôn ngữ mà máy hiểu.

Các ngôn ngữ bậc cao đều có trình biên dịch riêng, chẳng hạn như ngôn ngữ Pascal có trình biên dịch pascal, ngôn ngữ C có trình biên dịch C,...

II.5. Ngôn ngữ máy (machine language)

Là một ngôn ngữ lập trình mà trong đó mọi chỉ thị đều được biểu diễn dưới dạng số nhị phân (0 và 1). Đây là ngôn ngữ thế hệ đầu tiên. Bộ vi xử lý có thể hiểu trực tiếp ngôn ngữ này để xử lý.

```
1001001000101...
1100110011000...
0101001100110...
1010110100001...
.....
```

Hình 1.9: Ngôn ngữ máy

III. Ngôn ngữ lập trình C

Ngôn ngữ C do Dennis Ritchie đề xuất tại phòng thí nghiệm Bell vào những năm 70. Đến năm 1978 giáo trình “Ngôn ngữ lập trình C” cho chính tác giả viết được xuất bản và phổ biến rộng rãi.

Hiện nay ngôn ngữ C được hầu hết các trường đại học dùng để dạy cho các sinh viên chuyên ngành máy tính và các ngành kỹ thuật.

III.1. Tập ký tự (character set) trong C

Ngôn ngữ C được xây dựng trên bộ ký tự:

- 26 chữ cái hoa : A ...Z
- 26 chữ cái thường : a ... z
- 10 chữ số : 0...9
- Các ký hiệu toán học : +,-,*,/,=,()
- Ký tự gạch nối dưới (underscore) : _
- Các ký hiệu đặc biệt: .,:[]{}?!&%#\$...
- Ký tự khoảng trắng (space) dùng để cách các từ

III.2. Từ khóa trong C (keyword)

Từ khóa là những từ có một ý nghĩa xác định. Nó dùng để diễn đạt các phát biểu như khai báo các kiểu dữ liệu, viết các toán tử và các câu lệnh. Từ khóa không được dùng để đặt tên biến, tên hàm và tên hằng.

Nhóm từ khai báo kiểu dữ liệu:

- Kiểu số nguyên : **char , int , short , unsigned , long**
- Kiểu số thực: **float , double**
- Kiểu rời rạc : **enum**
- Kiểu cấu trúc : **struct , union**
- Kiểu rỗng: **void**
- Tự định kiểu: **typedef**
- Khai báo hằng: **const**
- Khai báo biến: **static , extern , auto, register**

Nhóm từ dành cho các phát biểu:

- Phát biểu chọn : **if, else, switch, case, default**
- Phát biểu lặp: **for, while, do**
- Từ khóa điều khiển: **break, continue, return, goto**

III.3. Tên (identifier):

- Tên là 1 từ: dùng để xác định các đối tượng khác nhau trong chương trình như: tên hằng, tên biến, tên mảng, tên hàm...
- Các từ trong C phân biệt chữ hoa chữ thường (case-sensitive)
- Bắt đầu của tên phải là ký tự chữ hoặc ký tự gạch nối dưới “_”, các ký tự sau là ký tự chữ, số, gạch nối dưới “_”

Ví dụ:

Sau đây là các tên đúng:

Baitap, Baitap1, _HoTen, Ho_Ten, Cong_, A, b

Sau đây là các tên sai:

Bai tap: sai vì có khoảng trắng

2_Cau: sai vì ký tự đầu tên là số

1Cau: sai vì ký tự đầu tên là số

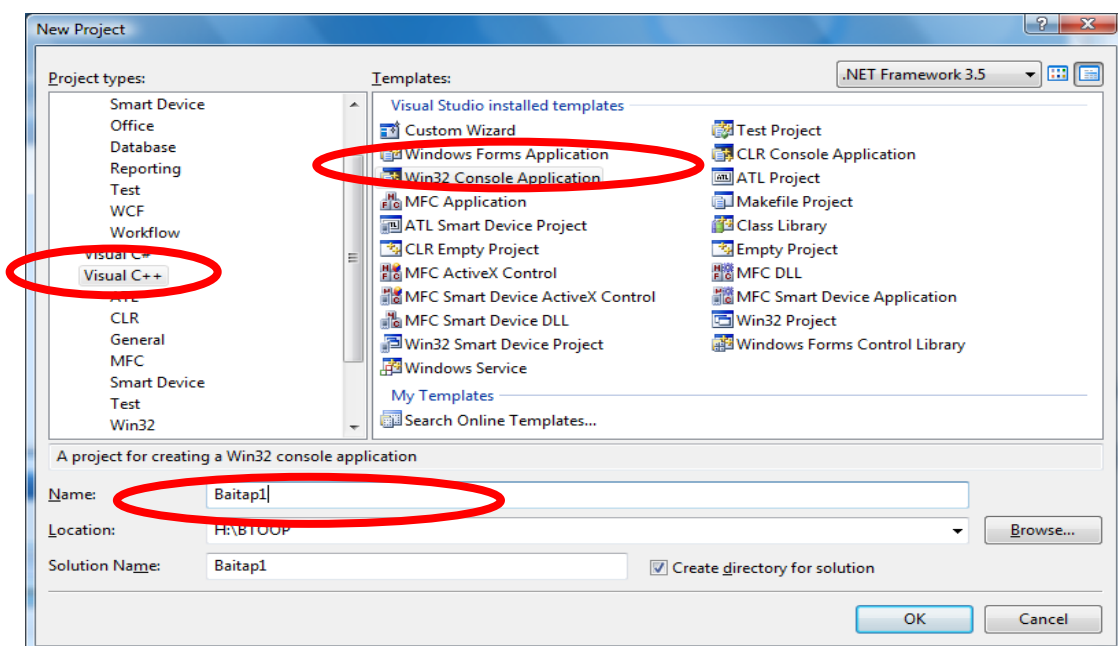
Ho-Ten: sai vì sử dụng ký tự gạch ngang –

Test#: sai vì sử dụng ký tự đặc biệt #

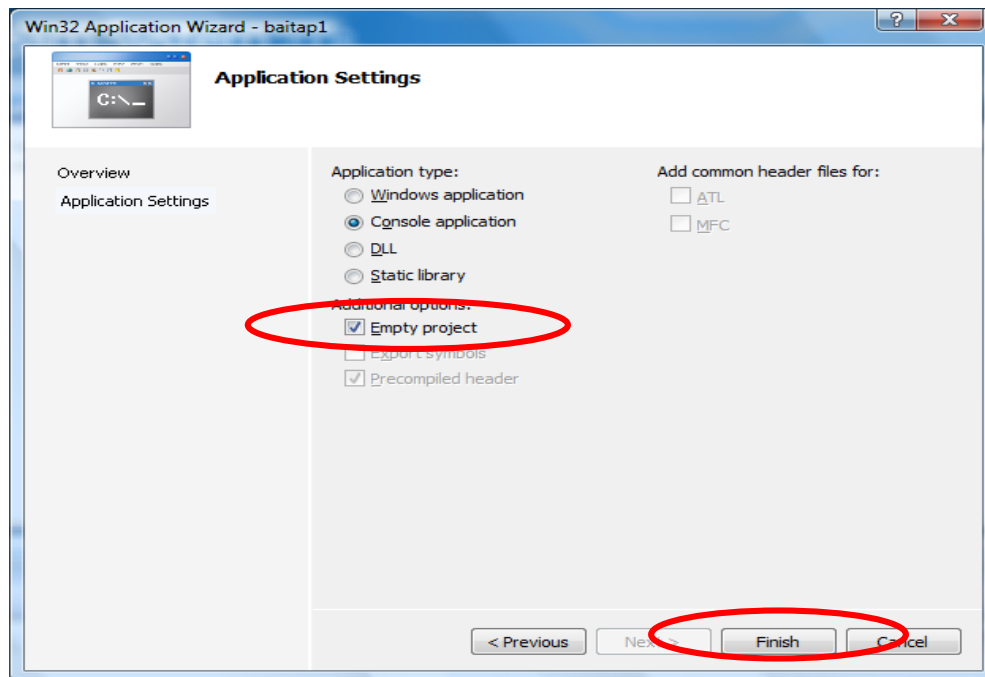
III.4. Một số chương trình đơn giản trong C

Ví dụ 1: Viết chương trình hiện dòng chữ “Hello students” lên màn hình. Sử dụng Visual Studio C++. Từng bước làm như sau:

- Khởi động Visua Studio
- Từ menu File → Chọn New → Project. Hoặc (Control + Shift + n)
- Chọn Visual C++ → chọn Win32 Console Application
- Đặt tên Baitap1 → Ok → Next



Hình 1.10a: Tạo chương trình C trong Visual Studio

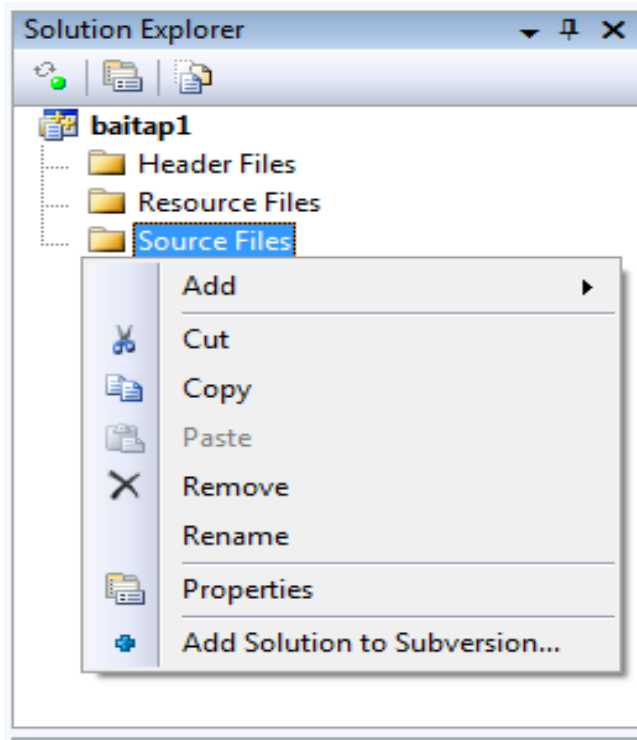


Hình 1.10b: Tạo chương trình C trong Visual Studio

- Click chọn Console Application; Chọn Empty project
- Chọn **Finish**

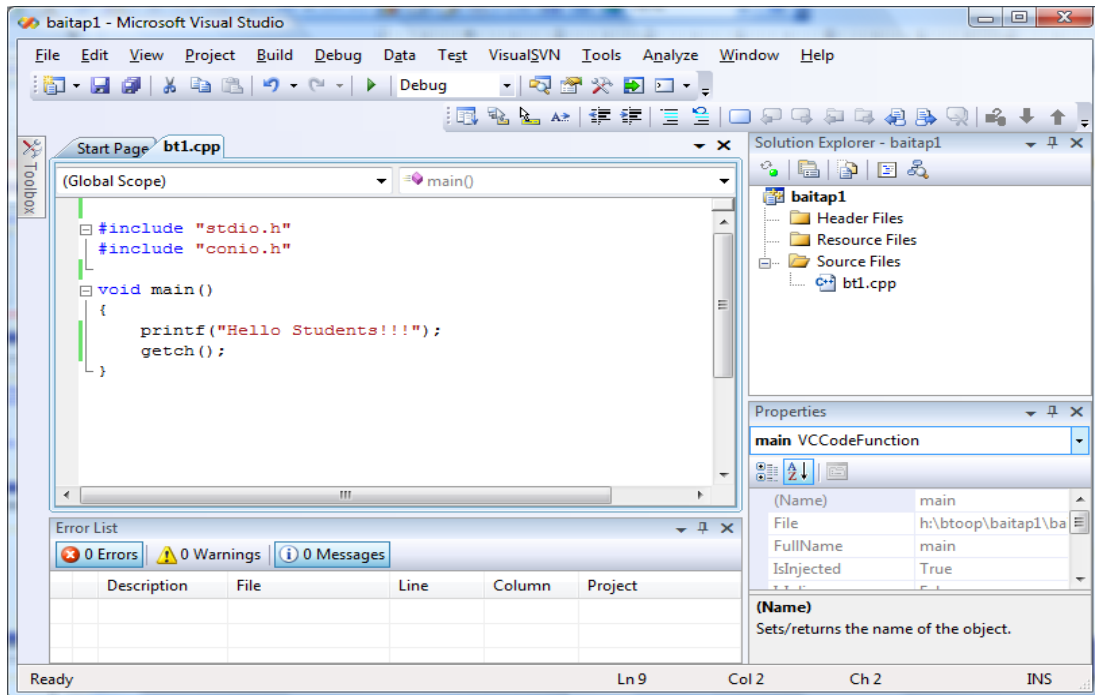
Từ cửa sổ Solution Explorer

- Right click Source file
- Add
- New Item
- C++ file (.cpp)
- Đặt tên file → **Add**



Hình 1.10c: Tạo chương trình C trong Visual Studio

- Gõ đoạn code như sau rồi nhấn F5 để chạy chương trình:



Hình 1.10d: Tạo chương trình C trong Visual Studio

Ví dụ 2: Viết chương trình in hình tam giác vuông hình dấu * ra màn hình:

```
#include <stdio.h>
#include <conio.h>
void main()
{
    printf("*\n");
    printf("**\n");
    printf("***\n");
    printf("****");
    getch(); //Chờ nhấn phím bất kỳ để kết thúc chương trình
}
```

Ví dụ 3: Viết chương trình nhập 2 số nguyên, tính tổng 2 số đó.

```
#include <stdio.h>
#include <conio.h>
void main()
{
```



```
int a, b, sum;
printf("Nhap gia tri a :");
scanf("%d",&a);
printf("Nhap gia tri b :");
scanf("%d",&b);
sum = a+b;
printf("Tong cua 2 so la : %d",sum);
getch(); //Chờ nhấn phím bất kỳ để kết thúc chương trình
}
```

Ví dụ 4: Viết chương trình nhập bán kính r, tính diện tích hình tròn.

```
#include <stdio.h>
#include <conio.h>
void main()
{
    float r, dientich;
    printf("Nhap ban kinh r :");
    scanf("%f",&r);
    dientich = 3.14 * r*r;
    printf("Dien tich hinh tron la : %6.2f",dientich);
    getch();
}
```

Ghi chú: %6.2f là hiển thị giá trị có độ rộng là 6 và 2 số lẻ

IV. Bài tập tự làm

Câu 1: Viết chương trình hiện ra màn hình hai câu thông báo sau:

Chao ban!

Chuc mung ban den voi mon hoc Nhap Mon Lap Trinh.

Câu 2: Viết chương trình in ra màn hình hình tam giác cân gồm các dấu sao (*).

Câu 3: Viết chương trình nhập vào hai số nguyên dương. Tính tổng, hiệu, tích của 2 số. Hiển thị kết quả ra màn hình.

Câu 4: Viết chương trình tính tổng bình phương của hai số được nhập vào từ bàn phím.

Câu 5: Viết chương trình nhập vào một số dương có 2 chữ số, in ra số đảo ngược của số đó

Câu 6: Viết chương trình nhập giờ, phút, giây. In ra tổng số giây.

Câu 7: Viết chương trình nhập bán kính r, tính diện tích của hình tròn.

Câu 8: Viết chương trình nhập cạnh a, tính diện tích của hình vuông

Câu 9: Viết chương trình nhập bán kính r, tính chu vi của hình tròn

Câu 10: Viết chương trình nhập đáy nhỏ, đáy lớn và chiều cao của hình thang cân, tính chu vi

Câu hỏi trắc nghiệm:

Câu 1: Một chương trình máy tính do?

a/ Máy tính tạo ra

b/ Con người tạo ra

c/ Mặc nhiên có sẵn trong mọi máy tính

d/ Tất cả đều đúng

Câu 2: Virus máy tính là một chương trình?

a/ Đúng

b/ Sai

Câu 3: Virus máy tính do?

a/ Con người tạo ra

b/ Máy tính tự sinh ra

c/ Phần mềm sinh ra

d/ Tất cả đều sai

Câu 4: Phần mềm được sử dụng trong các lĩnh vực?

a/ Giáo dục

b/ Y tế

c/ Kế toán

d/ Trong hầu hết các lĩnh vực

Câu 5: Các tập tin thư viện trong ngôn ngữ C có phần mở rộng là:

a/ doc

b/ cpp

c/ c

d/ h

Câu 6: File mã nguồn sau khi biên dịch thành công sẽ ra một file cùng tên và có phần mở rộng là:

a/ c

b/ h

c/ exe

d/ cpp

Câu 7: Kết thúc một câu lệnh trong ngôn ngữ C sẽ là:

a/ Dấu chấm phẩy

b/ Dấu chấm

c/ Dấu hai chấm

d/ Dấu phẩy

Câu 8: Trong ngôn ngữ C cách đặt tên nào sau đây bị sai?

a/ _BaiTap

b/ BaiTap_

c/ Bai_Tap

d/ Bai Tap

Câu 9: Trong ngôn ngữ C cách đặt tên nào sau đây bị sai?

a/ 10_cau

b/ Cau_10

c/ Cau10

d/ CAU10

Câu 10: Từ nào dưới đây không phải là từ khóa trong ngôn ngữ C?

a/ float

b/ for

c/ integer

d/ long

Câu 11: Từ nào dưới đây không phải là từ khóa trong ngôn ngữ C?

a/ char

b/ string

c/ while

d/ break

Câu 12: Từ nào dưới đây không phải là từ khóa trong ngôn ngữ C?

a/ double

b/ return

c/ do

d/ begin

Câu 13: Phát biểu nào sau đây đúng?

a/ Các từ khóa trong C chỉ được viết chữ hoa

b/ Các từ khóa trong C chỉ được viết chữ thường

c/ Các từ khóa trong C có thể viết chữ hoa hoặc chữ thường đều đúng

d/ Không có câu nào đúng

Câu 14: Trong Visual Studio C++, để chạy chương trình nhấn phím?

a/ F4

b/ F5

c/ F9

d/ Ctrl + F9

Câu 15: Trong ngôn ngữ C, hàm getch() chứa trong thư viện nào?

a/ stdio.h

b/ stdlib.h

c/ conio.h

d/ math.h

Câu 16: Trong ngôn ngữ C, các hàm printf(), scanf() chứa trong thư viện nào?

a/ stdio.h

b/ string.h

c/ windows.h

d/ math.h

Câu 17: Trong ngôn ngữ C, để khai báo một thư viện thì bắt đầu bằng ký tự:

a/ #

b/ *

c/ &

d/ \$

Câu 18: Trong ngôn ngữ C, hàm getch() dùng để:

a/ Dừng chương trình để chờ nhấn 1 phím bất kỳ

b/ Kết thúc chương trình

c/ Nhập một giá trị từ bàn phím

d/ Tất cả đều sai

Câu 19: Sinh viên năm thứ nhất ngành CNTT thường được học ngôn ngữ lập trình gì?

a/ Pascal

b/ C#

c/ C

d/ PHP

Câu 20: Bạn đã được học ngôn ngữ lập trình nào sau đây?

[]. C

[]. C++

[]. C#

[]. Pascal

[]. PHP

[]. Java

[]. Cobol

Câu 21: Bạn suy nghĩ gì khi học bài đầu tiên?

.....

.....

.....

.....

.....

BÀI 2: GIẢI THUẬT - MÃ GIẢ - LƯU ĐỒ

I. Giải thuật

Còn gọi là thuật toán, là các bước thực thi **rõ ràng, hữu hạn, có trình tự** thực hiện bài toán từ trạng thái bắt đầu đến khi đạt được kết quả mong muốn cuối cùng.

Ví dụ: giải thuật giải phương trình bậc nhất $P(x): ax+b=0$

1. Nếu $a=0$
 - a. nếu $b=0$ thì phương trình vô số nghiệm
 - b. nếu $b \neq 0$ thì phương trình vô nghiệm
2. Nếu $a \neq 0$
 - a. phương trình có nghiệm duy nhất $x = (-b/a)$

Giải thuật hiện nay thường dùng để chỉ giải thuật giải quyết các vấn đề tin học. Giải thuật giúp người lập trình hiểu được cách giải quyết bài toán và chuyển đổi thành chương trình.

Có hai cách mô tả giải thuật:

- Mô tả giải thuật bằng ngôn ngữ tự nhiên, gần giống với ngôn ngữ máy, chúng ta gọi là mã giả.
- Mô tả giải thuật bằng hình vẽ, chúng ta gọi là lưu đồ.

II. Mô tả giải thuật

II.1. Mã giả (Pseudocode):

Là một mô tả giải thuật lập trình máy tính ngắn gọn bằng các quy ước có cấu trúc của một số ngôn ngữ lập trình, nhưng bỏ đi những chi tiết không cần thiết để hiểu rõ giải thuật hơn, chẳng hạn bỏ đi khai báo biến, chương trình con, và những đoạn mã đặc biệt của hệ thống. Mã giả giúp chúng ta đọc hiểu chương trình dễ hơn khi đọc bằng ngôn ngữ lập trình. Mã giả được miêu tả cô đọng. Một chương trình viết bằng mã giả thì không thực thi được.

Một số chuẩn thường dùng trong mã giả:

Tuần tự: Một số từ khóa thực hiện các chỉ dẫn tuần tự

- Input: READ, INPUT, OBTAIN, GET
- Output: DISPLAY, PRINT, SHOW
- Compute: CALCULATE, COMPUTE, DETERMINE
- Initialize: SET, INIT
- Add one: INCREMENT, BUMP

Rẽ nhánh IF-ELSE

```
IF condition THEN
    sequence_1
ELSE
    sequence_2
ENDIF
```

Rẽ nhánh CASE

```
CASE expression
    condition_1: sequence_1
    condition_2: sequence_2
    condition_3: sequence_3
    ....
    OTHERS:
        sequence_n
ENDCASE
```

Vòng lặp WHILE

```
WHILE condition
    sequence
ENDWHILE
```

Vòng lặp DO WHILE

```
DO
    sequence
WHILE condition
```

Vòng lặp REPEAT-UNTIL

```
REPEAT
    sequence
```


UNTIL condition

Vòng lặp FOR

FOR i= init TO end

sequence

ENDFOR

Ví dụ 1: dùng mã giả để mô tả giải thuật nhập 2 số nguyên a, b. Xuất ra tổng, hiệu, tích thương.

BEGIN

INPUT a

INPUT b

tong = a+b

hieu = a-b

tich = a*b

IF $B \neq 0$ THEN

thuong = a/b

ENDIF

DISPLAY tong

DISPLAY hieu

DISPLAY tich

DISPLAY thuong

END

Ví dụ 2: dùng mã giả để mô tả giải thuật nhập vào 2 số a, b. Giải phương trình bậc nhất $ax + b = 0$.

BEGIN

INPUT a

INPUT b

IF a=0 THEN

IF b=0 THEN

DISPLAY “VSN”

ELSE

DISPLAY “VN”

ENDIF

ELSE

nghiem = -b/a

DISPLAY nghiem

ENDIF

END

Ví dụ 3: dùng mã giả để mô tả giải thuật nhập vào số nguyên dương N. Tính tổng $S = 1+2+3 + \dots + N$




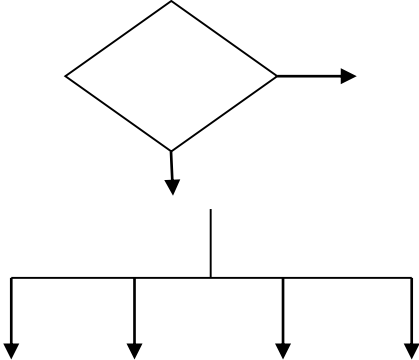

```

BEGIN
    DO
        INPUT N
        WHILE (N<=0) //BẮT BUỘC PHẢI NHẬP VÀO SỐ NGUYÊN DƯƠNG
        S = 0
        FOR i=1 TO N
            S = S+i
        ENDFOR
        DISPLAY S
    END

```

II.2. Lưu đồ (Flowchart)

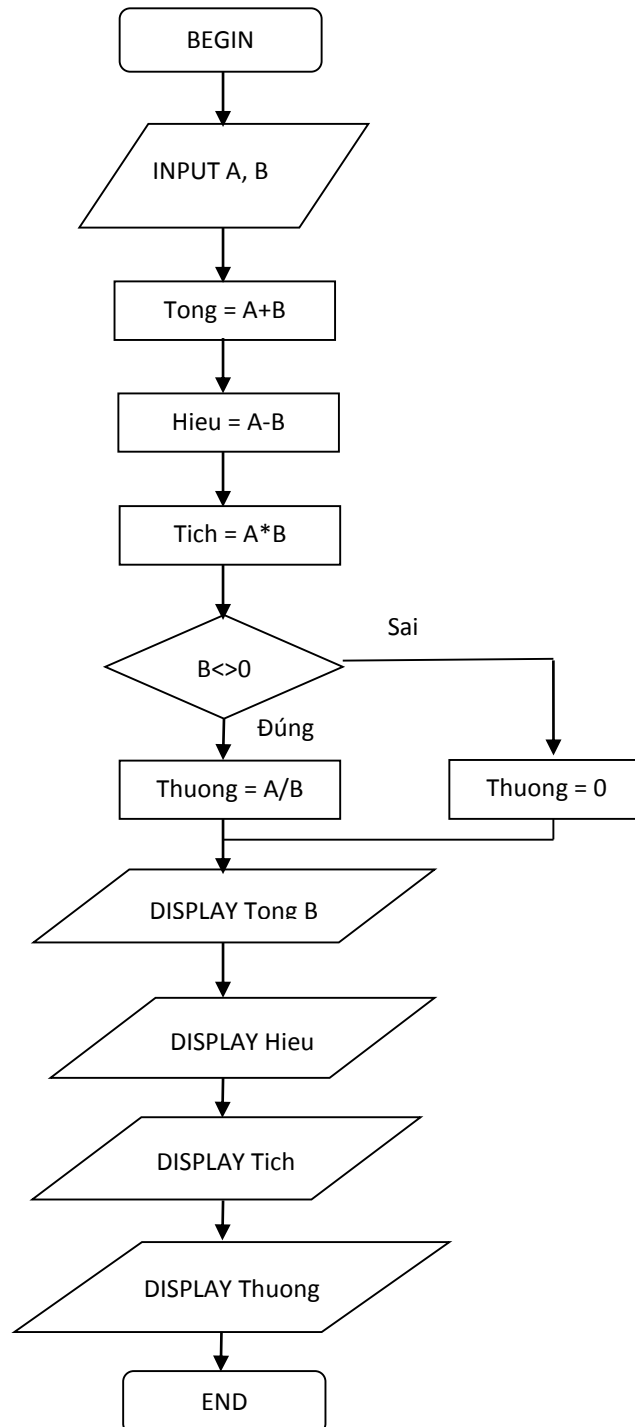
Dùng hình vẽ để mô tả các bước của giải thuật, giúp cho người đọc dễ hình dung. Một số hình vẽ quy ước trong lưu đồ:

	Bắt đầu/kết thúc
	Nhập xuất dữ liệu
	Tính toán, xử lý
	Điều kiện, rẽ nhánh
	Kết nối với phần tiếp theo



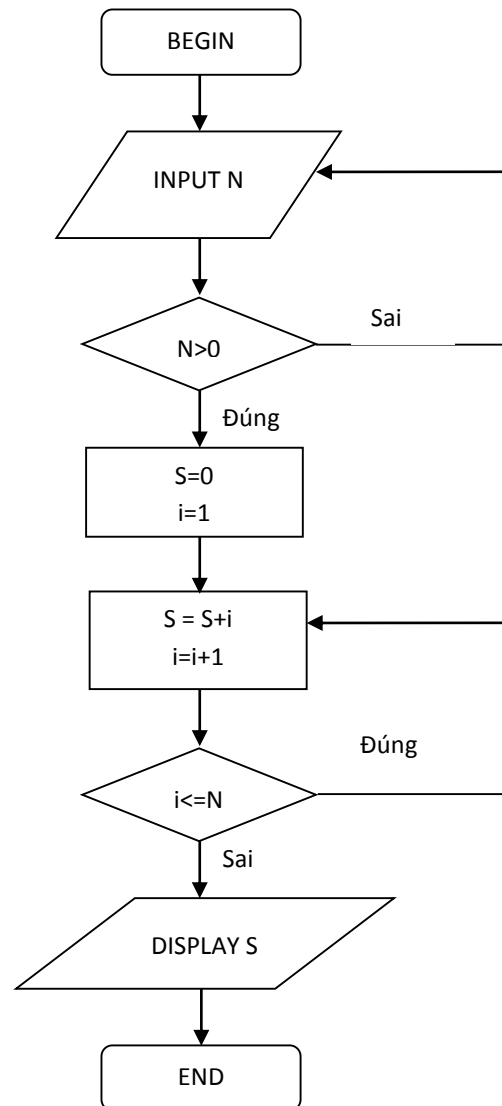
Hình 2.1: Lưu đồ giải thuật

Ví dụ 1: vẽ lưu đồ giải thuật, nhập vào 2 số A, B. Xuất ra Tổng, Hiệu, Tích, Thương.



Hình 2.2: Lưu đồ giải thuật ví dụ 1

Ví dụ 2: Vẽ lưu đồ giải thuật cho bài toán, nhập vào số nguyên N, xuất ra tổng $S = 1+2+\dots+N$.



Hình 2.3: Lưu đồ giải thuật ví dụ 2

III. Bài tập tự làm

Hãy vẽ lưu đồ giải thuật và mã giả cho các câu sau:

- Câu 1. Cho hằng số $PI=3.14$. Nhập vào bán kính r , tính và xuất ra chu vi, diện tích hình tròn.
- Câu 2. Nhập vào 3 số nguyên a, b, c . Giải phương trình bậc 2: $ax^2 + bx + c=0$

- Câu 3. Nhập vào điểm số của ba môn Toán, Lý, Hóa. Tính và xuất điểm trung bình, xếp loại học lực. Biết rằng giỏi nếu $TB \geq 8.0$, khá nếu $TB \geq 6.5$, Trung bình nếu $TB \geq 5.0$ và Yếu nếu < 5.0
- Câu 4. Nhập vào số thực X và số nguyên N . Tính và xuất ra màn hình tổng:
 $S = X + 2X + 3X + \dots + NX$
- Câu 5. Nhập vào 3 số thực a, b, c . Kiểm tra và cho biết chúng có tồn tại một tam giác có chiều dài 3 cạnh lần lượt là a, b, c không? Nếu có thì in ra diện tích và chu vi tam giác đó. Trong đó $S = \sqrt{p(p-a)(p-b)(p-c)}$ với p là nửa chu vi.
- Câu 6. Một bãi đậu xe hơi tính giá tối thiểu 2\$ cho 3 giờ đỗ xe (Nếu đỗ dưới 3 giờ vẫn phải trả 2\$), tính thêm 0.5\$ cho mỗi giờ thêm sau đó. Nếu đỗ xe trong suốt 24 giờ thì tính 10\$. Giả sử không có xe nào đậu quá 24 giờ. Viết chương trình cho nhập vào số giờ đỗ xe của một khách hàng, tính và in ra tiền đỗ xe cho khách hàng ấy.
- Câu 7. Nhập vào 2 số a, b . Tìm ước số chung lớn nhất của 2 số.
- Câu 8. Nhập vào số nguyên dương a , kiểm tra xem a có phải là số nguyên tố không?
- Câu 9. Nhập vào một số nguyên X . Kiểm tra xem X có là chẵn hay lẻ.
- Câu 10. Nhập vào một số nguyên X . Kiểm tra xem X có chia hết cho cả 2 và 3 hay không. Ví dụ: 6 chia hết cho cả 2 và 3.
- Câu 11. Nhập vào một số nguyên dương cơ số 10. Chuyển số này sang cơ số 2.
- Câu 12. Nhập vào số nguyên dương X . Kiểm tra xem X có phải là số chính phương hay không.

Câu hỏi trắc nghiệm:

Câu 1: Điều nào sau đây đúng với giải thuật:

- a/ Các bước rõ ràng, vô hạn thực hiện bài toán
- b/ Các bước hữu hạn, rõ ràng, có tuần tự thực hiện bài toán không cần thiết ra kết quả cuối cùng.
- c/ Các bước hữu hạn, rõ ràng, có tuần tự thực hiện bài toán đạt được kết quả mong muốn cuối cùng.
- d/ Một đoạn chương trình có thể chạy được.

Câu 2: Biểu diễn giải thuật bằng những cách nào sau đây:

- a/ Bảng mã giả hoặc lưu đồ
- b/ Bằng giọng nói

c/ Bảng đoạn văn

d/ Bảng các ngôn ngữ lập trình như Java, C, C++, C#

Câu 3: Đoạn mã giả sau thực hiện điều gì?

```
BEGIN
    INPUT a, b, c
    m=a
    IF m>b THEN m=b
    IF m>c THEN m=c
    DISPLAY m
END
```

a/ xuất giá trị lớn nhất trong 3 số a, b, c.

b/ xuất giá trị bé nhất trong 3 số a, b, c.

c/ m luôn có giá trị a nên xuất giá trị a.

d/ xuất giá trị c

Câu 4: Đoạn mã giả sau thực hiện điều gì?

```
BEGIN
    INPUT N
    S=0
    FOR I=1 TO N: I=I+2
        S = S +I
    ENDFOR
    DISPLAY S
END
```

a/ xuất tổng các số lẻ từ 1 đến N.

b/ xuất S có giá trị là 3.

c/ xuất s có giá trị là 0.

d/ xuất tổng các số từ 1 đến N.

Câu 5: Đoạn mã giả sau thực hiện điều gì?

```
BEGIN
    M = DanhSach[0] //DanhSach[i] là phần tử thứ i trong danh sách
                    // DanhSach[0] là phần tử đầu tiên
```

```
FOR I=1 TO N-1
  IF M>DanhSach[i] THEN
    M = DanhSach[i]
  ENDIF
ENDFOR
DISPLAY M
END
```

a/ xuất tổng các phần tử trong danh sách.

b/ xuất phần tử lớn nhất trong danh sách.

c/ xuất phần tử nhỏ nhất trong danh sách.

d/ xuất cuối cùng trong danh sách.

Ghi chú: *DanhSach là một tập hợp gồm nhiều phần tử các số nguyên và được đánh số thứ tự từ 0 đến n-1 (nếu trong danh sách có n phần tử). DanhSach[0] là phần tử đầu tiên, DanhSach[n-1] là phần tử cuối cùng.*

Câu 6: Đoạn mã giả sau thực hiện điều gì?

```
BEGIN
  M = DanhSach[0]  //DanhSach[i] là phần tử thứ i trong danh sách
                  // DanhSach[0] là phần tử đầu tiên
  FOR I=1 TO N-1
    IF M<DanhSach[i] THEN
      M = DanhSach[i]
    ENDIF
  ENDFOR
  DISPLAY M
END
```

a/ xuất tổng các phần tử trong danh sách.

b/ xuất phần tử lớn nhất trong danh sách.

c/ xuất phần tử nhỏ nhất trong danh sách.

d/ xuất cuối cùng trong danh sách.

Câu 7: Đoạn mã giả sau thực hiện điều gì?

```
BEGIN
    M = 0
    FOR I=0 TO N-1
        M = M + DanhSach[i]
    ENDFOR
    DISPLAY M
END
```

a/ xuất tổng các phần tử trong danh sách.

b/ xuất giá trị trung bình cộng các phần tử trong danh sách.

c/ xuất phần tử nhỏ nhất trong danh sách.

d/ xuất cuối cùng trong danh sách.

Câu 8: Đoạn mã giả sau thực hiện điều gì?

```
BEGIN
    M = 0
    FOR I=0 TO N-1
        M = M + DanhSach[i]
    ENDFOR
    IF N<>0 THEN
        DISPLAY M/N
    ENDIF
END
```

a/ xuất tổng các phần tử trong danh sách.

b/ xuất giá trị trung bình cộng các phần tử trong danh sách.

c/ xuất phần tử nhỏ nhất trong danh sách.

d/ xuất cuối cùng trong danh sách.

Câu 9: Đoạn mã giả sau thực hiện điều gì?

```
BEGIN
    DO
        INPUT N
    WHILE N<=0
```


END

a/ nhập N liên tục cho tới khi $N \leq 0$ thì dừng.

b/ nhập N và kiểm tra điều kiện $N \leq 0$ nếu đúng thì không nhập tiếp.

c/ nhập N và kiểm tra điều kiện $N \leq 0$ nếu đúng thì nhập tiếp tục.

d/ bắt buộc nhập N là số âm.

Câu 10: Đoạn mã giả sau thực hiện điều gì?

BEGIN

DO

INPUT N

WHILE $N \leq 0 \parallel N > 100$

END

a/ nhập N liên tục cho tới khi $N \leq 0$ và $N > 100$ thì dừng.

b/ nhập N với điều kiện N trong khoảng từ 1 đến 100 thì dừng.

c/ nhập N với điều kiện N trong khoảng từ 1 đến 100 thì nhập tiếp tục.

d/ bắt buộc nhập N là số âm hoặc lớn hơn 100.

BÀI 3: BIỂU THỨC VÀ CÁC PHÉP TOÁN

I. Biểu thức, toán hạng, toán tử

I.1. Khái niệm:

Biểu thức là sự kết hợp giữa các toán hạng và các toán tử.

Ví dụ sau đây là các biểu thức:

4

-6

4+21

a*(b + c/d)/20

q = 5*2

x = ++q % 3

q > 3

- Các toán hạng có thể là hằng, biến, hoặc kết hợp cả 2.
- Toán tử có thể là phép toán số học, phép toán logic, phép toán tăng giảm, hàm,...
- Một biểu thức có thể là sự kết hợp của nhiều biểu thức con.
- Sử dụng dấu cặp dấu ngoặc “(“ và “)” để nhóm các biểu thức con

I.2. Ví dụ minh họa:

Nhập số thực x và số nguyên n, tính giá trị của biểu thức sau:

$$Z = \frac{2x^n}{(\sqrt{x} + 1)} \quad z = (2 * \text{pow}(x,n)) / (\text{sqrt}(x) + 1)$$

Phân tích bài toán:

- Biểu thức Z trên có các toán hạng: biến số thực x, biến số nguyên n, các hằng số 2 và 1.
- Các toán tử gồm: nhân, chia, cộng, hàm mũ, hàm căn bậc 2.
- Biểu thức Z là sự kết hợp của các biểu thức con như $2x^n$ và $(\sqrt{x} + 1)$.
- Để làm bài này chúng ta khai báo biến x là kiểu số thực, biến n là số nguyên, biến Z là kiểu số thực (biến Z sẽ được gán bằng giá trị của biểu thức). Sử dụng các hàm tính số mũ và hàm tính căn bậc 2.

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
```

```
#include "conio.h"
```

```
#include "math.h" //Thư viện chứa hàm pow và sqrt
void main()
{
    int n;
    float x,Z=0;
    printf("Nhap n");
    scanf("%d",&n);
    printf("Nhap x");
    scanf("%f",&x);
    Z = (2 * pow(x,n))/(sqrt(x)+1); //tính toán biểu thức
    printf("Z = %.2f",Z); //In kết quả Z ra màn hình
    getch(); //Chờ nhấn phím bất kỳ để kết thúc
}
```

//Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
import java.math.*; //Thư viện chứa hàm pow và sqrt
public class ViduMinhhhoa
{
    public static void main(String [] arge)
    {
        int n;
        double x,Z=0;
        Scanner Input = new Scanner(System.in);
        System.out.print("Nhap n:");
        n = Input.nextInt();
        System.out.print("Nhap x:");
        x = Input.nextFloat();
        Z = (2 * Math.pow(x,n))/(Math.sqrt(x)+1);
        System.out.print("Z = " + Z);
    }
}
```

II. Các phép toán:

II.1. Phép toán số học:

Phép toán	Ý nghĩa	Ví dụ
+	Cộng	5+2=7

II.2. Phép toán quan hệ (so sánh):

Phép toán	Ý nghĩa	Ví dụ
>	Lớn hơn	$3 > 5 \rightarrow \text{sai}$
>=	Lớn hơn hoặc bằng	$3 >= 2 \rightarrow \text{đúng}$
<	Nhỏ hơn	$3 < 5 \rightarrow \text{đúng}$
<=	Nhỏ hơn hoặc bằng	$3 <= 5 \rightarrow \text{đúng}$
==	Bằng	$3 == 5 \rightarrow \text{sai}$
!=	Không bằng	$3 != 5 \rightarrow \text{đúng}$

- Trong ngôn ngữ C, kết quả của phép toán so sánh nếu đúng thì cho kết quả là 1 và sai là 0.
- Trong ngôn ngữ Java, kết quả của phép toán so sánh nếu đúng thì cho kết quả là true và sai là false.

II.3. Phép toán logic:

Phép toán	Ý nghĩa	Ví dụ
!	Not (phủ định)	!a
&&	And (và)	a && b
	Or (hoặc)	a b

- Trong ngôn ngữ C, kết quả của phép toán logic nếu đúng thì cho kết quả là 1 và sai là 0. Các số khác 0 xem là đúng.
- Trong ngôn ngữ Java, kết quả của phép toán logic nếu đúng thì cho kết quả là true và sai là false.

Ví dụ:

A	B	Kết quả		
		!A	A&&B	A B
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

II.4. Phép toán tăng giảm:

Phép toán	Ý nghĩa	Ví dụ
++	Tăng lên 1	n++
--	Giảm đi 1	n--

Lưu ý:

- Nếu phép toán ++ đứng trước biến thì tăng biến lên 1 trước khi sử dụng biến. Nếu phép toán ++ đứng sau biến thì tăng biến lên 1 sau khi sử dụng biến. Tương tự với phép toán --.
- Ví dụ : n = 3;
 $A = ++n;$ → ++n sẽ được thực hiện trước (n = 4), rồi mới gán cho A (lúc đó A cũng sẽ bằng 4).
 $A = n++;$ → A = n sẽ được thực hiện trước (lúc này n đang có giá trị là 3, nên A sẽ bằng 3), rồi mới tăng n lên 1.

II.5. Phép toán trên bit:

Phép toán	Ý nghĩa
~	Not bit (bù)
&	And bit (giao từng cặp bit)

	Or bit (hoặc từng cặp bit)
^	Xor bit (exclusive)-cặp bit khác nhau trả về 1
<<	Dịch bit trái
>>	Dịch bit phải

Các phép toán trên bit chỉ thực hiện trên các toán hạng có kiểu dữ liệu là số nguyên như: **char, int, long**.

Ví dụ:

Bit A	Bit B	Kết quả			
		~A	A & B	A B	A ^ B
0	0	1	0	0	0
0	1	1	0	1	1
1	0	0	0	1	1
1	1	0	1	1	0

II.5. Phép toán điều kiện:

<Điều kiện> ? <Biểu thức 1> : <Biểu thức 2>;

- Nếu <điều kiện> đúng → sẽ lấy kết quả của <biểu thức 1>
- Nếu <điều kiện> sai → sẽ lấy kết quả của <biểu thức 2>

Ví dụ:

$(5 > 7) ? 1 : 0;$ → kết quả là 0

$\text{int } x = (10 \% 3 == 1) ? 100 : 0;$ → kết quả $x = 100$

Biểu thức điều kiện này có thể thay cho cấu trúc điều kiện if...else... được trình bày trong bài 5.

II.5. Ví dụ minh họa:

Hãy nhập vào 2 số nguyên a và b. Tìm số lớn.

Phân tích bài toán:

- Để kiểm tra a và b, xem số nào lớn hơn, chúng ta phải dùng phép toán so sánh (lớn hơn > hoặc nhỏ hơn <).
- Trong bài này có quyết định rẽ nhánh (nếu $a > b$ thì, ngược lại thì....). Chúng ta chưa học cấu trúc if... else... nhưng vẫn giải quyết được bài toán này bằng cách sử dụng phép toán **điều kiện**.

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
#include "conio.h"
void main()
{
    int a, b, max;
    printf("Nhap a, b:");
    scanf("%d%d", &a, &b);
    max = (a > b) ? a : b; //sử dụng toán tử đk
    printf("Max = %d", max);
    getch();
}
```

//Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
import java.math.*;
public class ViduMinhhhoa
{
    public static void main(String [] arge)
    {
        int a, b, max;
        Scanner Input = new Scanner(System.in);
        System.out.print("Nhap a:");
        a = Input.nextInt();
        System.out.print("Nhap b:");
        b = Input.nextInt();
        max = (a > b) ? a : b;
        System.out.print("\nMax    "+ max);
    }
}
```

III. Bài tập tự làm

Câu 1: Viết chương trình nhập số nguyên n và số thực x tính các biểu thức:

$$1/ \quad Z = \frac{2x + \sqrt{n}}{13}$$

$$2/ \quad T = \frac{1}{2}xn + 2.32x^3$$

$$3/ \quad K = (x^2 + x + 1)^n + (x^2 - x + 1)^n$$

Câu 2: Viết chương trình nhập số lượng, đơn giá của một mặt hàng. Hãy tính và in ra kết quả ra màn hình có 2 số lẻ:

- Giảm giá = Số lượng * đơn giá * 12%
- Cước vận chuyển = Số lượng * đơn giá * 5%
- Số tiền phải trả = Số lượng * đơn giá – giảm giá + cước vận chuyển

Câu 3: Viết chương trình nhập một số nguyên dương có 2 chữ số. Hãy in ra màn hình số đảo ngược của nó. Ví dụ nhập 51 in ra 15.

Câu 4: Viết chương trình nhập bốn số nguyên a, b, c, d. Tìm số nhỏ nhất.

Câu 5: Viết chương trình nhập một số nguyên dương có 2 chữ số. Hãy tính tổng 2 chữ số của số vừa nhập. Ví dụ nhập 51, in ra tổng bằng 6.

Câu 6: Viết chương trình nhập số nguyên n, số thực x. Hãy tính căn bậc n của x.

Câu 7: Viết chương trình nhập giờ, phút, giây. Hãy tính và in ra màn hình tổng số giây.

Câu 8: Viết chương trình nhập bán kính, tính và in ra màn hình diện tích hình tròn.

Câu 9: Viết chương trình nhập đáy lớn, đáy bé, chiều cao của hình thang cân. Tính và in ra màn hình chu vi của hình thang.

Câu 10: Viết chương trình nhập 1 số nguyên dương N ($10 > N > 0$), hãy in ra bảng cửu chương của N.

Câu hỏi trắc nghiệm:

Câu 1: Hãy cho biết kết quả của Z khi thực hiện lệnh: `int Z = 10-6*2 + 1;`

a/ 9

b/ -1

c/ 12

d/ Một kết quả khác

Câu 2: Hãy cho biết kết quả của Z khi thực hiện lệnh: `int Z = (1==2-1);`

a/ 1

b/ 0

c/ 2

d/ Một kết quả khác

Câu 3: Hãy cho biết kết quả của Z khi thực hiện lệnh: `int Z = (2>3)?5:10;`

a/ 5

b/ 2

c/ 3

d/ 10

Câu 4: Kết quả biểu thức `17 % 2014` là?

a/ 2014

b/ 17

c/ 118

d/ Một kết quả khác

Câu 5: Để tính căn bậc n của x, làm như sau:

a/ `pow(x,1.0/n);`

b/ `pow(x,1/n);`

c/ `sqrt(x,n);`

d/ `pow(x,n);`

Câu 6: Hãy cho biết kết quả của câu lệnh: `printf("Z = %.3f",6.6666);`

a/ 6.6

b/ 6.667

c/ 6.6666

d/ 6.666

Câu 7: Kết quả của biểu thức `10 / 4` là?

a/ 2.5

b/ 2

c/ 10

d/ 4

Câu 8: Hãy cho biết kết quả của N khi thực hiện các lệnh sau:

`int a = 9;`

`int N = 2 + a++;`

a/ 12

b/ 13

c/ 11

d/ 10

Câu 9: Hãy cho biết kết quả của N khi thực hiện các lệnh sau:

`int a = 9;`

`int N = --a + 2;`

a/ 10

b/ 11

c/ 12

d/ Là một kết quả khác

Câu 10: Hãy cho biết kết quả của N khi thực hiện các lệnh sau:

`int a = 9;`

`int N = --(--a) + 2;`

a/ 9

b/ 10

c/ 11

d/ Là một kết quả khác

Câu 11: Các phép toán tăng ++ và giảm --, sẽ thực hiện tăng hoặc giảm một lần là bao nhiêu?

a/ 4

b/ 3

c/ 2

d/ 1

Câu 12: Các phép toán trên bit sẽ chỉ thực hiện được trên các kiểu dữ liệu nào?

a/ int, long

b/ int, long, char

c/ int, long, double

d/ float, double, char

Câu 13: Các phép toán logic luôn cho kết quả là?

a/ Hoặc đúng hoặc sai

b/ Đúng

c/ Sai

d/ Tất cả a, b, c đều sai

Câu 14: Hãy cho biết kết quả của biểu thức $!(2 > 4 - 1) \ \&\& \ (1 == 3 - 2)$

a/ Đúng

b/ Sai

Câu 15: Hãy cho biết kết quả của N khi thực hiện các lệnh sau:

int a = 10, b = 4;

float N = (float) a/b;

a/ 2

b/ 2.5

c/ 10

d/ 4

Câu 16: Hãy cho biết kết quả in ra màn hình khi thực hiện lệnh sau:

```
printf(“%d”,’A’);
```

a/ A

b/ 64

c/ 65

d/ 66

Câu 17: Hãy cho biết phép toán nào sau đây có độ ưu tiên thấp nhất?

a/ nhân *

b/ chia /

c/ lấy phần dư %

d/ cộng +

Câu 18: Hãy cho biết kết quả in ra màn hình khi thực hiện các lệnh sau:

```
int a = 4, b = 7;
```

```
int max = (a<b)? a:b;
```

```
printf(“%d”,max);
```

a/ 7

b/ 4

c/ 11

d/ 3

Câu 19: Hãy cho biết kết quả in ra màn hình khi thực hiện các lệnh sau:

```
int a=3, b=6, c=3, d=1;
```

```
int Z = a + b * (c - 2) / 10
```

```
printf(“%d”,Z);
```

a/ 3

b/ 3.6

c/ 0

d/ 0.9

Câu 20: Hãy cho biết kết quả in ra màn hình khi thực hiện các lệnh sau:

```
int a=3, b=6, c=3, d=1;  
float Z = a + b * (c - 2) / 10.0;  
printf("%.0f",Z);
```

a/ 3

b/ 4

c/ 3.6

d/ Một kết quả khác

Câu 21: Bạn suy nghĩ gì khi học xong bài thứ 3?

.....

.....

.....

.....

.....

BÀI 4: KIỂU DỮ LIỆU – HẲNG – BIẾN – NHẬP XUẤT

I. Kiểu dữ liệu

I.1. Khái niệm:

Kiểu dữ liệu (Datatype) **xác định** những **loại dữ liệu khác nhau**, chẳng hạn như số thực (real value), số nguyên (integer), luận lý (boolean), ký tự (character). Kiểu dữ liệu xác định kích thước, những phép tính (operator), nghĩa và cách thức lưu trữ loại dữ liệu đó.

- Ngôn ngữ lập trình khác nhau có những thuật ngữ khác nhau cho kiểu dữ liệu. Một số những kiểu dữ liệu cơ bản là:

I.1.1 Kiểu ký tự (character):

- Kích thước: mỗi ký tự chiếm 1 byte (8 bit) bộ nhớ, biểu diễn được 256 giá trị khác nhau, các ký tự được biểu diễn thông qua bảng mã ASCII.
- Ví dụ: 'A' có giá trị 65, 'B' có giá trị 66
- Ngoài ra, kiểu ký tự cũng được xem là kiểu số nguyên chiếm 1 byte bộ nhớ. Có 2 loại số nguyên kiểu char: unsigned char và char.
- unsigned char là số nguyên không dấu - miền giá trị 0 → 255
- char là số nguyên có dấu – miền giá trị -127 → 128
- Phân loại ký tự: có thể chia 256 ký tự thành 3 nhóm:
 - o Các ký tự điều khiển: có mã từ 0 đến 31, chẳng hạn ký tự có mã 13 là ký tự chuyển con trỏ về đầu dòng. Các ký tự trong nhóm này không hiển thị ra màn hình. Ví dụ: Mã số 8 là ký tự Backspace, mã số 9 là ký tự Tab.
 - o Các ký tự văn bản: có mã từ 32 đến 126. Ví dụ: 65 là mã ký tự 'A', 66 là 'B', 67 là 'C',... 97 là mã ký tự 'a', 98 là 'b',...
 - o Các ký tự đồ họa: có mã từ 127 đến 255. Các ký tự này có thể hiển thị ra màn hình. Ví dụ: 228 là mã ký tự 'Σ', 251 là mã ký tự '√'

I.1.2 Kiểu số nguyên (integer):

- Tên gọi: Trong C có 3 kiểu số nguyên cơ bản char, int và long. Ngoài ra để thể hiện số nguyên không dấu còn có unsigned char, unsigned int, unsigned long.

- Kích thước và miền giá trị của chúng như sau:

Kiểu	Miền giá trị	Kích thước
char	-127 \rightarrow 128	1 byte
unsigned char	0 \rightarrow 255	1 byte
int	-2.147.483.648 \rightarrow 2.147.483.647	4 byte
unsigned int	0 \rightarrow 4.294.967.295	4 byte
long	-2.147.483.648 \rightarrow 2.147.483.647	4 byte
unsigned long	0 \rightarrow 4.294.967.295	4 byte

1.1.3 Kiểu số thực:

Trong C cho phép sử dụng 3 loại kiểu số thực là float, double, long double. Kích thước và miền giá trị của chúng như sau:

Kiểu	Miền giá trị	Kích thước
float	$3.4 * (10^{-38}) \dots 3.4 * (10^{+38})$	4 byte
double	$1.7 * (10^{-308}) \dots 1.7 * (10^{+308})$	8 byte
long double	$3.4 * (10^{-4932}) \dots 1.1 * (10^{+4932})$	10 byte

II. Biến (Variable):

Cho một đoạn chương trình như sau: lưu số 5 và số 8 vào 2 ô nhớ khác nhau, sau đó thêm 1 vào số thứ nhất, rồi lấy hiệu của 2 số lưu vào ô nhớ thứ 3.

```
a = 5;
b = 8;
a = a + 1;
result = a - b;
```

Trong ví dụ trên thì a, b, result ta gọi là biến. Như vậy: **biến là tên gọi của ô nhớ, biến dùng để lưu trữ giá trị, có thể thay đổi giá trị.**

Để sử dụng biến trong C, phải khai báo biến, xác định kiểu dữ liệu của biến muốn sử dụng. Cú pháp khai báo biến như sau:

Kieudulieu tenbien;

Ví dụ:

Khai báo một biến a có kiểu số nguyên, b có kiểu số thực.

```
int a;
double b;
```


Có thể khai báo nhiều biến cùng kiểu dữ liệu như sau:

`int a, b, c; // biến a,b,c có kiểu số nguyên.`

hoặc:

`int a;`

`int b;`

`int c;`

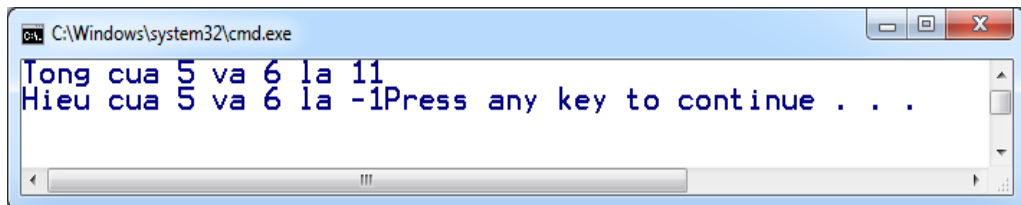
Có thể khởi tạo giá trị ban đầu cho biến.

`int a=5, b; // biến a và b có kiểu số nguyên. Khởi tạo a bằng 5.`

Ví dụ:

Khai báo và khởi tạo biến a bằng 5 và b bằng 6. Tính và xuất ra tổng hiệu của hai số.

```
#include <stdio.h>
void main()
{
    int a = 5, b=6;
    int tong = a+b;
    int hieu = a-b;
    printf("Tong cua %d va %d la %d\n", a, b, tong);
    printf("Hieu cua %d va %d la %d", a, b, hieu);
}
```



Hình 4.1: Kết quả chương trình tính tổng, hiệu 2 số

III. Hằng (Constant)

Là một đại lượng mà giá trị của nó **không thay đổi** trong quá trình tính toán.

III.1. Các loại hằng:

Hằng số int: Là số nguyên có giá trị từ $-2^{31} \rightarrow 2^{31} - 1$.

Hằng số long: Là số nguyên có miền giá trị từ $-2^{31} \rightarrow 2^{31} - 1$. Có 2 cách viết: 456389 hoặc 456389L. (Ta có thể thêm L hoặc l sau số nguyên. Những số ngoài miền giá trị của int là số long)

Ngoài ra, ta cũng có thể biểu diễn hằng nguyên dưới dạng hệ 8 và hệ 16

Ví dụ:

Hệ 8: 0123 - biểu diễn số có giá trị: $1*8*8 + 2*8 + 3 = 83$

Hệ 16: 0XA9 - biểu diễn số có giá trị $10*16 + 9 = 169$

Hằng số thực (float và double): Thể hiện các giá trị số thực. Có 2 cách thể hiện:

- Cách 1: (Dạng thập phân) gồm phần nguyên và phần phân. Ví dụ 214.35; 234.0
- Cách 2: (Dạng khoa học) số được tách thành 2 phần, phần định trị là số nguyên hoặc số thực dạng thập phân, phần bậc là số nguyên. Hai phần này cách nhau bởi ký tự e hoặc E.

Ví dụ: 123.456E3 (biểu diễn số 123456.0)

Hằng ký tự: Biểu diễn một ký tự, được đặt trong dấu nháy đơn ‘a’. Hằng ký tự ‘a’ có giá trị là mã của ký tự a trong bảng mã ASCII – có giá trị 97. Hằng ký tự có thể tính toán được.

Ví dụ: ‘9’ – ‘7’ = 2

Hằng chuỗi ký tự: là một chuỗi ký tự được đặt trong dấu nháy đôi “”.

Ví dụ: “Hello”

III.2. Khai báo hằng:

Cách 1:

#define tên_hằng giá_trị

Ví dụ: #define PI 3.14

Cách 2:

const tên_hằng = giá_trị;

Ví dụ: const PI = 3.14;

Cách 3:

const kiểu_DL tên_hằng = giá_trị;

Ví dụ: const double PI = 3.14;

Cách 4: dùng trực tiếp.

Ví dụ: ChuVi = 2*3.14*R; //3.14 là hằng số PI

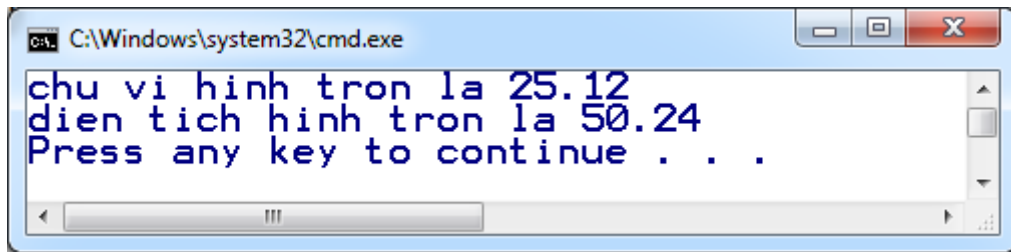
Lưu ý: Cách này không nên dùng vì trong chương trình có sử dụng ở nhiều nơi nếu muốn thay đổi ta phải thay đổi hằng này ở tất cả mọi nơi.

Ví dụ:

Cho bán kính. Tính chu vi và diện tích hình tròn

```
#include <stdio.h>
const double PI = 3.14; // Khai báo hằng PI
void main()
{
    double chuvi, dientich, r=4.0;
    chuvi = 2*PI*r;
    dientich = PI*r*r;
    printf("chu vi hình tron la %.2lf\n", chuvi);
    printf("dien tích hình tron la %.2lf", dientich);
}
```

Ghi chú: %.2lf là định dạng hiển thị số thực với 2 số lẻ



Hình 4.2: Kết quả chương trình tính diện tích, chu vi hình tròn.

IV. Kiểu enum

Giúp chúng ta định nghĩa một số kiểu dữ liệu mà trong C không định nghĩa.

cú pháp: `enum tên_kiểu {phần tử 1, phần tử 2 ...}`

Ví dụ:

```
enum Boolean {False, True} ;
```

Boolean được xem như là một kiểu dữ liệu mới miễn giá trị là True và False. Các phần tử sẽ mặc định gán giá trị, bắt đầu là 0. Trong ví dụ trên, False có giá trị 0, True có giá trị 1 và cũng có thể gán trực tiếp giá trị cho các phần tử.

Ví dụ:

```
enum Boolean {True=1, False=0};
```

Ví dụ:

```
#include <stdio.h>
enum Boolean {False, True};
void main()
{
    Boolean b, c; //b và c chỉ có thể gán True hoặc False
    b = True; // nếu xuất ra màn hình b sẽ có giá trị 1
    c = 1;      // báo lỗi
}
```

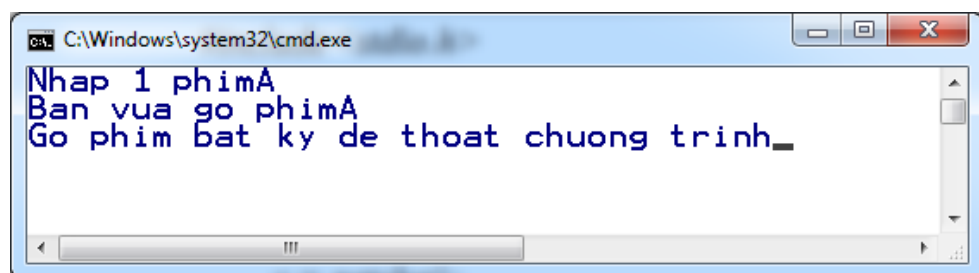
V. Nhập xuất (Input - Output)

V.1. Nhập xuất ký tự với conio.h

getche(): Nhập một ký tự bất kỳ có hiển thị phím bấm
getch(): Nhập một ký tự bất kỳ không có hiển thị phím bấm
putch(char): Xuất ký tự ra màn hình

Ví dụ:

```
#include <stdio.h>
#include <conio.h>
void main()
{
    char c;
    printf("Nhap 1 phim");
    c = getche();
    printf("\nBan vua go phim");
    putch(c);
    printf("\nGo phim bat ky de thoat chuong trinh");
    getch();
}
```

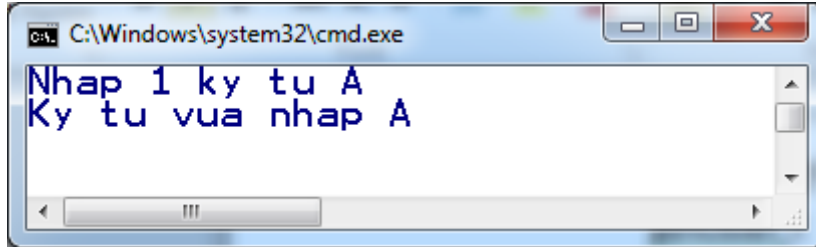


Hình 4.3: Kết quả chương trình dùng hàm `getche()`, `putch()`

V.2. Nhập xuất ký tự với stdio.h

getchar(): Nhập một ký tự từ bàn phím

putchar(char): xuất một ký tự ra màn hình



Hình 4.4: Kết quả chương trình dùng hàm getchar(), putchar()

V.3. Nhập xuất có định dạng với stdio.h

Để nhập và xuất ta dùng hàm scanf và printf

Cú pháp:

printf("chuỗi định dạng", biến, biến,...);

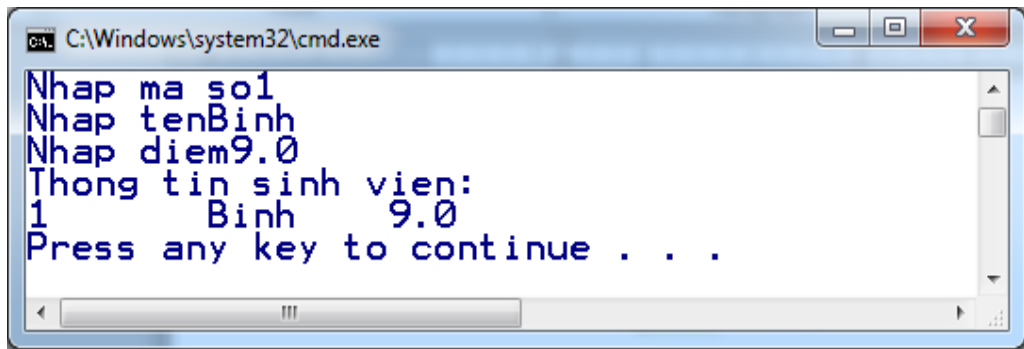
scanf("%định dạng", & biến);

Tùy theo kiểu của biến sẽ có chuỗi định dạng tương ứng. char(%c), int (%d), long(%ld), float(%f), double(%lf), string (%s).

Ví dụ:

Viết chương trình nhập vào thông tin một sinh viên gồm mã số, tên, tuổi và xuất ra màn hình.

```
#include <stdio.h>
void main()
{
    int maso;
    char ten[20];
    float diem;
    printf("Nhap ma so"); scanf("%d", &maso);
    printf("Nhap ten"); scanf("%s", &ten);
    printf("Nhap diem");scanf("%f", &diem);
    printf("Thong tin sinh vien:\n");
    printf("%d      %s      %.1f", maso, ten, diem);
}
```



Hình 4.5: Kết quả chương trình nhập và xuất thông tin sinh viên.

Lưu ý: Hàm `scanf` và định dạng `%s` không nhập được chuỗi có khoảng trắng. Muốn nhập chuỗi có khoảng trắng ta dùng hàm `gets(char[])` hoặc dùng hàm `scanf` có định dạng `scanf("%[A-Za-z0-9]", &biến);`

V.4. Các ký tự định dạng

Dữ liệu nhập hay xuất có kiểu gì, chúng ta phải sử dụng các ký tự định dạng tương ứng. Sau đây là một số ký tự định dạng cho các kiểu dữ liệu căn bản:

Kiểu dữ liệu	Ký tự định dạng
char	%c
int	%d
long	%ld
float	%f
double	%lf
string	%s

VI. Một số bài tập minh họa

Bài 1: Nhập vào 2 số a, b in ra kết quả các phép tính a+b, a-b, a/b, a*b.

Ví dụ: nhập a = 6, b = 4

$$6 + 4 = 10$$

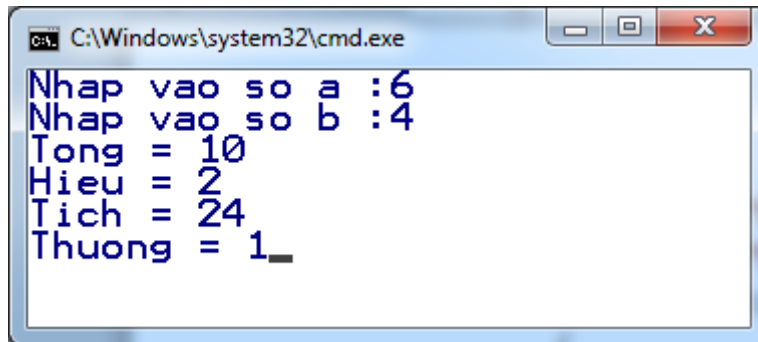
$$6 - 4 = 2$$

$$6 \times 4 = 24$$

$$6 / 4 = 1$$

```
#include "stdio.h"
#include "conio.h"
void main()
{
```

```
int a, b;
printf("Nhap vao so a :");
scanf("%d",&a);
printf("Nhap vao so b :");
scanf("%d",&b);
printf("Tong = %d",a+b);
printf("\nHieu = %d",a-b);
printf("\nTich = %d",a*b);
printf("\nThuong = %d",a/b);
getch();
}
```

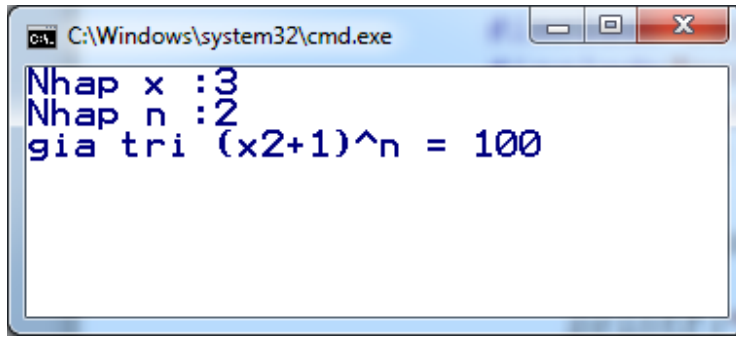


Hình 4.6: Kết quả chương trình nhập vào 2 số a, b , xuất ra tổng, hiệu, tích thương

Bài 2: Viết chương trình nhập vào số nguyên n và số thực x . Tính và in ra $(x^2 + 1)^n$

```
#include "stdio.h"
#include "conio.h"
#include "math.h"
void main()
{
    double x, giatri;
    int n;
    printf("Nhap x :");
    scanf("%lf",&x);
    printf("Nhap n :");
    scanf("%d",&n);
    giatri = pow(x*x+1,n);

    printf("gia tri (x^2+1)^n = %.0lf",giatri);
    getch();
}
```



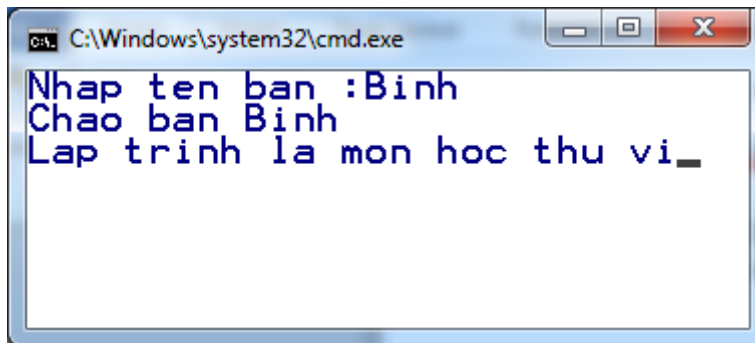
Hình 4.7: Kết quả chương trình tính biểu thức $(x^2 + 1)^n$

Bài 3: Viết chương trình nhập vào tên của bạn. Xuất ra màn hình

”Chao ban”

”Lap trinh la mon hoc thu vi”

```
#include "stdio.h"
#include "conio.h"
void main()
{
    char tenban[15]; // luu tru ten cua ban
    printf("Nhap ten ban :");
    scanf("%s",&tenban);
    printf("Chao ban %s \n",tenban);
    printf("Lap trinh la mon hoc thu vi");
    getch();
}
```



Hình 4.8: Kết quả chương trình nhập xuất chuỗi.

VII. Bài tập tự làm

Bài 1: Viết chương trình nhập vào điểm số của ba môn Toán, Lý, Hóa. Tính và in ra điểm trung bình.

Bài 2: Nhập vào một số n. In ra bảng cửu chương n

Bai 3: Nhập vào độ dài một cạnh hình vuông. Xuất chu vi và diện tích hình vuông này.

Bai 4: Viết chương trình cho phép nhập vào 1 kí tự. In ra mã ASCII của kí tự tương ứng và mã của kí tự đứng trước và đứng sau của kí tự đó.

Bai 5: Viết chương trình nhập vào độ dài chiều cao, 2 cạnh đáy, cạnh bên. Tính chu vi và diện tích hình thang cân. (diện tích: $\frac{1}{2}(a+b)h$)

Bai 6: Cho hằng số $PI=3.14$. Nhập vào bán kính r . Hãy in ra chu vi và diện tích hình tròn.

Bai 7: Nhập vào thông tin một hóa đơn bao gồm mã số hóa đơn, ngày hóa đơn, tên nhân viên bán hàng, tên khách hàng, tên hàng hóa, số lượng, đơn giá. Xuất thông tin hóa đơn ra màn hình theo mẫu:

So hoa don: 1	Ngày HD: 10/1/2014		
Nhan Vien: Tran Tinh	Khach hang: Nguyen Binh		

Ten hang	So luong	Don gia	Thanh tien
Sach	5	35000	175000

Câu hỏi trắc nghiệm

Câu 1: Điều nào sau đây đúng với định nghĩa **biến**:

a/ là tên gọi của một loại dữ liệu

b/ là các ký tự chẳng hạn như a, b, c.

c/ là tên gọi của một ô nhớ, dùng để lưu trữ giá trị, biến có thể thay đổi giá trị.

d/ chỉ có thể lưu trữ được số nguyên.

Câu 2: Điều nào sau đây đúng với định nghĩa **hằng**:

a/ là tên gọi của một loại dữ liệu

b/ là các ký tự chẳng hạn như a, b, c.

c/ là tên gọi của một ô nhớ, dùng để lưu trữ giá trị, hằng có thể thay đổi giá trị.

d/ là tên gọi của một ô nhớ, dùng để lưu trữ giá trị, hằng không thể thay đổi giá trị.

Câu 3: Điều nào sau đây đúng với định nghĩa **kiểu dữ liệu**:

- a/ là tên gọi của một loại dữ liệu
- b/ là các ký tự chẳng hạn như a, b, c.
- c/ là tên gọi của một ô nhớ, dùng để lưu trữ giá trị, hằng có thể thay đổi giá trị.

d/ là kiểu định dạng số trong ngôn ngữ lập trình.

Câu 4: Khai báo nào sau đây đúng với khai báo **hằng**:

- a/ const PI = 3.14;
- b/ const double PI = 31.4
- c/ #define PI 3.14

d/ Tất cả đều đúng.

Câu 5: Nhập một ký tự từ bàn phím ta có thể dùng hàm nào sau đây:

- a/ getch()
- b/ getchar()
- c/ getch()
- d/ Tất cả đều đúng.

Câu 6: Xuất một ký tự ra màn hình ta dùng hàm nào sau đây là sai:

- a/ putch()
- b/ putchar()
- c/ printf()
- d/ putche().

Câu 7: Các định dạng nhập xuất nào sau đây đúng với thứ tự: char, int, float, double

- a/ %d, %s, %f, %lf
- b/ %c, %d, %f, %lf
- c/ %s, %d, %f, %lf
- d/ %d, %s, %f, %s

Câu 8: Cách nào sau đây nhập chuỗi s từ bàn phím

- a/ scanf("%s", &s);
- b/ gets(s);
- c/ scanf("%[A-Za-z0-9]", &s);
- d/ Tất cả đều đúng.

Câu 9: Cách nào sau đây nhập số nguyên **x** từ bàn phím

a/ `scanf("%d", &x);`

b/ `getint(x);`

c/ `scanf("%f", &x);`

d/ `scanf("%d", x);`

Câu 10: Cách nào sau đây nhập số thực **double b** từ bàn phím

a/ `scanf("%f", &x);`

b/ `scanf("%lf", &x);`

c/ `scanf("%f", x);`

d/ `scanf("%lf", x);`

BÀI 5: CẤU TRÚC ĐIỀU KHIỂN

I. Cấu trúc if

I.1. Cú pháp:

- Trường hợp 1: Nếu biểu thức điều kiện đúng chương trình sẽ thực hiện lệnh ngay sau if, ngược lại chương trình sẽ bỏ qua lệnh đó.

if (điều kiện)
lệnh;

Ví dụ:

Ngôn ngữ C:

```
.....  
.....  
int age = 18;  
if (age >= 18)  
    printf("Ban da truong thanh");  
.....  
.....
```

Ngôn ngữ Java:

```
.....  
.....  
int age = 18;  
if (age >= 18)  
    System.out.print("Ban da truong thanh");  
.....  
.....
```

- Trường hợp 2: Nếu biểu thức điều kiện đúng chương trình sẽ thực hiện các lệnh nằm trong cặp dấu hoặc nhọn {...}, ngược lại chương trình sẽ bỏ qua chúng.

if (điều kiện)
{
lệnh 1;
lệnh 2;
...
lệnh n;
}

Ví dụ:

Ngôn ngữ C:

```

.....
.....
int age = 22;
if (age >= 18)
{
    printf("Ban da truong thanh");
    printf("Nam nay ban %d tuoi", age);
}
.....
.....

```

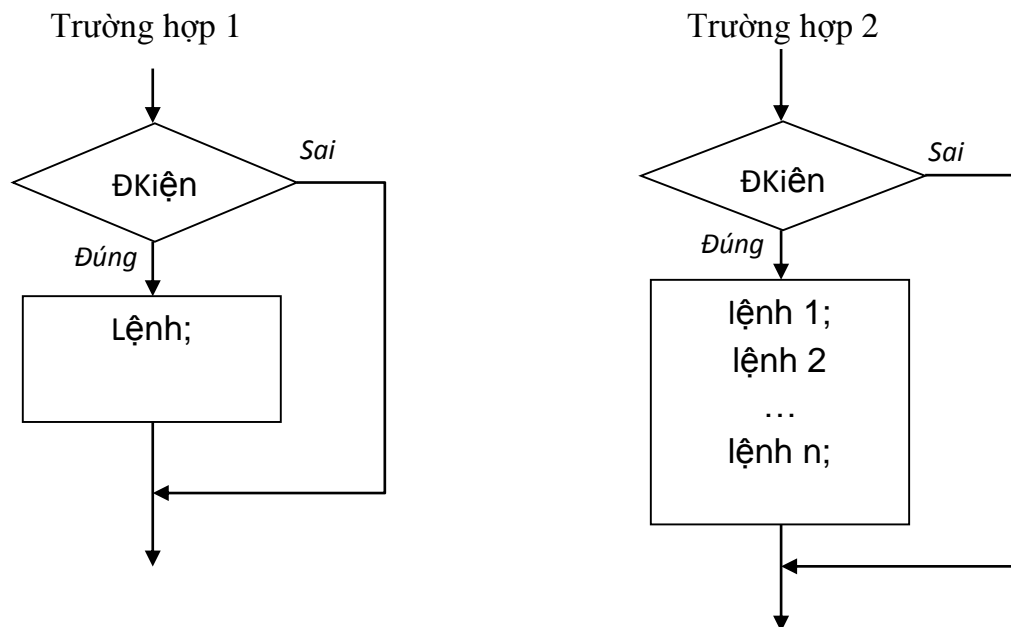
Ngôn ngữ Java:

```

.....
.....
int age = 22;
if (age >= 18)
{
    System.out.print("Ban da truong thanh");
    System.out.print("Nam nay ban " + age + " tuoi");
}
.....
.....

```

I.2. Lưu đồ:

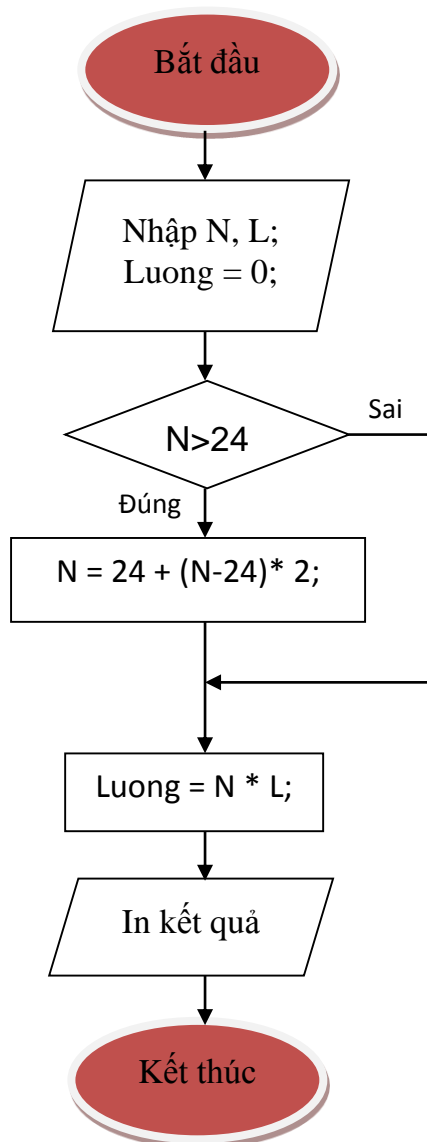


Hình 5.1: Lưu đồ cấu trúc điều kiện if

I.3. Ví dụ minh họa:

- **Ví dụ 1:** Nhập N là số ngày công và L là tiền lương một ngày. Tính tiền lương được lãnh, tuy nhiên nếu số ngày công mà lớn hơn 24 thì mỗi ngày làm thêm được hưởng 2 lần.

//Lưu đồ giải thuật:



Hình 5.2: Lưu đồ giải thuật ví dụ 1

//Mã giả

BEGIN

INPUT N, L

IF N > 24 THEN

$N = 24 + (N-24) * 2$

ENDIF

Luong = N * L;

DISPLAY Luong

END.

// Chương trình viết bằng ngôn ngữ C:

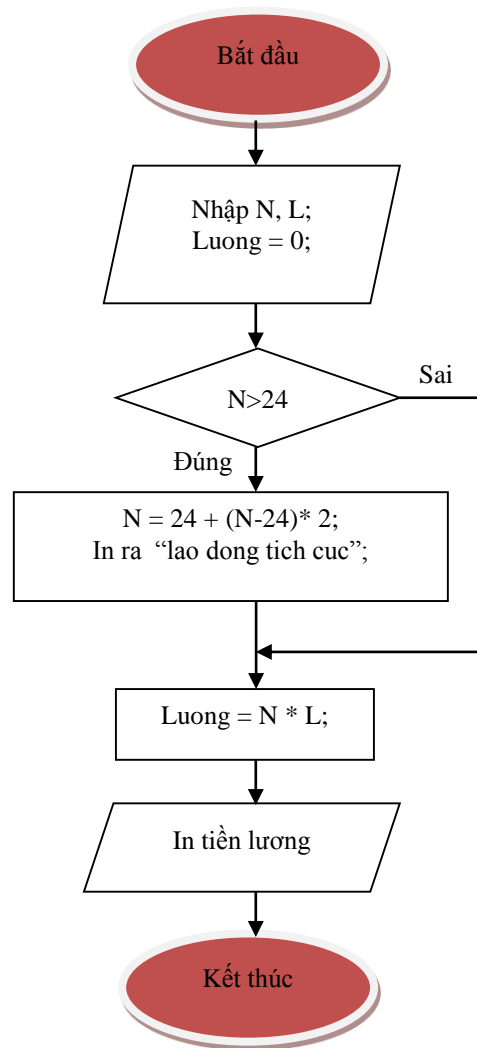
```
#include "stdio.h"
#include "conio.h"
void main()
{
    int N, L;
    float Luong = 0;
    printf("Nhap So ngay cong :");
    scanf("%d",&N);
    printf("Nhap tien luong 1 ngay cong :");
    scanf("%d",&L);
    if (N>24)
        N = 24 + (N-24)*2;
    Luong = N * L;
    printf("Tien luong = %.0f",Luong);
    getch();
}
```

//Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
public class Vidul
{
    public static void main(String [] arge)
    {
        int N, L;
        float Luong = 0;
        Scanner Input = new Scanner(System.in);
        System.out.print("So ngay cong :");
        N = Input.nextInt();
        System.out.print("Tien luong 1 ngay cong :");
        L = Input.nextInt();
        if (N>24)
            N = 24 + (N-24)*2;
        Luong = N * L;
        System.out.println("Tien luong = " + Luong);
    }
}
```

}
}

- **Ví dụ 2:** Nhập N là số ngày công và L là tiền lương một ngày. Tính tiền lương được lãnh, tuy nhiên nếu số ngày công mà lớn hơn 24 thì thông báo là lao động tích cực, đồng thời mỗi ngày làm thêm được hưởng 2 lần.



Hình 5.3: Lưu đồ giải thuật ví dụ 2

//Mã giả

BEGIN

INPUT N, L

IF N > 24 THEN


```
        N = 24 + (N-24) * 2
        DISPLAY Lao động tích cực
    ENDIF
    Luong = N * L;
    DISPLAY Luong
END.
```

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
#include "conio.h"
void main()
{
    int N, L;
    float Luong = 0;
    printf("Nhap So ngay cong :");
    scanf("%d",&N);
    printf("Nhap tien luong 1 ngay cong :");
    scanf("%d",&L);
    if (N>24)
    {
        N = 24 + (N-24)*2;
        Printf("lao dong tích cực");
    }
    Luong = N * L;
    printf("Tien luong = %.0f",Luong);
    getch();
}
```

//Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
public class Vidu2
{
    public static void main(String [] arge)
    {
        int N, L;
        float Luong = 0;
        Scanner Input = new Scanner(System.in);
        System.out.print("So ngay cong :");
        N = Input.nextInt();
        System.out.print("Tien luong 1 ngay cong :");
    }
}
```

```
L = Input.nextInt();
if (N>24)
{
    N = 24 + (N-24) * 2;
    System.out.println("Lao dong tinh cuc");
}
Luong = N * L;
System.out.println("Tien luong = " + Luong);
}
}
```

II. Cấu trúc if ... else...

II.1. Cú pháp:

- Trường hợp 1: Nếu biểu thức điều kiện đúng chương trình sẽ thực hiện lệnh sau **if**, ngược lại chương trình sẽ thực hiện lệnh sau **else**.

if (Biểu thức điều kiện)

lệnh;

else

lệnh;

Ví dụ:

Ngôn ngữ C:

```
.....
.....
int age = 18;
if (age >= 18)
    printf("Ban da truong thanh");
else
    printf("Ban chua truong thanh");
.....
.....
```

Ngôn ngữ Java:

```
.....
.....
int age = 18;
if (age >= 18)
    System.out.print("Ban da truong thanh");
else
```

```
System.out.print("Ban chua truong thanh");
```

```
.....  
.....
```

- Trường hợp 2: Nếu biểu thức điều kiện đúng chương trình sẽ thực hiện các lệnh sau **if**, ngược lại chương trình sẽ thực hiện các lệnh sau **else**.

if (Biểu thức điều kiện)

```
{  
    lệnh 1;  
    lệnh 2;  
    ...  
    lệnh n;  
}  
else  
{  
    lệnh 1;  
    lệnh 2;  
    ...  
    lệnh n;  
}
```

Ví dụ:

Ngôn ngữ C:

```
.....  
.....  
int age = 18;  
if (age >= 18)  
{  
    printf("Ban da truong thanh\n");  
    printf("Nam nay ban %d tuoi", age);  
}  
else  
{  
    printf("Ban chua truong thanh");  
    printf("Nam nay ban moi %d tuoi", age);  
}  
.....  
.....
```

Ngôn ngữ Java:

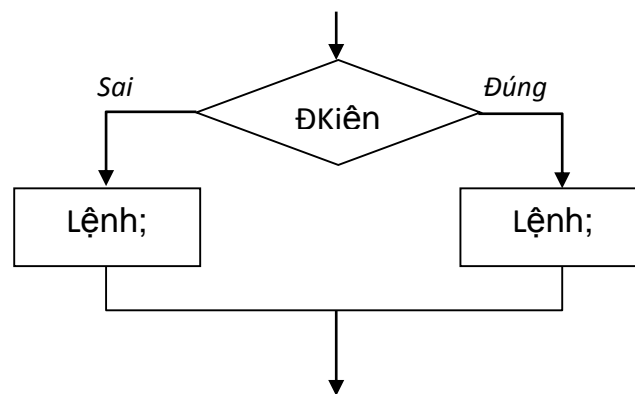
```

.....
.....
int age = 18;
if (age >= 18)
{
    System.out.print("Ban da truong thanh");
    System.out.print("Nam nay ban da "+age+" tuoi");
}
else
{
    System.out.print("Ban chua truong thanh");
    System.out.print("Nam nay ban moi "+age+" tuoi");
}
.....
.....

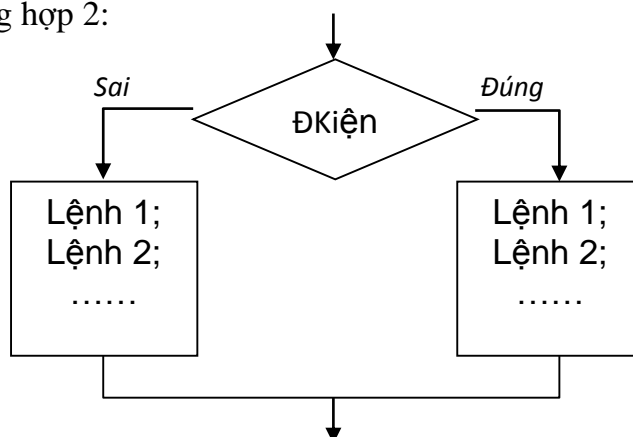
```

II.2. Lưu đồ:

Trường hợp 1:



Trường hợp 2:

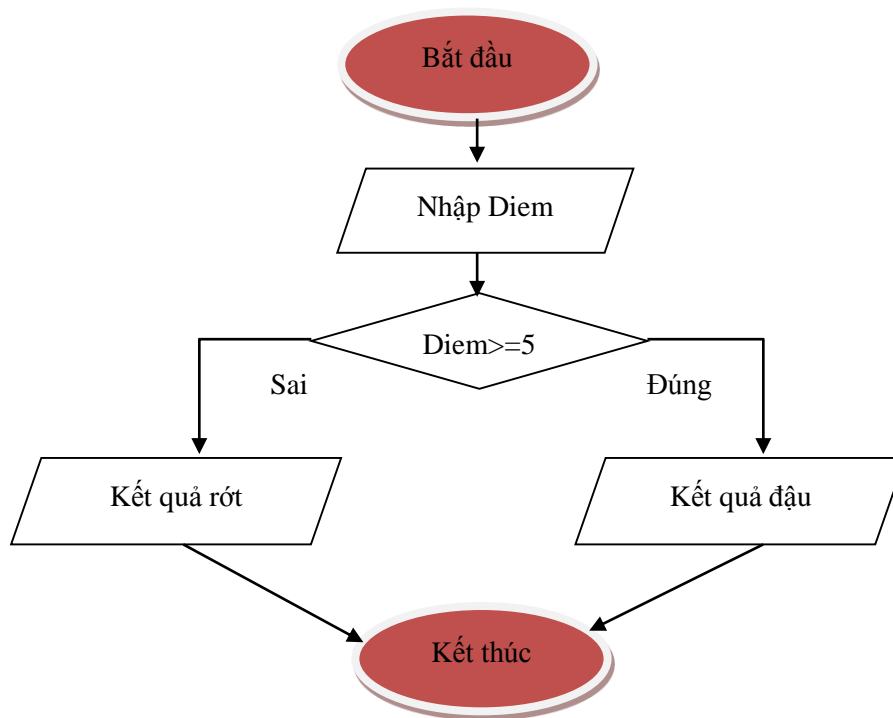


Hình 5.4: Lưu đồ cấu trúc điều kiện if...else...

II.3.Ví dụ minh họa:

- **Ví dụ:** Nhập điểm trung bình. Nếu điểm TB lớn hơn hoặc bằng 5 thì kết quả là đậu, ngược lại thì rớt.

//Lưu đồ giải thuật:



Hình 5.5: Lưu đồ giải thuật

//Mã giả

```
BEGIN
    INPUT Diem
    IF Diem >= 5 THEN
        DISPLAY Kết quả -đậu
    ELSE
        DISPLAY Kết quả rớt
    ENDIF
END
```

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
#include "conio.h"
void main()
{
    float Diem = 0;
    printf("Nhap Diem trung binh :");
    scanf("%f",&Diem);
    if (Diem>=5)
        printf("Ket qua Dau");
    else
        printf("Ket qua ROT");
    getch();
}
```

//Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
public class bt1
{
    public static void main(String [] arge)
    {
        float DiemTB;
        Scanner Input = new Scanner(System.in);
        System.out.print("Nhap diem TB:");
        DiemTB = Input.nextFloat();
        if (DiemTB>5)
            System.out.print("ket qua DAU");
        else
            System.out.printf("Ket qua ROT");
    }
}
```

III. Cấu trúc if ... else...lồng nhau

III.1. Cú pháp:

```
if (điều kiện 1)
    lệnh 1;
else if(điều kiện 2)
    lệnh 2;
```

```

        .
        .
        .
    else if(điều kiện n)
        lệnh n;
    else lệnh n+1;
    
```

III.2. Ví dụ minh họa:

Nhập điểm trung bình rồi xếp loại học tập của sinh viên. Biết rằng xếp loại học lực được tính như sau:

Điểm trung bình	Xếp loại
≥ 9 và ≤ 10	Xuất sắc
≥ 8 và < 9	Giỏi
≥ 7 và < 8	Khá
≥ 5 và < 7	Trung bình
≥ 0 và < 5	Yếu
< 0 hoặc > 10	Nhập điểm sai

//Mã giả

```

BEGIN
    INPUT Diem
    IF Diem < 0 OR Diem > 10 THEN
        DISPLAY Nhập điểm sai
    ELSE IF DIEM  $\geq 9$  THEN
        DISPLAY Xuất sắc
    ELSE IF DIEM  $\geq 8$  THEN
        DISPLAY Giỏi
    ELSE IF DIEM  $\geq 7$  THEN
    
```

```
        DISPLAY Khá
    ELSE IF DIEM >=5 THEN
        DISPLAY Trung bình
    ELSE
        DISPLAY Yếu
    ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
END
```

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
#include "conio.h"
void main()
{
    float Diem = 0;
    printf("Nhap Diem trung binh :");
    scanf("%f",&Diem);
    if (Diem<0 || Diem >10)
        printf("Nhap diem sai");
    else if (Diem>=9)
        printf("Xuat sac");
    else if (Diem>=8)
        printf("Gioi");
    else if (Diem>=7)
        printf("Kha");
    else if (Diem>=5)
        printf("Trunh binh");
    else printf("Yeu");
    getch();
}
```

//Chương trình viết bằng ngôn ngữ ^[a1] Java:

```
import java.io.*;
import java.util.Scanner;
public class ViduMinhhhoa
{
    public static void main(String [] arge)
```



```
{  
    float DiemTB;  
    Scanner Input = new Scanner(System.in);  
    System.out.print("Nhap diem TB:");  
    DiemTB = Input.nextFloat();  
    if (DiemTB<0 || DiemTB >10)  
        System.out.print("Nhap diem sai");  
    else if (DiemTB >=9)  
        System.out.printf("Xuat sac");  
    else if (DiemTB >=8)  
        System.out.printf("Gioi");  
    else if (DiemTB >=7)  
        System.out.printf("Kha");  
    else if (DiemTB >=5)  
        System.out.printf("Trung binh");  
    else  
        System.out.printf("Yeu");  
}
```

IV. Cấu trúc switch ... case

Dùng để thực hiện một quyết định rẽ nhánh khi thỏa một điều kiện trong nhiều điều kiện.

IV.1. Cú pháp:

```
switch (Biểu thức)  
{  
    case Constant_1:    lệnh 1; break;  
    case Constant_2:    lệnh 2; break;  
    .  
    .  
    .  
    case Constant_n:    lệnh n; break;  
    default:    lệnh n+1;  
}
```

- *Biểu thức* phải có kết quả là trị nguyên
- *default* là thành phần không bắt buộc.
- Khi *biểu thức* không thỏa điều kiện nào thì sẽ nhảy tới câu lệnh có nhãn default, nếu không có default thì sẽ thoát ra khỏi switch.
- Khi gặp câu lệnh break thì sẽ thoát ra khỏi thân switch.
- Nếu mỗi case không có break sẽ thực hiện câu lệnh case tiếp theo

IV.2.Ví dụ minh họa:

Nhập vào một số nguyên N (từ 1 đến 8), kiểm tra xem đó là ngày thứ mấy trong tuần. (Biết rằng nếu N=1: Chủ Nhật, N=2 :Thứ 2, N=3: Thứ 3,...).

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
#include "conio.h"
void main()
{
    int n;
    printf("Nhap mot so nguyen: ");
    scanf("%d",&n);
    switch (n)
    {
        case 1: printf("Chu Nhat"); break;
        case 2: printf("Thu hai"); break;
        case 3: printf("Thu ba"); break;
        case 4: printf("Thu tu"); break;
        case 5: printf("Thu nam"); break;
        case 6: printf("Thu sau"); break;
        case 7: printf("Thu bay"); break;
        default : printf("Nhap ngay sai");
    }
}
```

//Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
public class ViduMinhhhoa
{
    public static void main(String [] arge)
    {
        int Day;
        Scanner Input = new Scanner(System.in);
        System.out.print("Nhap ngay:");
        Day = Input.nextInt();
        switch (Day)
        {
            case 1: System.out.print("Chu Nhat"); break;
            case 2: System.out.print("Thu hai"); break;
            case 3: System.out.print("Thu ba"); break;
            case 4: System.out.print("Thu tu"); break;
            case 5: System.out.print("Thu nam"); break;
            case 6: System.out.print("Thu sau"); break;
        }
    }
}
```

```

        case 7: System.out.print("Thu bay"); break;
        default: ystem.out.print("Nhập sai ngay");
    }
}
}

```

V. Bài tập tự làm

Câu 1: Viết chương trình nhập một số nguyên, kiểm tra số đó có chia hết cho 3 hay không.

Câu 2: Viết chương trình giải phương trình bậc nhất ($ax+b=0$).

Câu 3: Viết chương trình giải phương trình bậc hai ($ax^2+bx+c=0$).

Câu 4: Viết chương trình nhập vào số KWh điện tiên thu và tính tiền điện phải trả theo công thức sau:

Từ KWh thứ 1 đến KWh 100: giá điện là 1000đ/KWh

Từ KWh thứ 101 đến KWh 150: giá điện là 1200đ/KWh

Từ KWh thứ 151 đến KWh 200: giá điện là 1600đ/KWh

Từ KWh 201 trở lên: giá điện là 2000đ/KWh.

Câu 5: Viết chương trình nhập số KM sau đó tính tiền cước Taxi. Biết rằng:

KM đầu tiên là 16.000đ.

Cứ 200m tiếp theo là 2.000đ.

Từ 31km trở lên thì mỗi km giá 14.000đ.

Câu 6: Viết chương trình nhập vào 4 số nguyên, tìm số nguyên lớn nhất và nhỏ nhất trong 4 số đó.

Câu 7: Viết chương trình nhập vào một số nguyên dương n với $1 \leq n \leq 7$. Tùy theo $n=1, 2, 3, \dots, 7$ hãy in tương ứng các từ Sunday, Monday, Tuesday, ..., Saturday ra màn hình.

Câu 8: Nhập vào 3 cạnh của một tam giác, hãy kiểm tra xem đó có phải là tam giác vuông không.

Câu 9: Nhập điểm toán, lý, hóa của một học sinh, hãy xếp loại học sinh đó dựa vào bảng sau:

Điểm trung bình 3 môn	Điều kiện khác	Xếp loại
≥ 9 và ≤ 10	Không có môn nào < 8	Xuất sắc
≥ 8 và < 9	Không có môn nào < 7	Giỏi

≥ 7 và < 8	Không có môn nào < 6	Khá
≥ 5 và < 7	Không có môn nào < 4	Trung bình
≥ 0 và < 5		Yếu
< 0 hoặc > 10		Nhập điểm sai

Câu 10: Hãy nhập vào năm sinh của một người. Hãy cho biết người đó tuổi gì? Ví dụ: người sinh năm 1973: tuổi quý sửu, sinh năm 1981: tuổi tân dậu.

Câu 11: Nhập 3 cạnh của một tam giác. Hãy kiểm tra tam giác đó có hợp lệ không? Nếu hợp lệ thì cho biết tam giác đó là tam giác gì?

Câu hỏi trắc nghiệm:

Câu 1: Hãy cho biết các câu lệnh sau sai ở chỗ nào?

```
if 5>7  
    printf("Condition Test");
```

a/ Thiếu dấu chấm phẩy “;” ở cuối hàng if

b/ Trong câu lệnh if thiếu mở ngoặc và đóng ngoặc

c/ Sai ở câu lệnh thứ 2

d/ Tất cả đều sai

Câu 2: Hãy cho biết câu lệnh sau sai ở chỗ nào?

```
IF (5>7) printf("Condition Test");
```

a/ Chữ if viết hoa

b/ Thiếu dấu chấm ngay sau if và trước printf(“...”)

c/ Không được phép viết if và printf cùng trên một dòng

d/ Không có chỗ nào sai

Câu 3: Kết quả của chương trình sau có kết quả là gì?

```
void main()
{
    if(1)
        printf("Good morning");
    else
        printf("Good afternoon");
}
```

a/ Good morning

b/ Good afternoon

c/ Good morning Good afternoon

d/ Good afternoon Good morning

Câu 4: `if(100=2*45) int n=1;`

Câu lệnh này có một chỗ sai, viết lại như thế nào cho đúng?

a/ `if(100=(2*45)) int n=1;`

b/ `if(100=2*45) int n==1;`

c/ `if(100=2*45); int n=1;`

d/ `if(100==2*45) int n=1;`

Câu 5: Đoạn chương trình sau có kết quả là?

```
void main()
{
    int N=10;
    switch(N%2)
    {
        case 0: printf("N la so chan!");
        case 1: printf("N la so le.");
    }
}
```

a/ N la so chan!

b/ N la so le.

c/ N la so chan!N la so le.

d/ Tất cả đều sai

Câu 6: Câu lệnh `break;` có ý nghĩa gì trong cấu trúc `switch case`?

a/ Tiếp tục thực hiện các câu lệnh tiếp theo

b/ Thoát khỏi cấu trúc switch case

c/ Thoát khỏi chương trình

d/ Tạm dừng chương trình

Câu 8: Đoạn chương trình sau có kết quả là?

```
void main()  
{  
    int N=100;  
    if (N%2==1)  
        N++;  
    printf("%d",N);  
}
```

a/ 100

b/ 101

c/ Không có kết quả in ra màn hình

d/ Là một kết quả khác

Câu 9: if (2==1+1) x =1; else x =0;

Câu lệnh trên tương đương với?

a/ x = (2==1+1) ? 0 : 1;

b/ x = (2==1+1) ? 1 : 0;

c/ x = (2==1+1) : 0 ? 1;

d/ x = (2==1+1) : 1 ? 0;

Câu 10: Để tìm số lớn nhất trong 3 số a, b, c, làm như sau:

a/ int Max = ((a>b)?a:b)>c? ((a>b)?a:b):c;

b/ int Max = ((a>b)?a:b)<c? ((a>b)?a:b):c;

c/ int Max = ((a<b)?a:b)>c? ((a>b)?a:b):c;

d/ int Max = ((a>b)?a:b)<c? ((a<b)?a:b):c;

BÀI 6: CẤU TRÚC VÒNG LẶP

I. Vòng lặp for

Cấu trúc lặp được dùng khi muốn một tác vụ nào đó được thực hiện lặp lại nhiều lần. Khi tác vụ lặp với số lần lặp xác định ta nên dùng vòng lặp for.

I.1. Cú pháp:

for(Initiation; Condition; Statement)

```
{  
    Statements;  
}
```

Trong đó:

- Initiation: Khởi tạo giá trị đếm cho vòng lặp
- Condition: Điều kiện lặp. Nếu điều kiện đúng thì vòng lặp thực thi.
- Statement: Tác vụ thực thi làm thay đổi biến đếm
- Statements: Các tác vụ cần thực hiện lặp

Ví dụ:

Viết chương trình xuất 10 chữ “Hello” ra màn hình

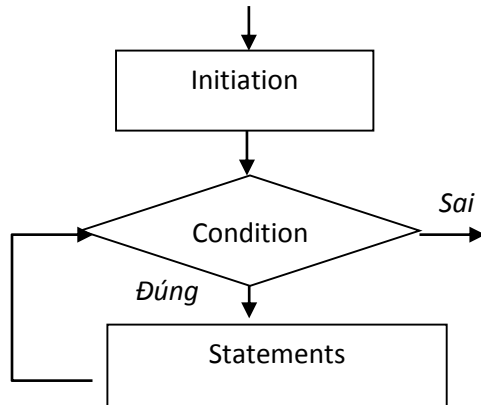
Ngôn ngữ C:

```
for(int i=1; i<=10; i++)  
{  
    printf("Hello "); // xuất 10 từ "Hello" ra màn hình  
}
```

Ngôn ngữ Java:

```
for(int i=1; i<=10; i++)  
{  
    System.out.println("Hello ");  
}
```

I.2. Lưu đồ:



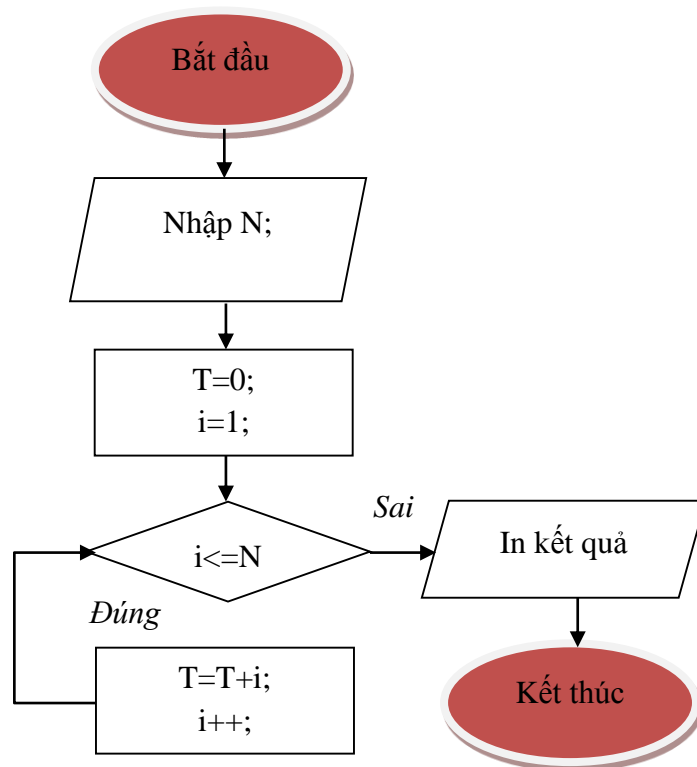
Hình 6.1: Lưu đồ vòng lặp for

I.3 Bài tập minh họa

Ví dụ 1:

Viết chương trình nhập vào số nguyên N. Tính Tổng biểu thức $T=1+2+3+\dots+N$ và xuất ra màn hình giá trị tổng.

➔ Lưu đồ giải thuật:



Hình 6.2: Lưu đồ giải thuật tính biểu thức $T=1+2+3+\dots+N$

//Mã giả:

```
BEGIN
    INPUT N;
    T = 0;
    FOR i=1 to N
        T=T+i;
        i++;
    END FOR
    DISPLAY T;
END.
```

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
#include "conio.h"
void main()
{
    int N, i, T=0;
    printf("Nhap so nguyen N :");
    scanf("%d",&N);

    for(i=1; i<=N; i++)
        T=T+i;

    printf("Tong la = %d",T);
    getch();
}
```

//Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
public class Vidu1
{
    public static void main(String [] arge)
    {
        int N, T=0, i;
        Scanner Input = new Scanner(System.in);
        System.out.print("Nhap so nguyen N :");
        N = Input.nextInt();

        for(i=1; i<=N; i++)
```

```
        T=T+i;

        System.out.println("Tong la = " + T);
    }
}
```

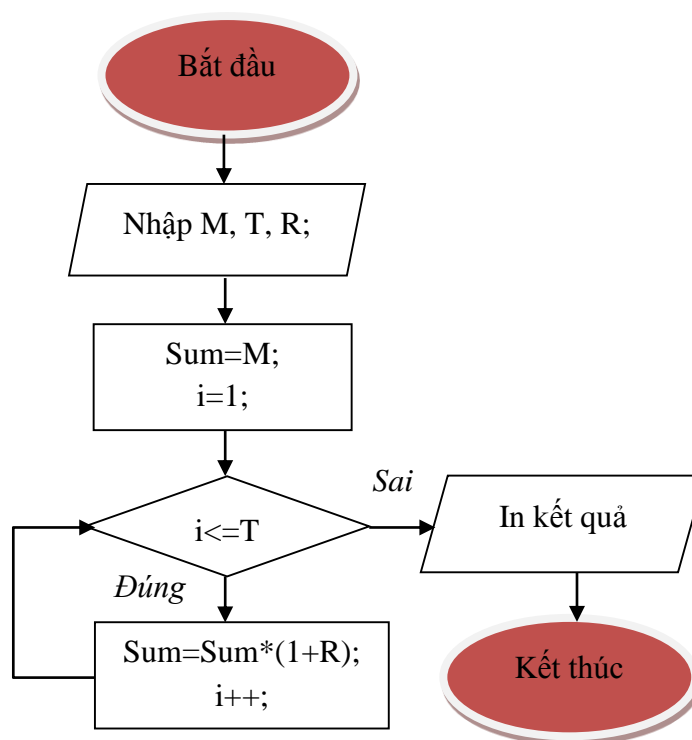
Ví dụ 2:

Viết chương trình nhập vào số tiền gửi tiết kiệm M, số tháng gửi tiết kiệm T, mức lãi suất theo tháng R. Tính tổng số tiền có được khi gửi ngân hàng T tháng. Biết rằng công thức tính tiền như sau:

$$\text{Sum} = M * (1+R) * (1+R) * \dots * (1+R) = M * (1+R)^T.$$

Lưu ý: Không dùng hàm lũy thừa có sẵn.

// Lưu đồ giải thuật:



Hình 6.5: Lưu đồ giải thuật chương trình tính lãi tiết kiệm

//Mã giả:

```
BEGIN
    INPUT M, T, R;
    Sum = M;
    FOR i=1 to T
        Sum = Sum * (1+R);
        i++;
    END FOR
    DISPLAY Sum;
END.
```

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
#include "conio.h"
void main()
{
    int M, T, i, Sum;
    float R;
    printf("Nhap so tien gui tiet kiem :");
    scanf("%d",&M);

    printf("Nhap so thang gui tiet kiem :");
    scanf("%d",&T);

    printf("Nhap lai suat ngan hang theo thang:");
    scanf("%f",&R);

    Sum = M;
    for(i=1; i<=T; i++)
        Sum=Sum*(1+R);

    printf("Tong la ban co duoc la = %d", Sum);
    getch();
}
```

// Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
public class Vidu1
{
```

```
public static void main(String [] arge)
{
    int M, T, Sum, i;
    float R;
    Scanner Input = new Scanner(System.in);
    System.out.print("Nhap so tien goi tiet kiem :");
    M = Input.nextInt();

    System.out.print("Nhap so thang goi TK :");
    T = Input.nextInt();

    System.out.print("Nhap lai suat ngan hang :");
    R = Input.nextFloat();

    Sum = M;
    for(i=1; i<=T; i++)
        Sum = Sum*(1+R);

    System.out.println("Tong tien co duoc = "+Sum);
}
```

II. Vòng lặp while

Cũng giống như vòng lặp for. Tuy nhiên thường dùng khi thực hiện tác vụ lặp lại có số lần lặp không xác định.

II.1. Cú pháp:

```
Initiation;
while(Condition)
{
    Statements;
}
```

Trong đó:

Initiation: Khởi tạo giá trị đếm cho vòng lặp

Condition: Điều kiện lặp. Nếu điều kiện đúng thì vòng lặp thực thi.

Statements: Các tác vụ cần thực hiện lặp. Trong statements phải làm thay đổi giá trị so sánh điều kiện.

Ví dụ:

Viết chương trình in 10 chữ Hello ra màn hình

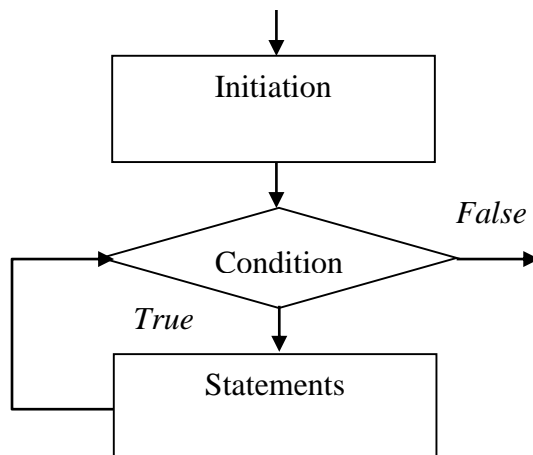
//Ngôn ngữ C:

```
int i=1;
while( i<=10)
{
    printf("Hello "); // xuất 10 từ "Hello" ra màn hình
    i++;
}
```

// Ngôn ngữ Java:

```
int i=1;
while(i<=10)
{
    System.out.println("Hello ");
    i++; //Thay đổi giá trị biến đếm
}
```

II.2 Lưu đồ

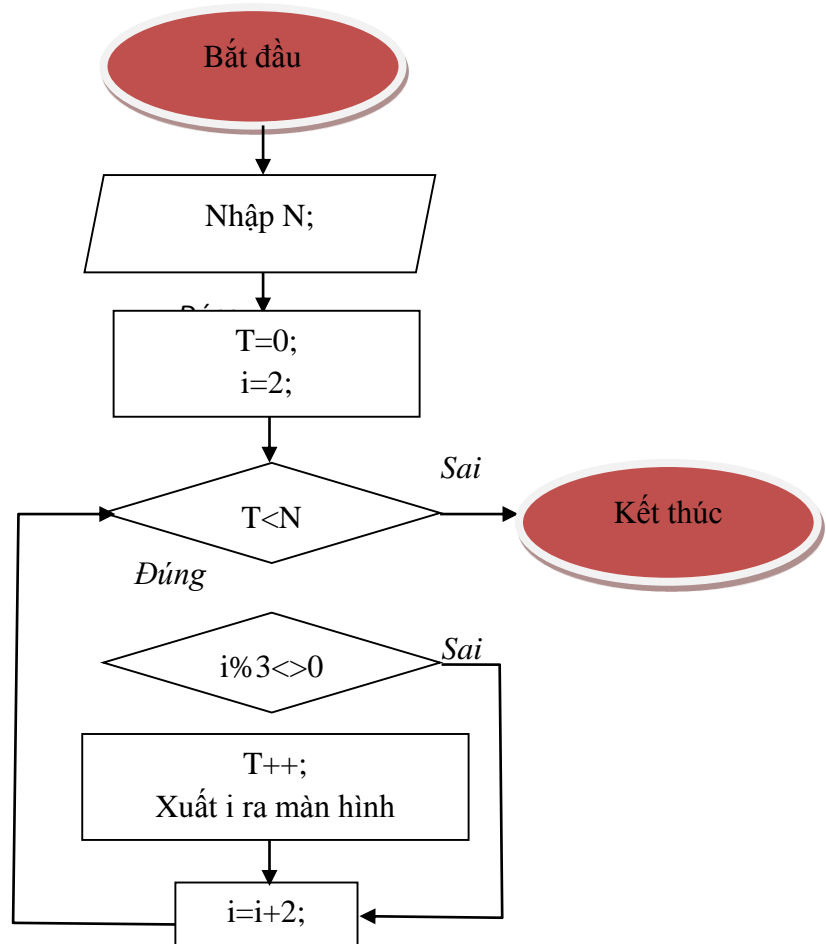


Hình 6.8: Lưu đồ vòng lặp while

II.3 Bài tập minh họa

Ví dụ 1: Viết chương trình nhập vào số nguyên N. Xuất ra N số nguyên đầu tiên là số chẵn mà không chia hết cho 3. (Ví dụ: N =5. Có các số là 2, 4, 8, 10, 14)

// Lưu đồ giải thuật:



Hình 6.9: Lưu đồ giải thuật chương trình xuất N số nguyên đầu tiên là số chẵn và không chia hết cho 3.

//Mã giả:

```
BEGIN
  INPUT N;
  T = 0; i=2;
  WHILE T<N
    IF i%3<>0 THEN
      Output i;
      T=T+1;
    ENDIF
  ENDIF
```

```
        i=i+2;  
    ENDWHILE
```

```
END.
```

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"  
#include "conio.h"  
void main()  
{  
    int N, i=2, T=0;  
    printf("Nhap so nguyen N :");  
    scanf("%d",&N);  
  
    while (T<N)  
    {  
        if (i%3!=0)  
        {  
            printf("%3d", i);  
            T++;  
        }  
        i=i+2;  
    }  
    getch();  
}
```

//Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;  
import java.util.Scanner;  
public class Vidu1  
{  
    public static void main(String [] arge)  
    {  
        int N, T=0, i;  
        Scanner Input = new Scanner(System.in);  
        System.out.print("Nhap so nguyen N :");  
        N = Input.nextInt();  
  
        while (T<N)  
        {  
            if (i%3!=0)  
            {  
                System.out.print(" " + i);  
                T=T+1;  
            }  
        }  
    }  
}
```

```

        i=i+2;
    }

}

```

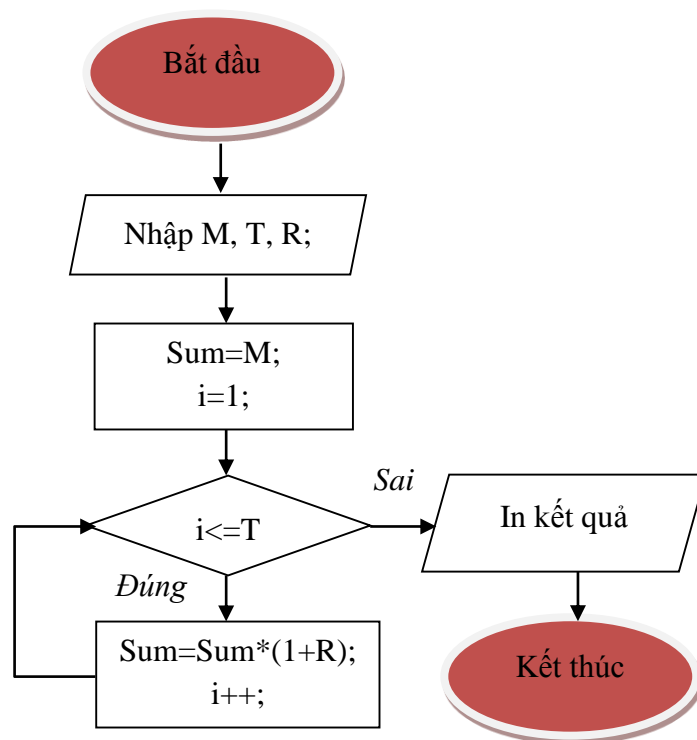
Ví dụ 2:

Viết chương trình nhập vào số tiền gửi tiết kiệm M, số tháng gửi tiết kiệm T, mức lãi suất theo tháng R. Tính tổng số tiền có được khi gửi ngân hàng T tháng. Biết rằng công thức tính tiền như sau:

$$\text{Sum} = M * (1+R) * (1+R) * \dots * (1+R) = M * (1+R)^T.$$

Lưu ý: Không dùng hàm lũy thừa có sẵn.

// Lưu đồ giải thuật:



Hình 6.12: Lưu đồ giải thuật chương trình tính tiền tiết kiệm

//Mã giả:

```

BEGIN
    INPUT M, T, R;
    Sum = M; i=1;
    WHILE i <= T

```



```
        Sum = Sum * (1+R);
        i++;
    END WHILE
    DISPLAY Sum;
END.
```

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
#include "conio.h"
void main()
{
    int M, T, i, Sum;
    float R;
    printf("Nhap so tien gui tiet kiem :");
    scanf("%d",&M);

    printf("Nhap so thang gui tiet kiem :");
    scanf("%d",&T);

    printf("Nhap lai suat ngan hang theo thang:");
    scanf("%f",&R);

    Sum = M;
    i=1;
    while(i<=T)
    {
        Sum=Sum*(1+R);
        i++;
    }

    printf("Tong la ban co duoc la = %d", Sum);
    getch();
}
```

//Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
public class Vidul
{
    public static void main(String [] arge)
    {
```

```
int M, T, Sum, i;
float R;
Scanner Input = new Scanner(System.in);
System.out.print("Nhap so tien goi tiet kiem :");
M = Input.nextInt();

System.out.print("Nhap so thang goi TK :");
T = Input.nextInt();

System.out.print("Nhap lai suat ngan hang :");
R = Input.nextFloat();
Sum = M;
i=1;
while(i<=T)
{
    Sum = Sum*(1+R);
    i=i+1;
}

System.out.println("Tong tien co duoc = "+Sum);
}
```

III. Vòng lặp do - while

Được dùng khi thực hiện tác vụ lặp lại có số lần lặp không xác định.

III.1. Cú pháp:

```
Initiation;
do{
    Statements;
} while(Condition);
```

Trong đó:

Initiation: Khởi tạo giá trị đếm cho vòng lặp

Condition: Điều kiện lặp. Nếu điều kiện đúng thì vòng lặp thực thi.

Statements: Các tác vụ cần thực hiện lặp. Trong statements phải làm thay đổi giá trị so sánh điều kiện.

Ví dụ:

//Ngôn ngữ C:

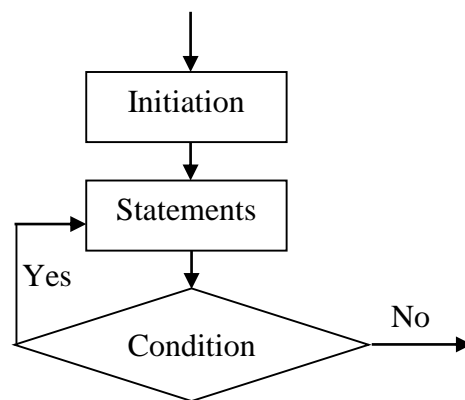
```
int i=1;
do{
```

```
printf("Hello "); // xuất 10 từ "Hello" ra màn hình
i++;
} while( i<=10);
```

//Ngôn ngữ Java:

```
int i=1;
do{
    System.out.println("Hello ");
    i++;
} while(i<=10);
```

III.2. Lưu đồ



Hình 6.15: Lưu đồ vòng lặp do-while

III.3 Bài tập minh họa

Ví dụ 1: Viết chương trình nhập vào các số nguyên dương. Nhập vào số không dương thì dừng. Tính tổng các số nguyên dương vừa nhập.

//Mã giả:

```
BEGIN
    T = 0;
    DO
        INPUT X;
        IF X>0 THEN T=T+X;
        ENDIF
    WHILE N>0
    OUTPUT T
END.
```

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
#include "conio.h"
void main()
{
    int X, T=0;

    do{
        printf("Nhap so nguyen X :");
        scanf("%d",&X);

        if(X>0) T=T+X;
    }while(X>0);
    printf("Tong cac so nguyen duong %d :", T);
    getch();
}
```

//Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
public class Vidu1
{
    public static void main(String [] arge)
    {
        int X, T=0;
        Scanner Input = new Scanner(System.in);

        do{
            System.out.print("Nhap so nguyen X :");
            X = Input.nextInt();

            if(X>=0) T=T+X;
        }while(X>0);
        System.out.println("Tong cac so nguyen duong "+T);
    }
}
```

Ví dụ 2: Viết chương trình nhập vào số tiền gửi tiết kiệm M, số tháng gửi tiết kiệm T, mức lãi suất theo tháng R. Tính tổng số tiền có được khi gửi ngân hàng T tháng. Biết rằng công thức tính tiền như sau:

$$\text{Sum} = M * (1+R) * (1+R) * \dots * (1+R) = M * (1+R)^T.$$

Lưu ý: Không dùng hàm lũy thừa có sẵn.

//Mã giả:

```
BEGIN
    INPUT M, T, R;
    Sum = M; i=1;
    DO
        Sum = Sum * (1+R);
        i++;
    WHILE i<=T
    DISPLAY Sum;
END.
```

//Chương trình viết bằng ngôn ngữ C:

```
#include "stdio.h"
#include "conio.h"
void main()
{
    int M, T, i, Sum;
    float R;
    printf("Nhap so tien gui tiet kiem :");
    scanf("%d",&M);

    printf("Nhap so thang gui tiet kiem :");
    scanf("%d",&T);

    printf("Nhap lai suat ngan hang theo thang:");
    scanf("%f",&R);

    Sum = M;
    i=1;
    do{
        Sum=Sum*(1+R);
        i++;
    }while(i<=T);

    printf("Tong la ban co duoc la = %d", Sum);
    getch();
}
```

// Chương trình viết bằng ngôn ngữ Java:

```
import java.io.*;
import java.util.Scanner;
public class Vidu1
{
    public static void main(String [] arge)
    {
        int M, T, Sum, i;
        float R;
        Scanner Input = new Scanner(System.in);
        System.out.print("Nhap so tien goi tiet kiem :");
        M = Input.nextInt();

        System.out.print("Nhap so thang goi TK :");
        T = Input.nextInt();
        System.out.print("Nhap lai suat ngan hang :");
        R = Input.nextFloat();

        Sum = M;
        i=1;
        do{
            Sum = Sum*(1+R);
            i=i+1;
        }while(i<=T);
        System.out.println("Tong tien co duoc="+Sum);
    }
}
```

IV. Lệnh break – continue

1. Lệnh break:

– Câu lệnh cho phép thoát khỏi một vòng lặp for, while, do-while và thoát khỏi lệnh switch. Khi có nhiều vòng lặp lồng nhau, muốn thoát khỏi vòng lặp nào thì dùng lệnh break trong vòng lặp đó.

Ví dụ 1:

```
int kq=0;
for(int i=1; i<100; i++)
{
    if(i>5) break;
    kq = kq+i;
}
```

Trong ví dụ trên, kq sẽ có giá trị là $1+2+3+4+5 = 15$. Vì khi $i=6$, lệnh break được thực thi, chương trình thoát khỏi vòng lặp.

Ví dụ 2:

```
for(int i=1; i<=3; i++)
    for(int j=1; j<=3; j++)
    {
        if(i<j) break;
        printf("(%3d,%3d) ", i, j);
    }
```

Trong ví dụ trên, chương trình sẽ in ra màn hình (1,1) (2,1) (2,2) (3,1) (3,2) (3,3). Vì khi $i=1, j=2$ lệnh break được thực thi, chương trình thoát khỏi vòng lặp bên trong. i tăng lên 2, vòng lặp bên trong lại tiếp tục thực thi, khi $i=2, j=3$ chương trình thoát khỏi vòng lặp bên trong. i tăng lên 3 thì vòng lặp bên trong thực thi cho tới khi $j=3$.

2. Lệnh continue:

– Câu lệnh cho phép bỏ qua một lần lặp trong vòng lặp for, while, do-while.

Ví dụ 1:

```
int kq=0;
for(int i=1; i<100; i++)
{
    if(i>5) continue;
    kq = kq+i;
}
```

Trong ví dụ trên, kq sẽ có giá trị là $1+2+3+4+5 = 15$. Vì khi $i=6$ trở đi, lệnh continue được thực thi, chương trình vẫn chạy cho tới khi $i=99$ nhưng bỏ qua lệnh $kq=kq+i$.

Ví dụ 2:

```
for(int i=1; i<=3; i++)
    for(int j=1; j<=3; j++)
    {
        if(i<j) continue;
        printf("(%3d,%3d) ", i, j);
    }
```

Trong ví dụ trên, chương trình sẽ in ra màn hình (1,1) (2,1) (2,2) (3,1) (3,2) (3,3). Vì khi $i=1, j=2$ lệnh continue được thực thi, chương trình bỏ qua việc xuất i, j . Trong ví dụ này 2 vòng lặp sẽ thực thi kiểm tra điều kiện 9 lần nhưng việc xuất i, j được thực hiện 6 lần.

V. Bài tập tự làm

- Bài 1: Viết chương trình nhập vào số nguyên dương N , xuất ra màn hình các số nguyên dương lẻ nhỏ hơn N sao cho cứ 5 số thì nằm trên cùng 1 dòng.
- Bài 2: Viết chương trình nhập vào số nguyên dương N , xuất ra màn hình tổng các số nguyên dương lẻ nhỏ hơn hoặc bằng N .
- Bài 3: Viết chương trình nhập vào số nguyên dương N , xuất ra màn hình các ước số của N mà nhỏ hơn N .
- Bài 4: Viết chương trình nhập vào số nguyên dương N , kiểm tra xem N có phải là số hoàn thiện không. (Số hoàn thiện là số có tổng các ước số nhỏ hơn nó bằng chính nó. Ví dụ 6 có các ước số nhỏ hơn nó là 1, 2, 3. Trong đó $6 = 1 + 2 + 3$, như vậy 6 là số hoàn thiện)
- Bài 5: Viết chương trình in bảng cửu chương.
- Bài 6: Viết chương trình nhập 2 số nguyên N, M tìm M^N
- Bài 7: Tính $P = n!$ với n nguyên dương nhập từ bàn phím
Ví dụ: nhập $x = 5$ thì $P = 5! = 120$
- Bài 8: Viết chương trình in các kí tự từ 'A' đến 'Z' xuôi và ngược, chữ hoa và chữ thường.
- Bài 9: Viết chương trình nhập vào số nguyên dương N , kiểm tra xem N có phải là số nguyên tố không?
- Bài 10: Viết chương trình nhập vào số nguyên dương N , xuất ra màn hình N số nguyên tố đầu tiên. (Ví dụ $N=5$: xuất ra màn hình 2 3 5 7 9)
- Bài 11: Viết chương trình nhập liên tục các số nguyên dương cho đến khi nhập số nhỏ hơn hoặc bằng 0 thì dừng. Tính tổng các số vừa nhập.
- Bài 12: Viết chương trình nhập liên tục các số nguyên dương cho đến khi nhập số nhỏ hơn hoặc bằng 0 thì dừng. Tính trung bình các số vừa nhập.
- Bài 13: Viết chương trình nhập liên tục các số nguyên dương cho đến khi nhập số nhỏ hơn hoặc bằng 0 thì dừng. Tính trung bình các số lẻ/chẵn vừa nhập.
- Bài 14: Viết chương trình nhập vào 2 số nguyên dương x, y . Tính và xuất ra ước số chung lớn nhất và bội số chung nhỏ nhất của x, y .
- Bài 15: Dãy số Fibonacci có 2 phần tử đầu tiên là 1, các phần tử còn lại bằng tổng 2 phần tử đứng trước đó. (1 1 2 3 5 8 13 21). Hãy viết chương trình

nhập vào số nguyên dương N. Xuất ra màn hình dãy các số Fibonacci trong đó số lớn nhất của dãy nhỏ hơn hoặc bằng N.

Bài 16: Viết chương trình nhập số nguyên N tìm T

$$T = 1 + 3 + 5 + \dots + N$$

$$T = 1 - 2 + 3 - 4 + \dots + (-1)^{N+1} N$$

Gợi ý: thêm biến dau, khởi tạo dau = 1, sau mỗi lần lặp, dau=-dau

$$T = 1 - 1/2 + 1/3 - 1/4 + \dots 1/N$$

$$T = 2 + 4 + 6 + \dots N-1$$

$$T = 1 + 2^2 + 3^2 + \dots + N^2$$

Bài 17: Viết chương trình nhập số nguyên N và x tìm tổng T

$$T = x + x/2 + x/3 + x/4 + \dots x/N$$

$$T = 1/X + 2!/X^2 + 3!/X^3 + \dots + N!/X^N$$

Bài 18: Viết chương trình nhập N. xuất ra các hình có dạng sau:

Giả sử N = 4

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

Câu hỏi trắc nghiệm

Câu 1. Thực hiện đoạn chương trình sau kết quả là

```
for(int i=1; i<=5; i++)
{
    if(i%2) continue;
    printf("%4d", i);
}
```

a/ 1 2 3 4 5

b/ 1 3 5

c/ 2 4 6

d/ 2 4

Câu 2. Thực hiện đoạn chương trình sau kết quả là

```
for(int i=1; i<=5; i++)
{
    if(i==3) break;
```

```
        printf("%4d", i);  
    }  
a/ 1   2   3   4   5  
b/ 1   2   4   5  
c/ 1   2   3  
d/ 1   2
```

Câu 3. Thực hiện đoạn chương trình sau kết quả là

```
int kq=0;  
for(int i=1; i<=5; i++)  
    for(int j=1; j<=5; j++)  
        kq++;  
printf("%d", kq);
```

```
a/ 5  
b/ 10  
c/ 20  
d/ 25
```

Câu 4. Thực hiện đoạn chương trình sau kết quả là

```
int kq=0;  
for(int i=1; i<=5; i++)  
    for(int j=1; j<=5; j++)  
    {  
        if(i>j) break;  
        kq++;  
    }  
printf("%d", kq);
```

```
a/ 5  
b/ 15  
c/ 20  
d/ 25
```

Câu 5. Thực hiện đoạn chương trình sau kết quả là

```
int kq=0;  
for(int i=1; i<=5; i++)  
    for(int j=1; j<=5; j++)  
    {  
        if(i==j) continue;  
        kq++;  
    }  
printf("%d", kq);
```

- a/ 5
- b/ 10
- c/ 20**
- d/ 25

Câu 6. Thực hiện đoạn chương trình sau kết quả là

```
int k = 5;
while(k)
{
    printf("%4d", k); k--;
}
```

a/ 5 4 3 2 1

b/ 5 4 3 2 1 0

c/ 5 4 3 2

d/ Chương trình biên dịch lỗi

Câu 7. Thực hiện đoạn chương trình sau kết quả là

```
int k = 5;
while(k)
{
    printf("%4d", k--);
}
```

a/ 5 4 3 2 1

b/ 5 4 3 2 1 0

c/ 4 3 2 1 0

d/ Chương trình biên dịch lỗi

Câu 8. Thực hiện đoạn chương trình sau kết quả là

```
int k = 5;
while(k++)
{
    printf("%4d", k--);
}
```

a/ 5 4 3 2 1

b/ 4 3 2 1 0

c/ Chương trình chạy mãi mãi

d/ Chương trình biên dịch lỗi

Câu 9. Thực hiện đoạn chương trình sau kết quả là

```
int k = 5;
while(1)
{
    if(k==3) break;
    printf("%4d", --k);
}
```

a/ 5 4

b/ 4 3

c/ Chương trình chạy mãi mãi

d/ Chương trình biên dịch lỗi

Câu 10. Thực hiện đoạn chương trình sau kết quả là

```
int k = 5;
while(k--) printf("%4d", k);
```

a/ 5 4 3 2 1

b/ 5 4 3 2 1 0

c/ 4 3 2 1 0

d/ Chương trình biên dịch lỗi

Câu 11. Thực hiện đoạn chương trình sau kết quả là

```
int k = 5;
while(--k) printf("%4d", k);
```

a/ 5 4 3 2 1

b/ 4 3 2 1

c/ 5 4 3 2

d/ Chương trình biên dịch lỗi

Câu 12. Thực hiện đoạn chương trình sau kết quả là

```
int k = 5;
do{
    printf("%4d", k);
}while(--k);
```

a/ 4 3 2 1

b/ 5 4 3 2 1

c/ 5 4 3 2 1 0

d/ Chương trình biên dịch lỗi

BÀI 7: HÀM VÀ CẤU TRÚC CHƯƠNG TRÌNH

I. Hàm

I.1. Khái niệm:

Khi bài toán càng lớn thì độ phức tạp của nó cũng tăng theo. Để giảm bớt độ phức tạp, chúng ta có thể chia nhỏ bài toán thành từng phần, mỗi phần thực hiện một nhiệm vụ. Mỗi phần tương ứng như vậy chúng ta xây dựng một hàm.

Xem ví dụ sau:

Viết chương trình tính biểu thức:

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

Để viết chương trình này chúng ta thấy khá phức tạp. Phải tính từng tử số, tính từng mẫu số, rồi tính tổng Z.

Để đơn giản, chúng ta chia nhỏ bài toán này từng phần và mỗi phần tương ứng một hàm như sau:

Hàm Tong(i): tính tổng 1+2+...+... (tử số)

Hàm Giaithua(i): để tính giai thừa của i (mẫu số)

Lúc này chương trình tính biểu thức trên có dạng như sau:

$$Z = 1 + \frac{Tong(2)}{Giaithua(2)} + \frac{Tong(3)}{Giaithua(3)} + \dots + \frac{Tong(n)}{Giaithua(n)}$$

Lúc này để tính biểu thức Z ta chỉ cần chạy 1 vòng lặp là đủ.

For i=1 to N

Z = Z + Tong(i)/Giaithua(i)

➔ Một hàm có thể được sử dụng nhiều lần và cũng có thể đặt ở nhiều vị trí khác nhau trong chương trình. Có những tính toán hoặc 1 công việc thường được lặp đi lặp lại nhiều lần trong chương trình thì chúng ta nên tạo một hàm riêng cho nó.

➔ Việc dùng các hàm sẽ giúp chúng ta dễ dàng giải quyết bài toán phức tạp, cũng như phát hiện và sửa các lỗi trong chương trình dễ dàng hơn.

I.2. Xây dựng hàm:

Cú pháp xây dựng hàm:

[Kiểu DL] Tên hàm ([Danh sách tham số hình thức])

```
{    [Khai báo biến nội bộ]
    Lệnh 1;
    Lệnh 2;
    ...
    Lệnh n;
    [return [biểu thức]];
}
```

Trong đó:

- **Kiểu DL:** Nếu hàm phải trả về một giá trị thì phải khai báo kiểu dữ liệu trả về. Nếu hàm không trả về giá trị thì dùng kiểu void.
- **Danh sách tham số:** đây là tham số hình thức hay còn gọi là biến hình thức. Những tham số này sẽ nhận giá trị thực bằng cách truyền tham số mỗi khi hàm được gọi. Các tham số này cũng khai báo kiểu dữ liệu như khai báo thông thường.
- **Thân hàm:** được giới hạn bởi cặp dấu móc nhọn {...}

Ví dụ 1:

Viết hàm Tính tổng $1+2+3+...+N$.

Phân tích và viết hàm:

Kiểu dữ liệu trả về: **int**

Tên hàm: **Tong**

Tham số: có 1 tham số kiểu nguyên là **N**

Thân hàm: sử dụng vòng lặp **for** và **cộng dồn**

```
int Tong(int N) //Khai báo 1 tham số N kiểu nguyên
{
    int T=0; //Khai báo và khởi tạo biến T =0;
    for (int i=1; i<=N; i++) //i chạy từ 1 đến N
        T+=i; //Cộng dồn
    return T; //trả về giá trị T
}
```

Ví dụ 2: Viết hàm in N hàng chữ “Đại học Hoa Sen” ra màn hình:

Phân tích và viết hàm:

Kiểu dữ liệu trả về: không có giá trị trả về nên sử dụng kiểu **void**

Tên hàm: **HoaSen**

Tham số: có 1 tham số kiểu nguyên là **N**

Thân hàm: sử dụng vòng lặp for, mỗi vòng lặp in hàng chữ **Dai Hoa Hoa Sen** ra màn hình

```
void HoaSen(int N) // Khai báo 1 tham số N kiểu nguyên
{
    for (int i=1; i<=N; i++) // Lặp N lần
        printf("Dai Hoc Hoa Sen\n");
}
```

I.3. Sử dụng hàm:

Sau khi đã xây dựng xong hàm, ta có thể sử dụng nó bằng cách gọi các hàm ở một vị trí nào đó trong chương trình.

Cú pháp gọi hàm:

Tên hàm ([Các tham số thực])

- Tên hàm: tên hàm phải đúng như tên hàm đã xây dựng (lưu ý có phân biệt chữ hoa chữ thường).
- Các tham số thực: Các tham số thực là các tham số truyền vào để hàm thực hiện. Lưu ý khi xây dựng hàm có bao nhiêu tham số thì khi gọi phải truyền đúng từng ấy tham số.

Ví dụ:

Viết chương trình tính tổng $S = 1+2+3+...+N$. Sau đó in N hàng chữ “Đại học Hoa Sen” ra màn hình.

```
void main()
{
    int N, S = 0;
    printf("Nhap so nguyen N:");
    scanf("%d",&N);
    S = Tong(N); //Gọi hàm tính tổng đã làm ở ví dụ 1
    printf("Tong cua day so = %d",S); in S ra màn hình
    HoaSen(N); // Gọi hàm in hàng chữ Đại học Hoa Sen
}
```

I.4. Truyền tham số:

- Khi sử dụng hàm cần phải truyền đúng số lượng tham số, mỗi tham số tương ứng phải đúng như kiểu dữ liệu đã khai báo khi xây dựng hàm.
- Các tham số được truyền vào này gọi là tham số thực của hàm.
- Có 2 cách truyền tham số:
 - + **Truyền bằng trị**: khi tham số hình thức thay đổi thì **tham số thực không thay đổi**.
 - + **Truyền bằng biến**: Khi tham số hình thức thay đổi thì **tham số thực cũng thay đổi theo**. Khi muốn truyền tham số bằng tham biến thì đặt dấu **&** trước tham số hình thức.

Ví dụ: xem đoạn code sau:

```
#include <stdio.h>
#include <conio.h>
void Change(int a, int &b) {           //a sẽ được truyền bằng trị
    //b sẽ được truyền bằng biến

    a = a+2;
    b = b+2;
}
void main()
{
    int x=10, y=12;
    Change(x,y); //truyền tham số x và tham số y vào
    printf("x = %d, y = %d",x,y); //kết quả x=10,y=14
}
```

- Sau khi gọi hàm Change(x,y), biến x vẫn giữ nguyên giá trị là 10, mặc dù trong hàm Change chúng ta có tăng biến a lên 2. Còn biến y thì thay đổi từ 12 thành 14, vì khi truyền bằng biến mọi thay đổi của tham số hình thức trong thân hàm thì biến truyền vào (tham số thực) sẽ thay đổi theo.

- Lưu ý: khi truyền tham số cần chú ý: nếu truyền bằng tham trị thì tham số truyền vào có thể là biến hoặc hằng, còn nếu truyền bằng tham biến thì tham số thực truyền vào phải là biến, không chấp nhận hằng. Ví dụ gọi hàm Change(x,10) → sẽ báo lỗi. Còn gọi hàm Change(10,y) thì chấp nhận.

I.5. Hàm trả về giá trị và hàm không trả về giá trị:

- Hàm trả về giá trị là hàm có khai báo kiểu dữ liệu trả về và trong thân hàm phải có lệnh **return giá_trị_trả_về**.

Ví dụ:


```
long Tinhgiaithua(int N)//Kiểu dữ liệu trả về là long
{
    long Giaithua=1;
    for (int i=1; i<=N; i++)
        Giaithua = Giaithua * i;
    return Giaithua; //Trả về giá trị của giai thừa
}
```

- **Hàm không trả về giá trị ta dùng kiểu void**. Hàm này thường dùng để thực hiện một nhiệm vụ gì đó cho chương trình.

Ví dụ:

Hàm viết bảng cửu chương N như sau:

```
void BangCuuChuong(int N)    //Kiểu dữ liệu void
{
    for (int i=1; i<=10; i++)
        printf("%d * %d = %d\n", N, I, N*i);
}
```

I.5. Khai báo hàm nguyên mẫu:

- Tất cả các hàm có trong chương trình, chúng ta nên khai báo trước khi xây dựng hàm, trừ hàm main(). Vị trí và cách khai báo như sau:

Khai báo thư viện

Khai báo hàm nguyên mẫu

Hàm main()

Xây dựng hàm

Ví dụ: Viết chương trình tính biểu thức:

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

Phân tích bài toán:

- Để tính biểu thức này chúng ta viết hàm tính tử và hàm tính mẫu như sau:

Hàm Tong(i): tính tổng 1+2+...+... (tử số)

Hàm Giaithua(i): để tính giai thừa của i (mẫu số)

- Lúc này chương trình tính biểu thức trên có dạng như sau:

$$Z = 1 + \frac{Tong(2)}{Giaithua(2)} + \frac{Tong(3)}{Giaithua(3)} + \dots + \frac{Tong(n)}{Giaithua(n)}$$

- Lúc này để tính biểu thức Z ta chỉ cần chạy 1 vòng lặp là đủ.

FOR i=1 TO N

Z = Z + Tong(i)/Giaithua(i)

//Mã code bằng C:

```
//Khai báo thư viện
#include "stdio.h"
#include "conio.h"
//Khai báo các hàm nguyên mẫu
int Tong(int i);
long Giaithua(int i);
//Viết hàm main()
void main()
{
    int N = 10;
    float Z=0;
    for (int i=1; i<=N; i++)
        Z = Z + (float) Tong(i)/Giaithua(i);
    printf("Tong Z = %.2f",Z);
    getch();
}
//Xây dựng các hàm đã khai báo nguyên mẫu ở trên
int Tong(int i) // xây dựng hàm tính tổng
{
    int T =0, j=1;
    while (j<=i)
        T += j++;
    return T;
}

long Giaithua(int i) // xây dựng hàm tính giai thừa
{
    long GT=1; // cùng kiểu dữ liệu với kiểu dữ liệu trả về
    for (int j=1; j<=i; j++)
        GT= GT* j;
    return GT;
}
```

I.6. Bài tập tự làm

Câu 1: Viết hàm tìm số lớn nhất trong 3 số

Câu 2: Viết hàm tìm số lớn nhất trong 4 số

Câu 3: Viết hàm kiểm tra số nguyên n có phải là số nguyên tố không?

Câu 4: Viết hàm tìm USCLN của 2 số nguyên

Câu 5: Viết hàm tính n!

Câu 6: Viết hàm in n số fibonacci đầu tiên

Câu 7: Viết các hàm tính:

$$T1 = 1 + 2 + 3 + 4 + \dots + n$$

$$T2 = 1 - 2 + 3 - 4 + \dots + (-1)^n$$

$$T3 = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n}$$

$$T4 = 1 + \frac{1}{(1+2)} + \frac{1}{(1+2+3)} + \dots + \frac{1}{(1+2+\dots+n)}$$

$$F1 = 1 + \frac{2^2}{\sqrt{2}} + \frac{3^3}{\sqrt[3]{3}} + \dots + \frac{n^n}{\sqrt[n]{n}}$$

$$F2 = 1 + \frac{2!}{(1+2)} + \frac{3!}{(1+2+3)} + \dots + \frac{n!}{(1+2+\dots+n)}$$

Câu 9: Viết hàm in tất cả những số nguyên tố nhỏ hơn hoặc bằng n (n là số nguyên dương nhập từ bàn phím).

Câu 10: Viết hàm kiểm tra xem x có chia hết cho y không? x và y là 2 số nguyên được truyền vào.

Câu 11: Viết hàm in ra bảng cửu chương n. (n là một số nguyên từ 1 đến 10 được truyền vào)

Câu 12: Viết chương trình nhập vào 2 số nguyên a, b. Tạo một menu và mỗi chức năng trong menu là 1 hàm, gồm:

- (1) Tổng hai số
- (2) Hiệu hai số
- (3) Thương 2 số
- (4) Tích 2 số
- (5) Tìm số lớn hơn
- (7) USCLN của hai số
- (8) Bội số chung nhỏ nhất của hai số

II. Hàm đệ qui

II.1. Khái niệm đệ qui và hàm đệ qui

- Một khái niệm k được định nghĩa theo đệ qui nếu trong định nghĩa k có sử dụng lại chính khái niệm k .
- Một hàm được gọi là đệ qui nếu trong hàm đó gọi lại chính nó.

Ví dụ:

```
void ABC()
{
    ...
    ABC();    //gọi lại chính nó
    ...
}
```

II.2. Xây dựng hàm đệ qui

- Hàm đệ qui thường được xây dựng theo thuật toán sau:

if (trường hợp suy biến)

{ trình bày cách giải bài toán }

else

{ gọi đệ qui tới hàm đang lập với tham số khác }

- Nếu đệ qui không có điều kiện chặn (suy biến) thì đệ qui đó sẽ không khả thi

II.3. Một số ví dụ về hàm đệ qui

Ví dụ 1: Viết hàm đệ qui tính $N!$

```
long GiaiThua(int N)
{
    if (N==0 || N==1)    //Trường hợp suy biến
        return 1;
    else
        return N * GiaiThua(N-1) ;
        // Gọi lại chính nó với tham số khác
}
```

Quy trình thực hiện khi gọi hàm **GiaiThua(5)** như sau:

$GiaiThua(5) = 5 * GiaiThua(4)$; máy ghi nhớ và tính $GiaiThua(4)$

$\text{GiaiThua}(4) = 4 * \text{GiaiThua}(3)$; máy ghi nhớ và tính $\text{GiaiThua}(3)$

$\text{GiaiThua}(3) = 3 * \text{GiaiThua}(2)$; máy ghi nhớ và tính $\text{GiaiThua}(2)$

$\text{GiaiThua}(2) = 2 * \text{GiaiThua}(1)$; máy ghi nhớ và tính $\text{GiaiThua}(1)$

Với $\text{GiaiThua}(1)$ thì kết quả sẽ là 1, lúc này máy sẽ đi tính lần ngược như sau:

$\text{GiaiThua}(2) = 2 * 1 \rightarrow 2$

$\text{GiaiThua}(3) = 3 * 2 \rightarrow 6$

$\text{GiaiThua}(4) = 4 * 6 \rightarrow 24$

$\text{GiaiThua}(5) = 24 * 5 \rightarrow \mathbf{120}$

Ví dụ 2: Viết hàm đệ qui tính giá trị của số fibonacci thứ N

```
long Fibonacci(int N)
{
    if (N==1 || N==2)    //Trường hợp suy biến
        return 1;
    else
        return Fibonacci(N-1) + Fibonacci(N-2);
    // Gọi lại chính nó với tham số khác
}
```

Quy trình thực hiện khi gọi hàm **Fibonacci(7)** như sau:

$\text{Fibonacci}(7) = \text{Fibonacci}(6) + \text{Fibonacci}(5)$;

$\text{Fibonacci}(6) = \text{Fibonacci}(5) + \text{Fibonacci}(4)$;

$\text{Fibonacci}(5) = \text{Fibonacci}(4) + \text{Fibonacci}(3)$;

$\text{Fibonacci}(4) = \text{Fibonacci}(3) + \text{Fibonacci}(2)$;

$\text{Fibonacci}(3) = \text{Fibonacci}(2) + \text{Fibonacci}(1)$;

Với $\text{Fibonacci}(2) + \text{Fibonacci}(1) = 1 + 1 \rightarrow 2$

$\text{Fibonacci}(4) = 2 + 1 \rightarrow 3$

$\text{Fibonacci}(5) = 3 + 2 \rightarrow 5$

$\text{Fibonacci}(6) = 5 + 3 \rightarrow 8$

$\text{Fibonacci}(7) = 8 + 5 \rightarrow \mathbf{13}$

Ví dụ 3: Viết hàm tính tổng tất cả các số nguyên dương $\leq N$

```
int Tong(int N)
{
    if (N<=1)    //Trường hợp suy biến
        return N;
    else
        return N+ Tong(N-1);
    //Gọi lại chính nó với tham số khác
}
```

Qui trình thực hiện khi gọi hàm **Tong(5)** như sau:

$\text{Tong}(5) = 5 + \text{Tong}(4);$

$\text{Tong}(4) = 4 + \text{Tong}(3);$

$\text{Tong}(3) = 3 + \text{Tong}(2);$

$\text{Tong}(2) = 2 + \text{Tong}(1);$

Với $\text{Tong}(1) = 1$

$\text{Tong}(2) = 2 + 1 \rightarrow 3$

$\text{Tong}(3) = 3 + 3 \rightarrow 6$

$\text{Tong}(4) = 4 + 6 \rightarrow 10$

$\text{Tong}(5) = 5 + 10 \rightarrow 15$

III. Cấu trúc chương trình

III.1. Cấu trúc chung một chương trình

- Cấu trúc chung của một chương trình bao gồm các thành phần sau:

- + Phần 1: Khai báo thư viện

- + Phần 2: khai báo biến toàn cục, các hàm nguyên mẫu

- + Phần 3: chương trình chính

- + Phần 4: Xây dựng các hàm

- Ví dụ một chương trình C:

```
//Khai báo thư viện
```

```
#include "stdio.h"
```

```
#include "conio.h"
```

```
//Khai báo biến toàn cục
```

```
int A[100], N; // Khai báo biến A, N là các biến toàn cục
```

```
//Khai báo các hàm nguyên mẫu
```

```
void TaoMang(int A[], int &N);
```

```
void XuatMang(int A[], int N);  
//Chương trình chính - hàm main()  
void main()  
{  
    int Tong, i,k; // Khai báo các biến cục bộ  
    //các lệnh  
}  
//Xây dựng các hàm đã khai báo ở trên  
void TaoMang(int A[], int &N)  
{  
    //Khai báo biến cục bộ  
    //Các lệnh  
}  
void XuatMang(int A[], int N)  
{  
    //Khai báo biến cục bộ  
    //Các lệnh  
}
```

III.2. Tầm vực biến

- Trong chương trình có 2 loại biến, biến toàn cục (global variable) và biến cục bộ (local variable)
- Biến toàn cục: có tầm vực là toàn bộ chương trình
- Biến cục bộ: có tầm vực trong nội bộ 1 hàm. Trong trường hợp tên biến cục bộ trùng với tên biến toàn cục thì sẽ ưu tiên biến cục bộ trong phạm vi hàm đó.

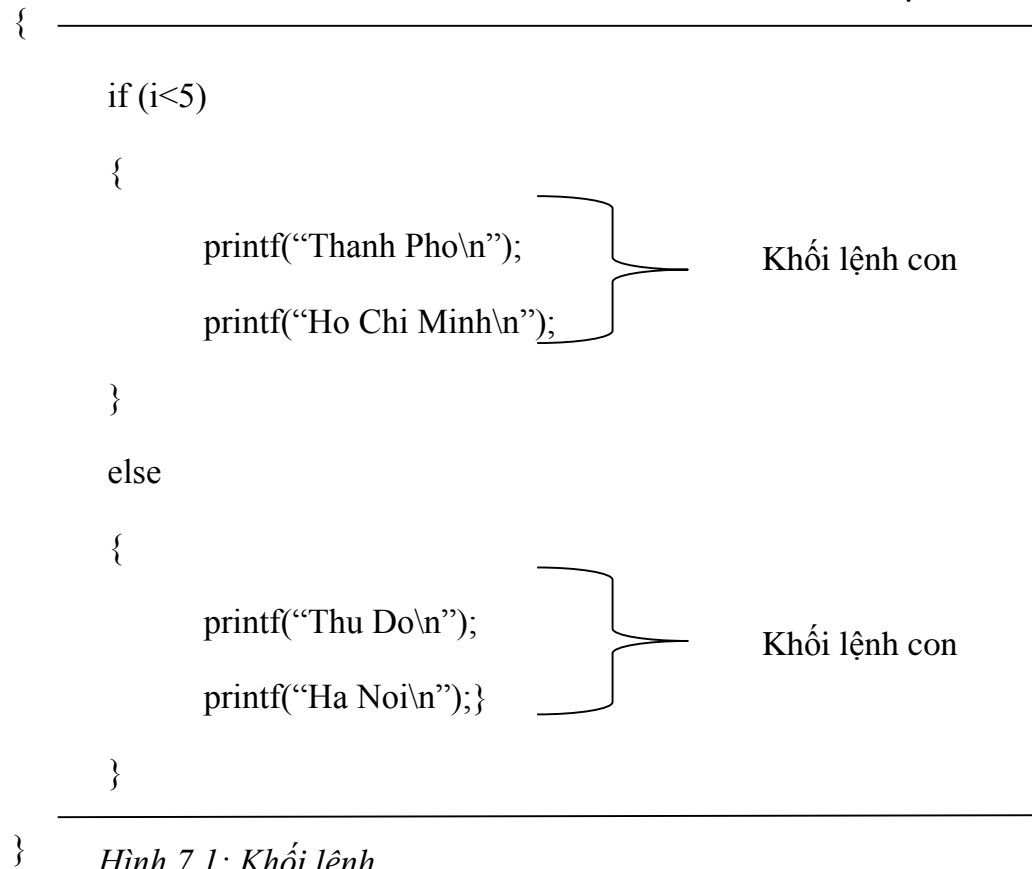
III.3. Khối lệnh

- Khối lệnh là một dãy các câu lệnh được bao bởi cặp dấu {...}
- Chỗ nào viết được 1 lệnh thì chỗ đó cũng đặt được 1 khối lệnh.
- Các khối lệnh có thể lồng nhau: bên trong một khối lệnh có thể có một khối lệnh khác. Sự lồng nhau như thế là không hạn chế.

Ví dụ:

```
for (int i=0; i<10; i++)
```

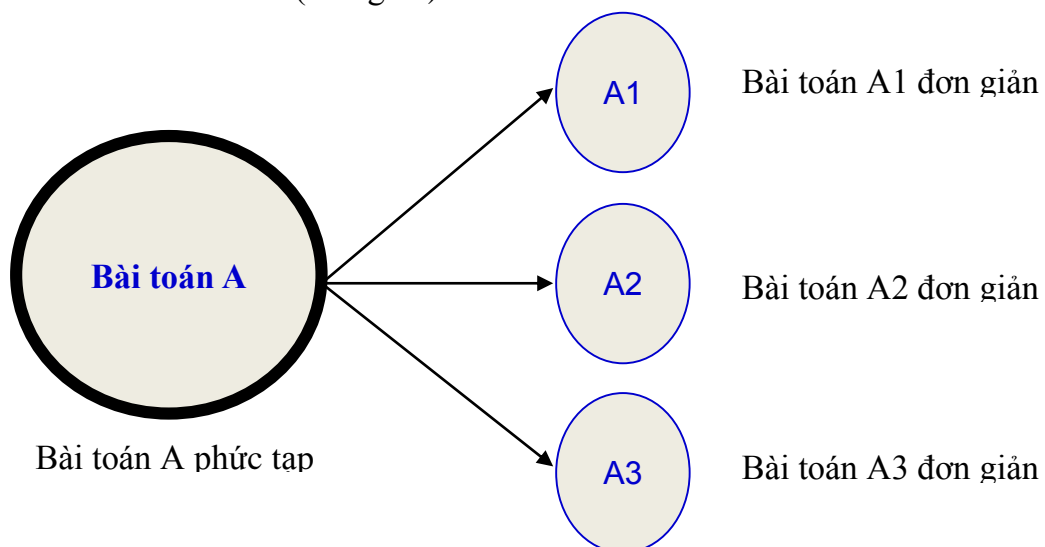
Khối lệnh



Hình 7.1: Khối lệnh

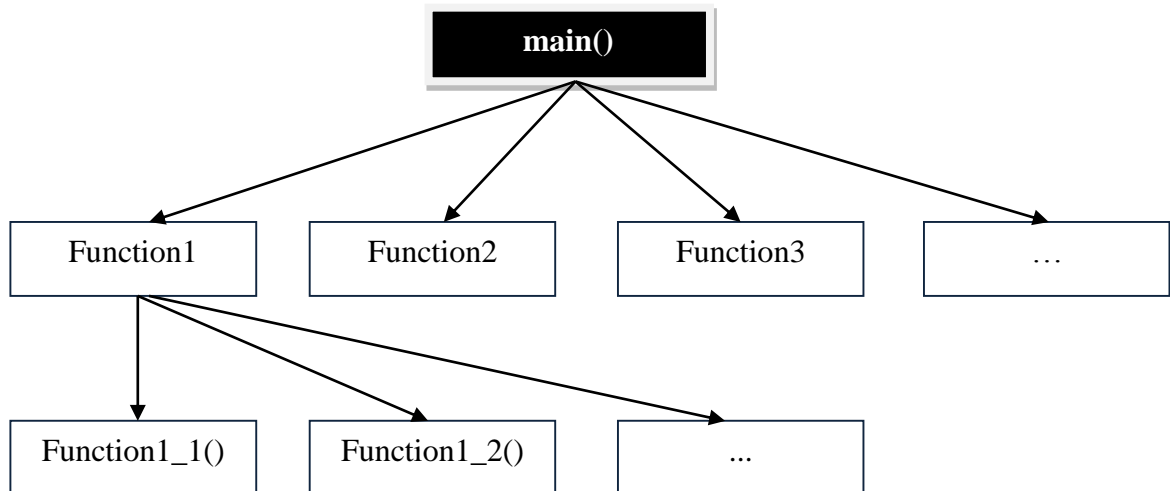
IV. Phân rã bài toán

- Phân rã bài toán là cách phân chia một bài toán lớn (phức tạp) thành những bài toán nhỏ hơn (đơn giản).



Hình 2.2a: Phân rã bài toán

- Trong lập trình phân rã bài toán tức là chia nhỏ chương trình lớn thành nhiều chương trình nhỏ hơn. Mỗi chương trình nhỏ đó có thể là một hàm. Một hàm có thể chia nhỏ thành nhiều hàm nhỏ hơn.



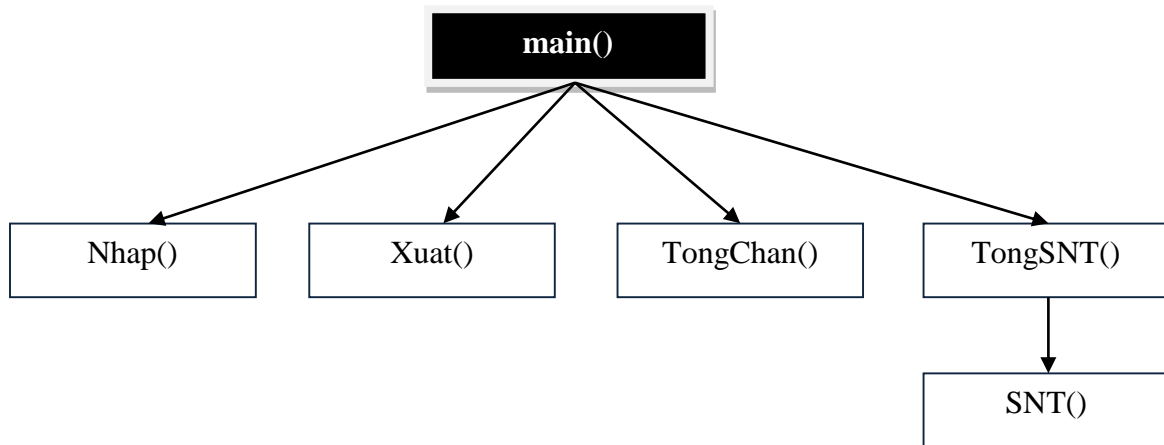
Hình 2.2b: Lưu đồ giải thuật

Ví dụ:

Viết chương trình Nhập một mảng số nguyên có N phần tử rồi thực hiện các công việc: xuất mảng đã nhập ra màn hình; tính tổng các phần tử chẵn; tính tổng các phần tử là số nguyên tố. (Tham khảo bài mảng)

Chúng ta có thể phân rã ví dụ 2 thành các hàm như sau:

- Hàm **Nhap(int A[], int &n)**: Nhập mảng
 - Hàm **Xuat(int A[], int n)**: Xuất mảng
 - Hàm **TongChan(int A[], int n)**: Tính tổng các p.tử chẵn
 - Hàm **TongSNT(int A[], int n)**: Tổng các p.tử là số N.tố
- + Hàm **SNT(int k)**: Kiểm tra số nguyên tố



Hình 2.2c: Phân rã bài toán

Chương trình viết bằng code C như sau:

```
//Khai báo thư viện
#include <stdio.h>
#include <conio.h>
#include <math.h>
//Khai báo các hàm nguyên mẫu
void Nhap(int A[], int &n);
void Xuat(int A[], int n);
int TongChan(int A[], int n);
int TongSNT(int A[], int n);
int SNT(int k);
//Chương trình chính
void main()
{
    int A[20], N;
    Nhap(A,N);
    Xuat(A,N);
    printf("\nTong chan = %d",TongChan(A,N));
    printf("\nTong so nguyen to = %d",TongSNT(A,N));
    getch();
}
//Xây dựng các hàm
void Nhap(int A[], int &n) //Hàm nhập các phần tử của mảng
{
    printf("Nhap so phan tu:");
    scanf("%d",&n);
    for (int i=0; i<n; i++)
    {
```

```
        printf("A[%d]:",i);
        scanf("%d",&A[i]);
    }
}
//Hàm in các phần tử của mảng ra màn hình
void Xuat(int A[], int n)    {
    printf("Mang:");
    for (int i=0; i<n; i++)
        printf("%3d",A[i]);
}
int TongChan(int A[], int n) //Hàm tính tổng các phần tử chẵn
{
    int Tong=0;
    for(int i=0; i<n; i++)
        if (A[i]%2==0) Tong +=A[i];
    return Tong;
}
int TongSNT(int A[], int n) //Hàm tính tổng các số nguyên tố
{
    int Tong=0;
    for(int i=0; i<n; i++)
        if (SNT(A[i])) Tong +=A[i];
    return Tong;
}
int SNT(int k) //Hàm kiểm tra số nguyên tố
{
    for (int i=2; i<=sqrtl(k); i++)
        if (k%i==0)
            return 0;
    return 1;
}
```

V. Bài tập tự làm

Bài 1: Viết chương trình vẽ các hình dưới đây, mỗi hình là một hàm có 2 tham số (chiều cao và ký tự vẽ).

Ví dụ chiều cao =4, ký tự vẽ là dấu sao *

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

H1

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

H2

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

H3

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

H4

Bài 2: Viết chương trình xuất ra màn hình:

- In ra N số nguyên tố đầu tiên
- In ra N số Fibonacci đầu tiên
- In ra N số chính phương đầu tiên
- In ra N số hoàn thiện đầu tiên (số hoàn thiện là số có tổng các ước số nhỏ hơn nó bằng chính nó). Ví dụ: $6 = 1 + 2 + 3$
- Lưu ý là chương trình có menu, cho phép người dùng lựa chọn các chức năng tùy ý.

Bài 3: Viết chương cho phép người dùng nhập vào năm sinh của mình. Tính xem người đó sinh vào tuổi gì? Chương trình cho phép người dùng nhập lại tuổi nhiều lần, chỉ khi nào nhấn phím ESC mới thoát.

Bài 4: Viết chương trình chuyển đổi hệ số 10 sang hệ số 2, hệ số 8 và hệ số 16. Chương trình có các chức năng như sau:

- 1/ Nhập một số hệ 10
- 2/ Chuyển sang hệ 2
- 3/ Chuyển sang hệ 8
- 4/ Chuyển sang hệ 16
- 5/ Thoát chương trình

Bài 5: Viết chương trình nhập một số nguyên dương N. Chương trình có các chức năng cho phép người dùng lựa chọn như sau:

- 1/ Nhập một số nguyên dương N
- 2/ Tính tổng các số nguyên tố nhỏ hơn N
- 3/ Kiểm tra N có phải là số chính phương không
- 4/ Tìm tất cả các số hoàn hảo nhỏ hơn hoặc bằng N
- 5/ Thoát chương trình

**Lưu ý: số hoàn hảo là số có tổng các ước số nhỏ hơn hoặc bằng chính nó*

Câu hỏi trắc nghiệm:

Câu 1: Một hàm có thể được sử dụng bao nhiêu lần trong chương trình?

a/ Chỉ một lần duy nhất

b/ Chỉ tối đa 2 lần

c/ Chỉ tối đa 3 lần

d/ Sử dụng bao nhiêu lần không hạn chế

Câu 2: Có mấy cách truyền tham số?

a/ Có một cách truyền tham số

b/ Có hai cách truyền tham số (truyền bằng trị và truyền bằng biến)

c/ Có ba cách truyền tham số

d/ Có bốn cách truyền tham số

Câu 3: Khi truyền bằng biến thì trước tham số hình thức phải thêm ký tự gì?

a/ \$

b/ %

c/ &

d/ #

Câu 4: Hàm không trả về giá trị thì khai báo kiểu trả về là?

a/ Kiểu int

b/ Kiểu void

c/ Kiểu char

d/ Kiểu string

Câu 5: Xây dựng hàm sau sai ở chỗ nào?

```
long Tong(int N)
{
    long T = 0;
    for (int i=1; i<=N; i++)
        T += i;
}
```

a/ Hàm này không có chỗ nào sai

b/ Sai ở hàng $T += i$; Phải viết lại $T = T + i$;

c/ Thiếu câu lệnh **return T**; ở cuối thân hàm

d/ Phải viết lại vòng lặp **for (int i=0; I < N; i++)**

Câu 6: Hãy điền vào chỗ trống ở đoạn mã sau:

```
int TongLe(int N) //Hàm tính tổng các số lẻ
{
    int Tong = 0;
    for (int i=1; i<=N; i++)
        if (.....)
            Tong +=I;
    return Tong;
}
```

a/ $i == 1$

b/ $i \% 2 == 1$

c/ $1 == i \% 2$

d/ $1 = i \% 2$

Câu 7: Hãy điền vào chỗ trống ở đoạn mã sau:

```
int USCLN(int a, int b) //Hàm tìm ước số chung lớn nhất
{
    while (a*b !=0)
    {
        if (a>b)
            a =a-b;
        else
            b = b - a;
    }
    return (.....);
}
```

a/ a

b/ b

c/ a - b

d/ $a + b$

Câu 8: Hãy điền vào chỗ trống ở đoạn mã sau:

```
float Average(int a, int b)
{
    return (.....);
}
```

a/ a / b

b/ $(a+b)/2$

c/ $(\text{float}) (a+b)/2$

d/ $(\text{float}) ((a+b)/2)$

Câu 9: Hãy điền vào chỗ trống ở đoạn mã sau:

```
int Fibonacci(int N) // Tính số Fibonacci thứ N
{
    int F, F1=1, F2=1;
    for (int i=0; i<N; i++)
    {
        if (i<2)
            F=1;
        else
            {.....}
    }
    return F;
}
```

a/ $F = F1 + F2$;

b/ $F = F1 + F2$; $F1 = F$; $F2 = F1$;

c/ $F = F1 + F2$; $F2 = F1$; $F1 = F$;

d/ $F = N$;

Câu 10: Hãy điền vào chỗ trống ở đoạn mã sau:

```
void Swap(int &a, int &b) //Hoán vị
{
    .....
    a = b;
    b = a;
}
```

a/ int c

b/ int c = a;

c/ int c = b

d/ int c = a+ b;

BÀI 8: MẢNG - ARRAY

I. Mảng một chiều

I.1. Khái niệm

Mảng là một tập các phần tử có cùng kiểu dữ liệu, được đặt liên tiếp nhau trong bộ nhớ. Các phần tử trong mảng được xác định bằng chỉ số.

Mảng một chiều là mảng mà trong đó các phần tử của nó được xác định bằng 1 chỉ số, chỉ số được bắt đầu từ 0.

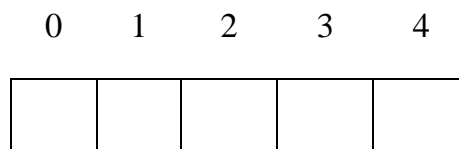
I.2. Khai báo

Cú pháp: **kiểu_dữ_liệu** **tên_mảng[số_phần_tử_tối_đa]** ;

- Kiểu dữ liệu có thể là char, int, long, float, double ...
- Số phần tử tối đa là một hằng số, do đó mảng là một vùng nhớ tĩnh mà kích thước được xác định sẵn trước khi thực thi chương trình.

Ví dụ: Khai báo một mảng số nguyên có 5 phần tử

```
int Arr[5];
```



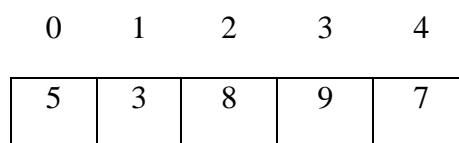
Hình 8.1: Minh họa mảng một chiều

Khởi tạo mảng: chúng ta khởi tạo giá trị ban đầu trong mảng như sau:

```
int Arr[5] = {5, 3, 8, 9, 7};
```

hoặc

```
int Arr[] = {5, 3, 8, 9, 7}; // mảng này lưu trữ 5 số nguyên
```



Hình 8.2: Minh họa mảng một chiều

I.3. Truy xuất các phần tử của mảng

Để truy xuất đến các phần tử của mảng, dùng chỉ số (index), index bắt đầu từ 0 đến $n-1$. Phần tử đầu tiên có chỉ số là 0, phần tử cuối cùng có chỉ số là $n-1$ (n là tổng số phần tử của mảng).

```
tenmang[index];
```

Arr[0]	Arr[1]	Arr[2]	Arr[3]	Arr[4]
5	3	8	9	7

Hình 8.3: Minh họa mảng một chiều

Ví dụ:

- Truy xuất đến phần tử thứ 3 của mảng trên là Arr[3] có giá trị 9.
- `int a = Arr[0] + Arr[2];` //a sẽ có kết quả là $5+8 = 13$
- `Arr[4] = 10;` // cập nhật (update) phần tử thứ 4 thành 10

I.4. Các thao tác trên mảng 1 chiều

a/ Duyệt mảng: Để thao tác trên mảng 1 chiều, có 2 cách thức duyệt mảng: duyệt xuôi và duyệt ngược:

Giả sử mảng có n phần tử hiện hành.

Duyệt xuôi:

- **Duyệt xuôi không có điều kiện:** từ phần tử thứ 0 , 1, 2 ... $n-1$

5	3	8	...	7
0	1	2	...	$n-1$

→

Hình 8.4: Minh họa duyệt xuôi mảng một chiều

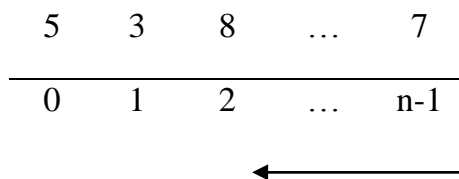
```
for(int i=0; i<=n-1; i++)
    xử lý mang[i];
```

- Duyệt xuôi có điều kiện: từ phần tử thứ 0 , 1, 2 ... n-1

```
for(int i=0; i<n; i++)  
    if(điều kiện đúng) xử lý mang[i];
```

Duyệt ngược:

- Duyệt ngược **không** có điều kiện: từ phần tử thứ n-1, n-2,0



Hình 8.5: Minh họa duyệt ngược mảng một chiều

```
for(int i=n-1; i>=0; i--)  
    xử lý mang[i];
```

- Duyệt ngược có điều kiện: từ phần tử thứ n-1, n-2,0

```
for(int i=n-1; i>=0; i--)  
    if(điều kiện đúng) xử lý mang[i];
```

b/ Tìm kiếm

Tìm vị trí của phần tử có giá trị x trong mảng?

Ý tưởng: Duyệt từ phần tử đầu đến phần tử cuối của mảng. Kiểm tra từng phần tử xem có bằng với phần tử cần tìm không? Nếu bằng thì dừng lại và trả về vị trí của phần tử hiện tại. Đi đến phần tử cuối cùng mà vẫn không tìm thấy phần tử nào bằng phần tử cần tìm thì trả về -1 (ta quy ước -1 là giá trị không tìm thấy)

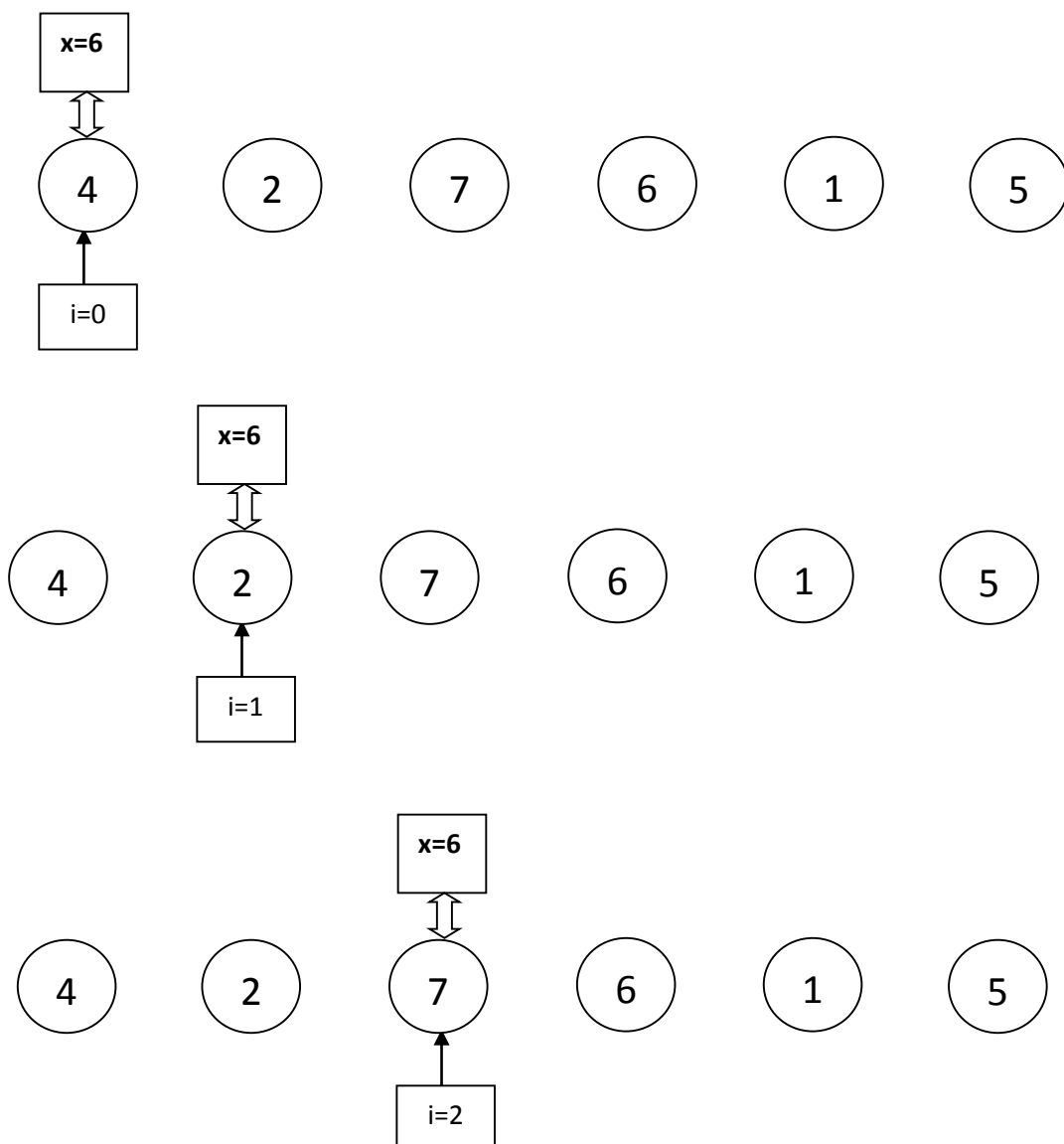
//Mã giả:

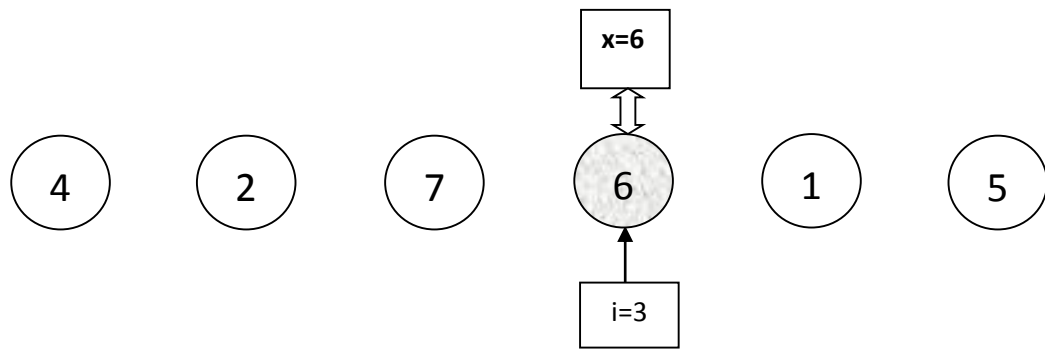
```
FOR i=0 TO n-1
    IF mang[i] = x THEN
        RETURN i
    ENDIF
ENDFOR
RETURN -1
```

Ví dụ:

Cho dãy số 4 2 7 6 1 5

Hãy tìm vị trí của phần tử x=6 nằm trong dãy





Tìm thấy phần tử $x=6$ tại vị trí $i=3$

Hình 8.6: Minh họa giải thuật tìm kiếm

Cài đặt

```
for(int i=0; i<=n-1; i++)  
    if(mang[i]==x) return i;  
return -1;
```

b/ Sắp xếp các phần tử trong mảng (thuật toán Bubble Sort)

Ý tưởng: Duyệt từ phần tử cuối đến đầu dãy chưa sắp xếp, tìm những cặp kế nhau nghịch thế (không đúng theo thứ tự) và hoán vị những cặp này. Lặp lại quá trình này $n-1$ lần.

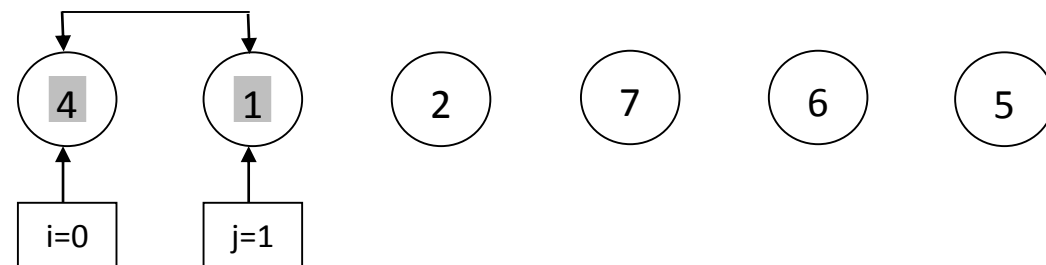
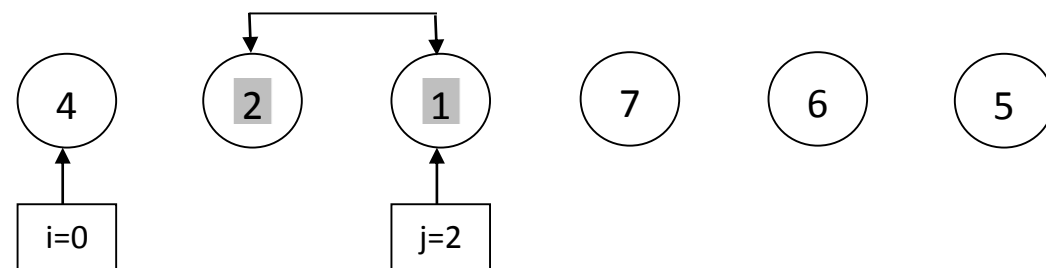
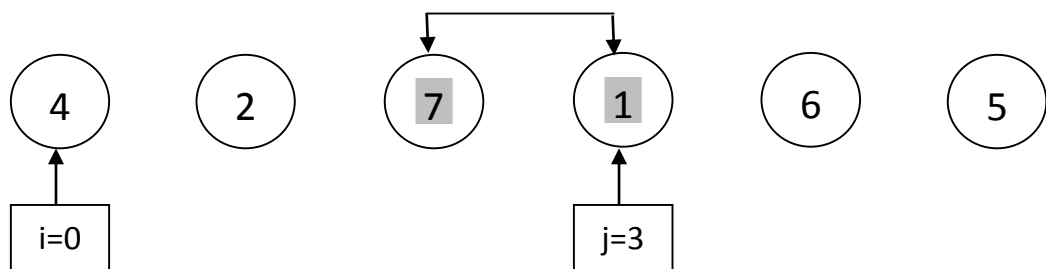
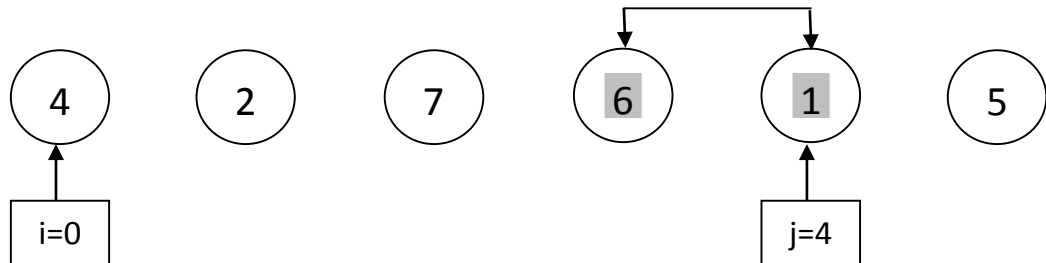
Mã giả:

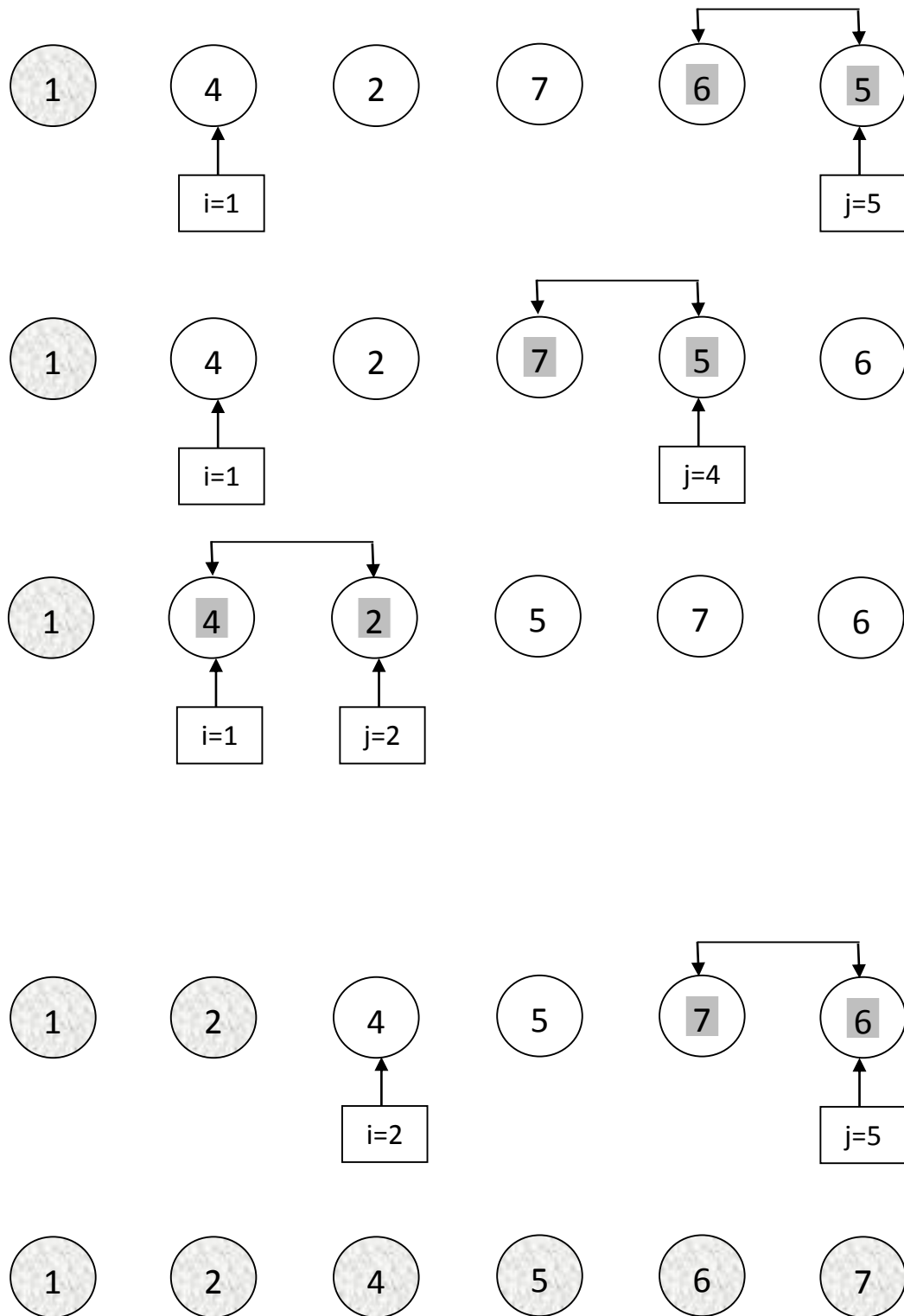
```
FOR i=0 TO n-2  
    FOR j=n-1 TO i+1  
        IF (mang[j-1] and mang[j] Nghịch thế) THEN  
            SWAP (mang[j-1] and mang[j])  
        ENDIF  
    ENDFOR  
ENDFOR
```

Ví dụ:

Cho dãy số: 4 2 7 6 1 5

Sắp xếp dãy trên theo thứ tự tăng dần





Hình 8.7: Minh họa giải thuật sắp xếp

Cài đặt

- Sắp xếp tăng dần:

```
for(int i=0; i<n-1; i++)
    for(int j=n-1; j>i; j--)
        if(mang[j-1]>mang[j])
        {
            int tam=mang[j-1];
            mang[j-1] = mang[j];
            mang[j] = tam;
        }
```

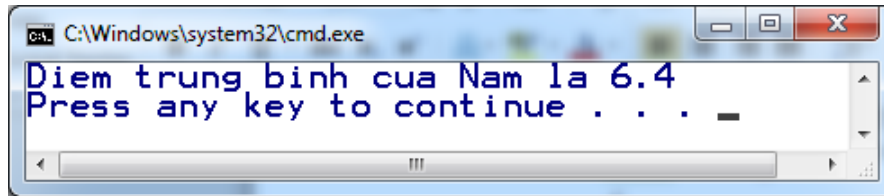
- Sắp xếp giảm dần:

```
for(int i=0; i<n-1; i++)
    for(int j=n-1; j>i; j--)
        if(mang[j-1]<mang[j])
        {
            int tam=mang[j-1];
            mang[j-1] = mang[j];
            mang[j] = tam;
        }
```

I.5. Bài tập minh họa

Bài 1: Khai báo 1 mảng các số nguyên lưu trữ 5 cột điểm cuối kỳ của sinh viên có tên Nam. Tính điểm trung bình và xuất ra màn hình có 1 số lẻ.

```
#include <stdio.h>
void main()
{
    int Nam[5] = {6, 4, 9, 10, 3};
    int tongdiem = 0;
    float diemtb;
    for(int i=0; i<5; i++)
        tongdiem = tongdiem+Nam[i];
    diemtb = tongdiem*1.0/5;
    printf("Diem trung binh cua Nam la %.1f", diemtb);
}
```

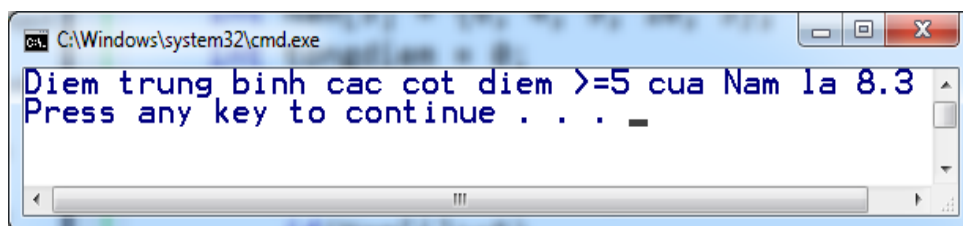



Hình 8.8: Kết quả của chương trình tính điểm trung bình

Bài 2: Giống với dữ liệu ở bài trên. Tính điểm trung bình các cột điểm lớn hơn hoặc bằng 5 và xuất ra màn hình.

```
#include <stdio.h>
void main()
{
    int Nam[5] = {6, 4, 9, 10, 3};
    int tongdiem = 0;
    float diemtb=0;
    int n=0; //đếm số phần tử >=5
    for(int i=0; i<5; i++)
        if (Nam[i]>=5)
        {
            tongdiem = tongdiem+Nam[i];
            n++;
        }

    if(n>0) diemtb = tongdiem*1.0/n;
    printf("Diem trung binh cac cot diem >=5 cua Nam là %.1f",
    diemtb);
}
```



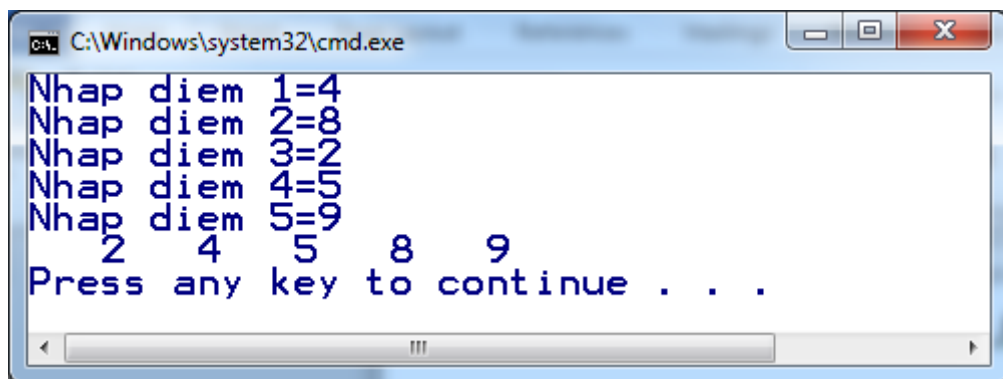
Hình 8.9: Kết quả chương trình tính điểm trung bình các cột >=5

Bài 3: Khai báo mảng gồm 5 số nguyên lưu trữ các cột điểm của Nam. Nhập điểm của Nam từ bàn phím, sắp xếp các cột điểm theo thứ tự tăng dần rồi xuất ra màn hình.

```
#include <stdio.h>
void main()
{
    int Nam[5] ;

    for(int i=0; i<5; i++)
    {
        printf("Nhap diem %d=", i+1);
        scanf("%d", &Nam[i]);
    }

    for(int i=0; i<4; i++)
        for(int j=4; j>i; j--)
            if(Nam[j-1]>Nam[j])
            {
                int tam=Nam[j-1];
                Nam[j-1] = Nam[j];
                Nam[j] = tam;
            }
    for(int i=0; i<5; i++)
        printf("%4d", Nam[i]);
}
```



Hình 8.10: Kết quả chương trình sắp xếp các cột điểm theo thứ tự tăng dần.

I.6. Bài tập tự làm

Bài 1: Cho mảng A 1 chiều có N phần tử ($N \leq 100$) viết chương trình

- Nhập mảng A
- Xuất mảng A
- Đếm số phần tử âm trong mảng

- d. Tính tổng mảng
- e. Kiểm tra mảng có đối xứng không. (Ví dụ: 7 4 3 3 4 7 là mảng đối xứng)
- f. Nhập vào phần tử x, tìm vị trí phần tử này trong mảng (nếu có).

Bài 2: Viết chương trình tạo một mảng có N phần tử số nguyên ngẫu nhiên. Các phần tử nằm trong khoảng từ 0 đến 20. Xuất các phần tử của mảng ra màn hình. Tính tổng các phần tử của mảng. Sắp xếp mảng tăng dần.

Bài 3: Tính đa thức bậc n

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 \quad \text{với } n \text{ nguyên và mảng thực } a.$$

Bài 4: Viết chương trình tạo một mảng có N phần tử số nguyên ngẫu nhiên. Các phần tử nằm trong khoảng từ 1 đến 200. Xuất các phần tử của mảng ra màn hình. Tìm Phần tử Max, Phần tử Min. Tính tổng các phần tử chẵn, tổng các phần tử lẻ. Sắp xếp mảng giảm dần.

II. Mảng nhiều chiều

II.1. Khái niệm

Mảng nhiều chiều được định nghĩa như là mảng của mảng. Chẳng hạn mảng hai chiều là mảng của mảng 1 chiều hay nói khác đi là danh sách các mảng 1 chiều. Mảng 2 chiều giống như 1 bảng (table) các phần tử có cùng kiểu, đây là mảng phổ biến nhất trong mảng nhiều chiều.

II.2. Khai báo mảng 2 chiều:

Mảng 2 chiều còn được gọi là ma trận (Matrix) có cú pháp như sau:

Kiểu_dữ_liệu Tên_Mảng[Số_dòng_tối_đa][Số_cột_tối_đa];

Trong đó:

- Kiểu_dữ_liệu: có thể là các kiểu dữ liệu cơ bản như char, int, float,.... hoặc các kiểu dữ liệu do người dùng định nghĩa.
- Số dòng tối đa và số cột tối đa là các hằng số

Ví dụ:

```
int Matrix[3][5];
```

	0	1	2	3	4
0					
1					
2					

Matrix[1][3]

Hình 8.11: Minh họa mảng hai chiều

Để truy xuất đến các phần tử trong mảng, chúng ta dùng chỉ số dòng và chỉ số cột. Chỉ số dòng bắt đầu từ 0 đến số dòng tối đa trừ 1, chỉ số cột bắt đầu từ 0 đến số cột tối đa trừ 1. Như hình ở trên, phần tử ở dòng số 1, cột thứ 3 có màu đậm được truy xuất là `Matrix[1][3]`

Khởi tạo giá trị ban đầu cho mảng 2 chiều:

```
int Matrix[2][3] = {{3, 6, 9},{1, 2, 3}};
```

	0	1	2
0	3	6	9
1	1	2	3

Hình 8.12: Minh họa mảng hai chiều

II.3. Các thao tác trên mảng 2 chiều

Để thao tác trên mảng 2 chiều, ta có 2 cách thức duyệt mảng. Giả sử mảng có d dòng hiện hành và c cột hiện hành.

a/ Duyệt mảng không có điều kiện:

```
for(int i=0; i<d; i++)  
    for(int j=0; j<c; j++)  
        xử lý Matrix[i][j];
```

b/ Duyệt mảng có điều kiện:

```
for(int i=0; i<d; i++)  
    for(int j=0; j<c; j++)  
        if(điều kiện đúng) xử lý Matrix[i][j];
```

II.4. Bài tập minh họa

Bài 1: Cho ma trận MT có $N \times M$ phần tử ($N, M \leq 100$) viết chương trình nhập ma trận MT và xuất các giá trị trong ma trận MT ra màn hình.

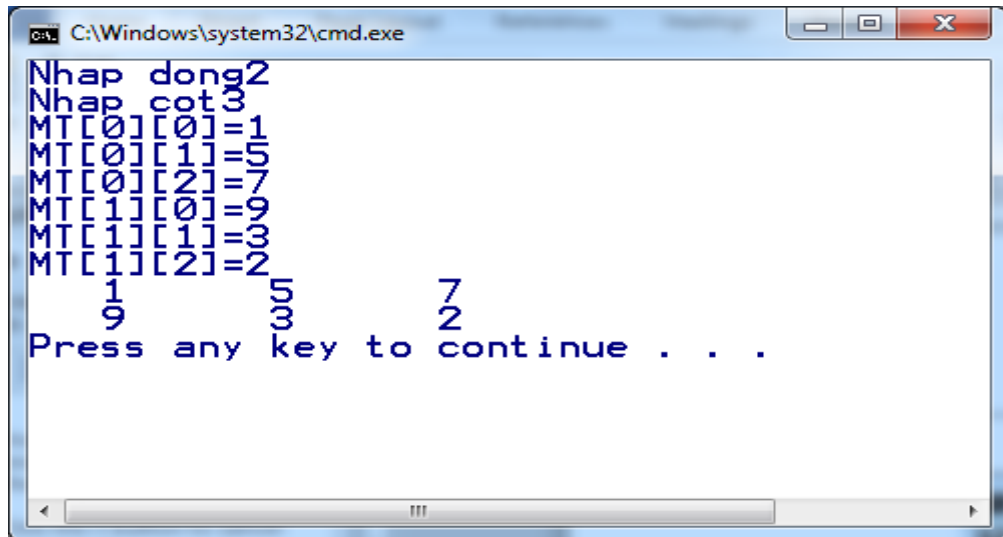
```
#include <stdio.h>
void NhapMT(int MT[][100], int &d, int &c)
{
    do{
        printf("Nhap dong");
        scanf("%d", &d);
    }while(d>100||d<=0);

    do{
        printf("Nhap cot");
        scanf("%d", &c);
    }while(c>100||c<=0);

    // nhập các phần tử vào ma trận
    for(int i=0; i<d; i++)
        for(int j=0; j<c; j++)
        {
            printf("MT[%d][%d]=", i, j);
            scanf("%d", &MT[i][j]);
        }
}

void XuatMT(int MT[][100], int d, int c)
{
    for(int i=0; i<d; i++)
    {
        for(int j=0; j<c; j++)
            printf("    %d    ", MT[i][j]);
        printf("\n"); // kết thúc 1 dòng và xuống dòng
    }
}

void main()
{
    int MT[100][100], d, c; // số dòng tối đa và số cột
    //tối đa là 100; d, c là số
    //dòng, cột hiện hành
    NhapMT(MT, d, c);
    XuatMT(MT, d, c);
}
```



Hình 8.13: Kết quả chương trình nhập xuất mảng hai chiều

Bài 2: Cho ma trận MT có 10 dòng, 10 cột. Viết chương trình khai báo và khởi tạo ngẫu nhiên các giá trị cho ma trận MT và xuất các giá trị chẵn trong ma trận MT ra màn hình.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void main()
{
    int MT[10][10];
    //Tạo mảng ngẫu nhiên
    srand(time(0)); //Tạo dãy khác nhau sau mỗi lần chạy
    for(int i=0; i<10; i++)
        for(int j=0; j<10; j++)
            MT[i][j]=rand() % 50; //ngẫu nhiên từ 0 → 49
    //Xuất các phần tử chẵn
    for(int i=0; i<10; i++)
        for(int j=0; j<10; j++)
            if(MT[i][j]%2==0)
                printf("%4d",MT[i][j]);
}
```

II.5. Bài tập tự làm

Bài 1: Cho ma trận MT 2 chiều có $N \times M$ phần tử ($N, M \leq 100$) viết chương trình

a. Nhập các giá trị cho ma trận

- b. Xuất ma trận MT
- c. Đếm số phần tử âm
- d. Tính tổng ma trận
- e. Tính tổng dòng k
- f. Tính tổng cột k

Bài 2: Tạo mảng 2 chiều có $N \times M$ phần tử, khởi tạo các phần tử có giá trị ngẫu nhiên trong khoảng từ -50 đến 50. Hãy hoán vị hàng thành cột, cột thành hàng.

III. Mảng là một tham số truyền vào hàm

Chúng ta có thể truyền mảng vào hàm như 1 tham số

III.1. Mảng một chiều là tham số truyền vào hàm

Ví dụ khai báo một hàm xuất các giá trị trong mảng ra màn hình.

```
void XuatMang(int arr[], int n) // n là số phần tử hiện hành
{
    for(int i=0; i<n; i++)
        printf(" %d ", arr[i]);
}
```

III.2. Mảng hai chiều là tham số truyền vào hàm

Ví dụ khai báo một hàm xuất các giá trị trong mảng hai chiều ra màn hình.

```
// n là số phần tử hiện hành, 100 là số cột tối đa
void XuatMT(int MT[][100], int d, int c)
{
    for(int i=0; i<d; i++)
    {
        for(int j=0; j<c; j++) printf(" %d ", MT[i][j]);
        printf("\n");
    }
}
```

III.3. Bài tập minh họa

Cho mảng một chiều A có N phần tử ($N \leq 100$) viết các hàm sau:

- a. Nhập mảng A

b. Xuất mảng A

c. Đếm số phần tử âm trong mảng

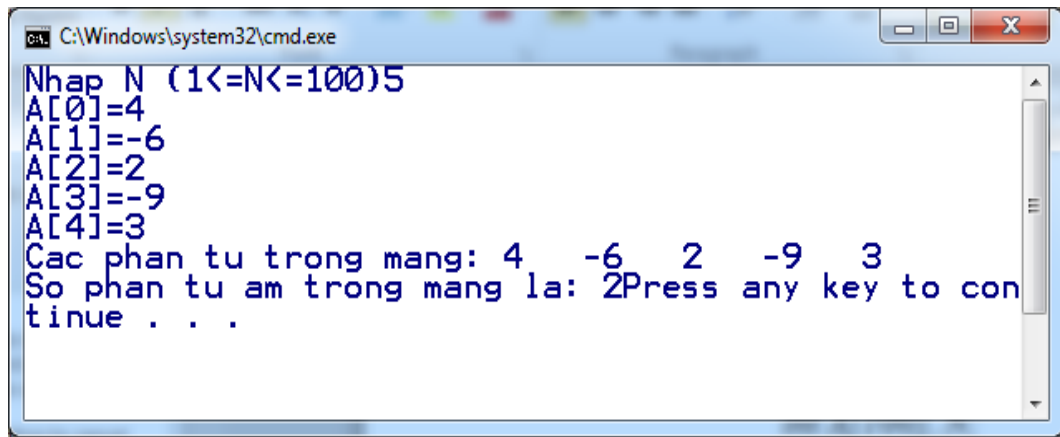
```
#include <stdio.h>
void NhapMang(int A[], int &N)
{
    do{
        printf("Nhap N (1<=N<=100) ");
        scanf("%d", &N);
    }while(N<1||N>100);

    for(int i=0; i<N; i++)
    {
        printf("A[%d]=", i);
        scanf("%d", &A[i]);
    }
}

void XuatMang(int A[], int N)
{
    printf("Cac phan tu trong mang:");
    for(int i=0; i<N; i++)
        printf(" %d  ", A[i]);
}

int SoPhanTuAm(int A[], int N)
{
    int dem=0;
    for(int i=0; i<N; i++)
        if(A[i]<0) dem++;
    return dem;
}

void main()
{
    int A[100], N;
    NhapMang(A, N);
    XuatMang(A, N);
    printf("\nSo phan tu am: %d", SoPhanTuAm(A, N));
}
```

```
C:\Windows\system32\cmd.exe
Nhap N (1<=N<=100)5
A[0]=4
A[1]=-6
A[2]=2
A[3]=-9
A[4]=3
Cac phan tu trong mang: 4 -6 2 -9 3
So phan tu am trong mang la: 2Press any key to con
tinue . . .
```

Hình 8.14: Kết quả chương trình nhập, xuất và đếm số phần tử âm trong mảng

III.4. Bài tập tự làm

Bài 1: Cho mảng một chiều A có N phần tử ($N \leq 100$) viết các hàm sau:

- Tính tổng các phần tử lẻ trong mảng
- Đếm các phần tử dương trong mảng
- Đếm các phần tử chia hết cho 3 trong mảng
- Kiểm tra xem mảng có đối xứng không
- Tìm phần tử lớn nhất trong mảng
- Tìm phần tử nhỏ nhất trong mảng
- Sắp xếp mảng
- Kiểm tra phần tử x có nằm trong mảng không
- Tìm cặp số nguyên kế nhau có tổng lớn nhất.

Bài 2: Viết chương trình tạo một mảng có N phần tử số nguyên ngẫu nhiên. Các phần tử nằm trong khoảng từ -100 đến 100. Tìm vị trí của 2 phần tử liên nhau có tổng giá trị tuyệt đối nhỏ nhất.

Bài 3: Viết chương trình tạo một mảng có N phần tử số nguyên ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1 đến 1000. Tính tổng các phần tử là số nguyên tố.

Bài 4: Viết chương trình tạo một mảng có 100 phần tử số thực ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1.0 đến 5.0. Hãy tạo ra một mảng khác và kiểm tra mảng mới này có phải là mảng con của mảng trên không.

Bài 5: Cho mảng 2 chiều MT có r dòng và c cột. Viết các hàm sau:

- a. Nhập mảng
- b. Xuất mảng
- c. Tính tổng dòng k của mảng
- d. Tính tổng cột k của mảng
- e. Tính tổng đường chéo chính
- f. Tìm cột có tổng lớn nhất
- g. Sắp xếp cột k tăng dần.

Bài 6: Viết chương trình tạo một mảng 2 chiều có MxN phần tử số nguyên ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1 đến 1000. Tính tổng các phần tử nằm trên đường viền.

Bài 7: Viết chương trình tạo một mảng 2 chiều có MxN phần tử số nguyên ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1 đến 1000. Hãy chuyển mảng từ dạng MxN sang NxM (Hàng thành cột, cột thành hàng)

Bài 8: Viết chương trình nhân 2 ma trận

IV. Câu hỏi trắc nghiệm

Câu 1. Cách khai báo mảng một chiều sau đây là Sai:

a/ `int a[];`

b/ `int a[5];`

c/ `int a[] = { 1, 2, 3, 4};`

d/ `int a[4] = { 1, 2, 3, 4};`

Câu 2. Cách khai báo mảng một chiều sau đây là Đúng:

a/ `int a[] = { 1, 2, 3, 4};`

b/ `int a[4] = { 1, 2, 3, 4};`

c/ A, B đều đúng

d/ A, B đều sai

Câu 3. Cách khai báo mảng hai chiều sau đây là Đúng:

a/ `int a[][] = { 1, 2, 3, 4};`

b/ `int a[2][2] = { 1, 2, 3, 4};`

c/ A, B đều đúng

d/ A, B đều sai

Câu 4. Cách khai báo mảng hai chiều sau đây là Đúng:

a/ `int a[][2] = { 1, 2, 3, 4};`

b/ `int a[2][2] = { 1, 2, 3, 4};`

c/ `int a[][2] = {{ 1, 2}, {3, 4}};`

d/ Tất cả đều đúng

Câu 5. Truyền một mảng một chiều arr vào hàm, cách khai báo hàm nào sau đây là đúng:

a/ `void func(int arr[], int n)`

b/ `void func(int arr[5], int n)`

c/ A, B sai

d/ A, B đúng

Câu 6. Truyền một mảng một chiều arr vào hàm, cách khai báo hàm nào sau đây là đúng:

a/ `void func(int []arr, int n)`

b/ `void func(int [5]arr, int n)`

c/ A, B sai

d/ A, B đúng

Câu 7. Cho một hàm func được khai báo như sau:

```
void func(int arr[], int n)
```

```
{
```

```
...
```

```
}
```

Cách nào gọi hàm func sau là sai.

a/ `func(int arr[], n);`

b/ func(arr [], n)

c/ A, B sai

d/ A, B đúng

Câu 8. Cho một hàm func được khai báo như sau:

```
void func(int arr[], int n)

{
    ...
}
```

Cách nào gọi hàm func sau là đúng.

a/ func(int arr[], n);

b/ func(arr [], n)

c/ func(arr, n);

d/ Tất cả đều đúng

Câu 9. Truyền một mảng hai chiều mt vào hàm, cách khai báo hàm nào sau đây là đúng:

a/ void func(int mt[4][4], int d, int c)

b/ void func(int mt[][4], int d, int c)

c/ A, B đúng

d/ A, B sai

Câu 10. Thực thi đoạn chương trình sau kết quả là:

```
int Tinh(int a[], int n)
{
    int S = 0, i=0;
    for( ; ; ){
        if(i>=n) break;
        if(i%2) S = S + a[i++];
    }
    return S;
}

void main()
{
    int a[] = {1, 4, 2, 3, 5, 6};
    printf(" %d", Tinh(a, 6));
}
```

a/ 13

b/ 12

c/ 8

d/ 9

BÀI 9: CHUỖI - STRING

I. Khái niệm về chuỗi

Chuỗi là tập hợp nhiều ký tự, bao gồm các ký tự là chữ cái A...Z, a...z, ký số 0...9, các ký tự đặc biệt, khoảng trắng. Chẳng hạn như họ và tên của sinh viên, địa chỉ nhà, địa chỉ email,...

Trong ngôn ngữ C, chuỗi là một mảng ký tự kết thúc bằng ký tự NULL (ký tự '\0'). Ký tự NULL này được tự động thêm vào cuối mỗi khi chuỗi được tạo ra. Nhờ ký tự NULL để nhận biết được vị trí kết thúc của một chuỗi.

II. Khai báo chuỗi

Vì chuỗi là một mảng ký tự nên khai báo chuỗi giống như cách khai báo một mảng và các phần tử của mảng là một ký tự kiểu char.

char tênbiến[độ dài của chuỗi];

hoặc

char *tênbiến; // khai báo kiểu con trỏ

Ví dụ:

char hoten[30]; // khai báo một chuỗi họ tên có độ dài tối đa 30 ký tự

char diachi[50]; // khai báo một chuỗi họ tên có độ dài tối đa 50 ký tự

hoặc

char *hoten; // khai báo kiểu con trỏ, độ dài của chuỗi phụ thuộc vào việc cấp phát bộ nhớ cho biến hoten.

char *diachi; // tương tự hoten

Lưu ý: với phép gán chuỗi thì chỉ được thực hiện đối với khai báo kiểu con trỏ mà thôi.

char *hoten; hoten = "Nguyễn Trung Sơn" // chấp nhận

char hoten[30]; hoten = "Nguyễn Sơn Trung" //Không chấp nhận

III. Nhập xuất chuỗi

Trong ngôn ngữ C nhập xuất chuỗi thực hiện bằng cách sau:

- Nhập chuỗi:

scanf("%s",&tênbiến);

hoặc

```
gets(tênbiến);
```

Lưu ý: dùng hàm scanf() sẽ không cho phép nhập khoảng trắng. Trong trường hợp muốn nhập một chuỗi có khoảng trắng thì dùng hàm gets(). Chẳng hạn như nhập họ và tên, địa chỉ,... thì dùng hàm gets() để nhập có khoảng trắng.

- Xuất chuỗi:

```
printf("%s", tênbiến);
```

hoặc

```
puts(tênbiến);
```

Lưu ý: dùng hàm puts() sau khi xuất chuỗi sẽ xuống dòng, trong khi hàm printf() muốn xuống dòng phải thêm \n.

IV. Truy xuất từng ký tự của chuỗi

Chuỗi là một mảng các ký tự, vì thế việc truy xuất các ký tự của chuỗi cũng giống như truy xuất các phần tử của một mảng.

Ví dụ: chuỗi S chứa chữ "HELLO"

Chuỗi S được mô tả như một mảng sau:

Các ký tự :	H	E	L	L	O	NULL
Vị trí :	0	1	2	3	4	

Độ dài của chuỗi S là 5, vị trí của ký tự đầu tiên là 0, vị trí của ký tự tiếp theo là 1 và vị trí cuối cùng là 4.

Như vậy để truy xuất từng ký tự trong chuỗi, chỉ cần dùng một vòng lặp và chạy từ 0 đến độ dài của S - 1.

S[0] = 'H'; S[1] = 'E'; S[2] = 'L'; S[3] = 'L'; S[4] = 'O'

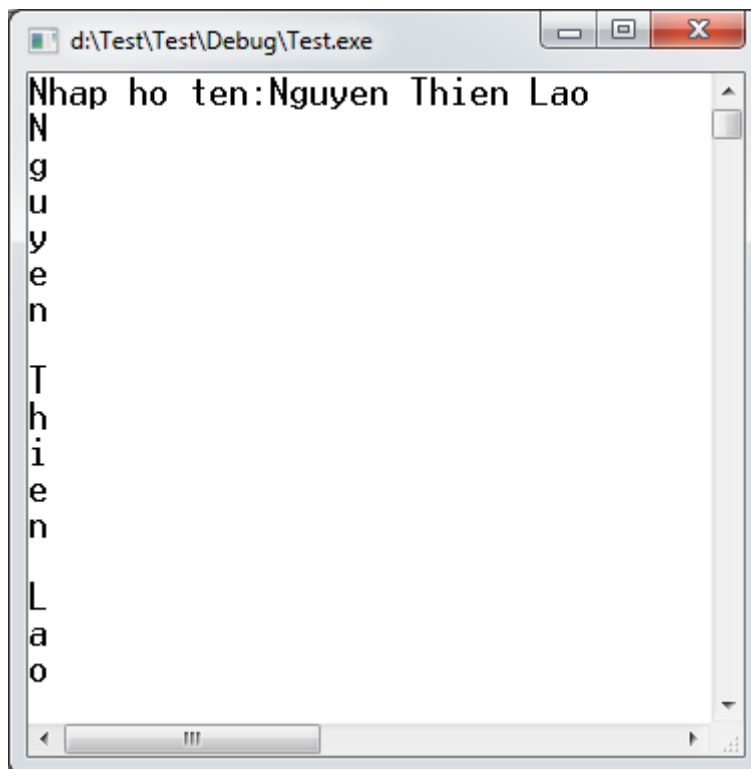
Ví dụ:

Viết chương trình nhập một chuỗi là họ tên của một người rồi in từng ký tự thành từng hàng ra màn hình.

```
#include "stdio.h"
#include "conio.h"
#include "string.h"
void main()
```

```
{
    char Hoten[50]; // Khai báo mảng ký tự Hoten có tối đa 50
    printf("Nhap ho ten:");
    gets(Hoten);
    int i=0;
    while (Hoten[i]!=NULL)
        printf("%c\n",Hoten[i++]);
    getch();
}
```

Chạy chương trình: nhập họ tên = Nguyen Thien Lao sẽ cho kết quả như sau:



Hình 8.1: Kết quả ví dụ in từng ký tự của chuỗi

V. Một số hàm xử lý chuỗi trong C

Ngôn ngữ C cung cấp một số hàm dùng sẵn chứa trong thư viện “stdlib.h” hoặc string.h, các hàm ký tự chứa trong thư viện “ctype.h”. Tên các hàm bắt đầu bằng 3 chữ cái str (viết tắt của string), 3 chữ cái tiếp theo là chữ viết tắt của ý nghĩa hàm đó. Sau đây là một số hàm phổ biến:

- **Hàm strlen(S)**
 - Hàm này cho biết độ dài của chuỗi S có bao nhiêu ký tự
 - Hàm này trả về một giá trị kiểu int
 - Ví dụ:

S = “Hoa Lan”

int L = strlen(S) → L = 7

- **Hàm strcmp(S1,S2)**

- Hàm này so sánh chuỗi S1 với chuỗi S2
- Nếu kết quả của hàm:
 - Bằng 0 thì S1 = S2
 - Lớn hơn 0 thì S1 > S2
 - Nhỏ hơn 0 thì S1 < S2
- Ví dụ:

S1 = “Hoa Lan”, S2 = “Hoa Sen”

int N = strcmp(S1,S2) → N < 0 → S1 < S2

- **Hàm strcpy(đích, nguồn)**

- Hàm này copy chuỗi nguồn vào chuỗi đích
- Ví dụ:

strcpy(S,”Nguyen Thi Be”) → S = “Nguyen Thi Be”

- **Hàm strcat(S1,S2)**

- Hàm này nối chuỗi S2 vào sau chuỗi S1
- Ví dụ:

S1 = “Em be”; S2 = “ thích dua”;

strcat(S1,S2) → S1 = “Em be thích dua”

- **Hàm atoi(S)**

- Hàm này đổi chuỗi S thành số nguyên (kiểu int)
- Ví dụ:

S1 = “1234”;

int i = atoi(S1) → i = 1234

S2 = “1234 mot hai ba”;

int k = atoi(S2) → k = 1234

S3 = “mot hai ba 123”;

int j = atoi(S3) → j = 0

- **Hàm atol(S)**

- Hàm này đổi chuỗi S thành số nguyên (kiểu long)
- Ví dụ:

S1 = “123456”;

int L1 = atol(S1) → L1 = 123456

S2 = “1234567 mot hai ba bon nam sau”;
int L2 = atoi(S2) → L2 = 1234567

S3 = “mot hai ba bon nam sau bay 1234567”;
int L3 = atoi(S3) → L3 = 0

- **Hàm atof(S)**

- Hàm này đổi chuỗi S thành số thực
- Ví dụ:
S1 = “1234.03”;
double R1 = atof(S1) → R1 = 1234.02

S2 = “1234.03 dong viet nam”;
double R2 = atof(S2) → R2 = 1234.02

S3 = “toi co 1234.03 dong viet nam”;
double R3 = atof(S3) → R3 = 0.0

- **Hàm toupper(ch)**

- Hàm này đổi ký tự ch thành chữ hoa
- Ví dụ:
char c = toupper('a') → c = 'A'

- **Hàm isupper(ch)**

- Hàm này kiểm tra ký tự ch có phải chữ hoa không.
- Ví dụ:
int kt = isupper('A') → kt = 1 //Không phải là chữ hoa
int kt = isupper('b') → kt = 0 //Đúng là chữ hoa

- **Hàm tolower(ch)**

- Hàm này đổi ký tự ch thành chữ thường
- Ví dụ:
char c = tolower('A') → c = 'a'

- **Hàm islower(ch)**

- Hàm này kiểm tra ký tự ch có phải chữ thường không.
- Ví dụ:
int kt = islower('A') → kt = 0 //Không phải là chữ thường
int kt = islower('b') → kt = 1 //Đúng là chữ thường

- **Hàm isalpha(ch)**

- Hàm này kiểm tra ký tự ch có phải là chữ cái không (a...z hoặc A...Z)
- Ví dụ:
int kt = isalpha('H') → kt = 1 //Đúng là chữ cái

int kt = isupper('/') → kt=0 //Không phải là chữ cái

- **Hàm isdigit(ch)**

- Hàm này kiểm tra ký tự ch có phải là ký tự số hay không (từ 0...9)

- Ví dụ:

int kt = isdigit('6') → kt=1 //Đúng là ký tự số

int kt = isdigit('H') → kt=0 //Không phải là ký tự số

VI. Một số ví dụ minh họa về xử lý chuỗi

- Ví dụ 1:

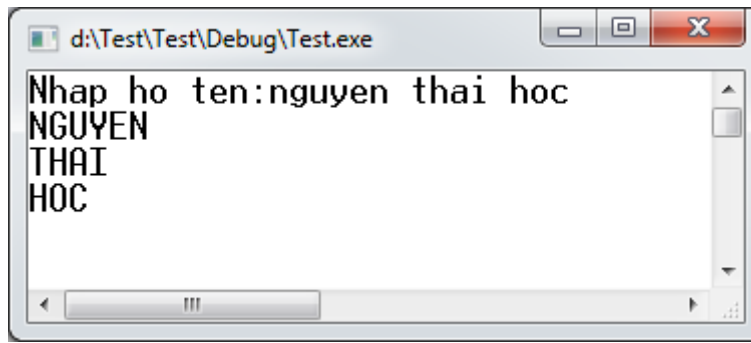
Nhập họ tên của một người, xuất mỗi chữ thành một hàng đồng thời đổi tất cả ra chữ hoa.

// Gợi ý cách giải:

- Khai báo và nhập chuỗi, sử dụng hàm gets() để nhập chuỗi
- Khai báo biến i kiểu int và khởi tạo bằng 0.
- Chạy vòng lặp để kiểm tra từ ký tự đầu tiên đến ký tự cuối cùng của chuỗi, sử dụng vòng lặp while().
- Trong vòng lặp kiểm tra từng ký tự, nếu là ký tự khoảng trắng thì xuống hàng, ngược lại thì đổi ký tự đó ra chữ hoa và in ra màn hình.
- Tăng biến i lên 1 sau mỗi vòng lặp;

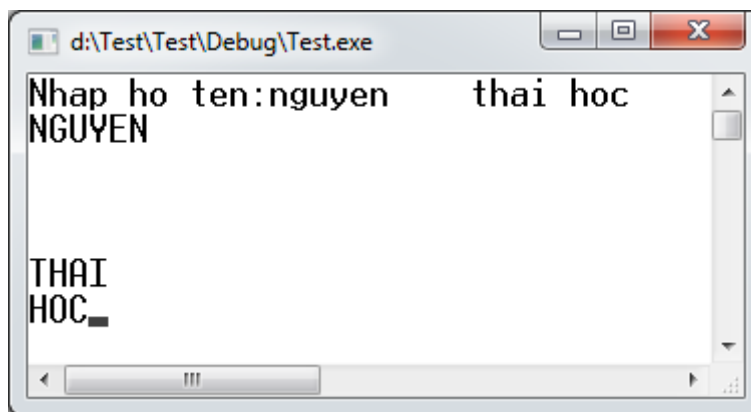
// Bài giải gợi ý viết bằng C:

```
#include "stdio.h"
#include "conio.h"
#include "ctype.h"
void main()
{
    char Hoten[50];
    printf("Nhap ho ten:");
    gets(Hoten);
    int i=0;
    while (Hoten[i]!=NULL)
    {
        if (Hoten[i] == ' ') //Kiểm tra khoảng trắng
            printf("\n"); //Khoảng trắng thì xuống hàng
        else
            printf("%c",toupper(Hoten[i]));
        i++;
    }
    getch();
}
```



Hình 8.2: Kết quả ví dụ in chuỗi ra chữ hoa

Lưu ý: khi nhập họ tên, giữa họ và họ lót có nhiều hơn một khoảng rỗng thì sẽ gặp vấn đề như kết quả sau:



Hình 8.3: Kết quả in chuỗi ra chữ hoa có nhiều khoảng khoảng trắng

Yêu cầu: sửa lại bài này để khắc phục sai sót trên

- **Ví dụ 2:**

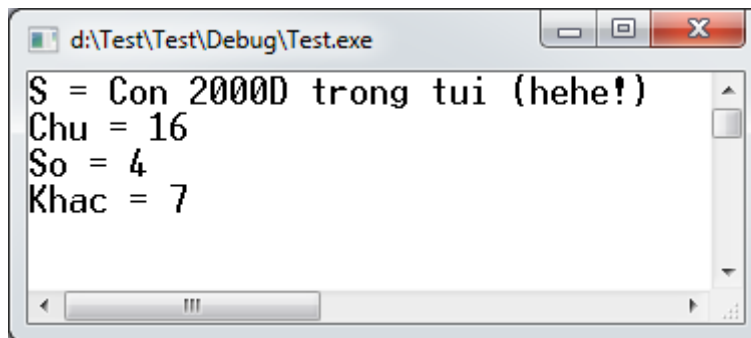
Nhập một chuỗi bất kỳ, hãy đếm xem trong chuỗi có bao nhiêu chữ cái, bao nhiêu ký tự số, và bao nhiêu ký tự còn lại:

// Gợi ý cách giải:

- Khai báo và nhập chuỗi, sử dụng hàm gets() để nhập chuỗi
- Khai báo và khởi tạo các biến i=0, Chu =0, So=0, Khac=0;
- Chạy vòng lặp để kiểm tra từ ký tự đầu tiên đến ký tự cuối cùng của chuỗi, sử dụng vòng lặp while().
- Trong vòng lặp sử dụng các hàm isalpha(), isdigit() để kiểm tra từng ký tự, nếu là ký tự là chữ thì tăng biến Chu lên 1, nếu là số thì tăng biến So lên 1, còn lại thì tăng biến Khac lên 1.
- Tăng biến i lên 1 sau mỗi vòng lặp;
- Xuất các giá trị ra màn hình (ngoài vòng lặp)

// Bài giải gợi ý viết bằng C:

```
#include "stdio.h"
#include "conio.h"
#include "ctype.h"
void main()
{
    char S[50];
    printf("S = ");
    gets(S);
    int i=0, Chu =0, So=0, Khac=0;
    while (S[i]!=NULL)
    {
        if (isalpha(S[i]))
            Chu++;
        else if (isdigit(S[i]))
            So++;
        else
            Khac++;
        i++;
    }
    printf("Chu = %d\n", Chu);
    printf("So = %d\n", So);
    printf("Khac = %d\n", Khac);
    getch();
}
```



Hình 8.4: Kết quả ví dụ đếm số ký tự chữ, số, khác

- Ví dụ 3:
Nhập họ tên của một người, in ngược tên của người đó ra màn hình theo ví dụ sau:

Hoten = “Han Mac Tu” → in ra Tu Han Mac

Hoten = “Nguyen Trinh” → in ra Trinh Nguyen

// Gợi ý cách giải:

- Khai báo và nhập chuỗi, sử dụng hàm gets() để nhập chuỗi
- Khai báo và khởi tạo các biến: L gán bằng độ dài của chuỗi, Vitri bằng vị trí của tên.
- Viết hàm xác định vị trí đầu tiên của phần tên trong chuỗi.
- Chạy vòng lặp từ Vitri của tên đến cuối chuỗi, trong vòng lặp in các ký tự của tên.
- In một ký tự khoảng trắng để cách khoảng giữa phần tên và phần họ.
- Chạy vòng lặp từ ký tự đầu tiên đến Vitri – 1, trong vòng lặp in từng ký tự ra màn hình.

// Bài giải gợi ý viết bằng C:

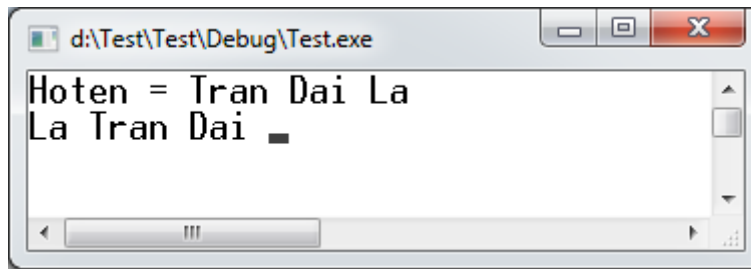
```
#include "stdio.h"
#include "conio.h"
#include "string.h"

int ViTriTen(char S[50])
{
    int Vitri = 0;
    int L = strlen(S);
    for (int i=0; i<L; i++)
        if (S[i] != ' ' & (i==0 || S[i-1] == ' '))
            Vitri = i;
    return Vitri;
}

void main()
{
    char Hoten[50];
    printf("Hoten = ");
    gets(Hoten);
    int L = strlen(Hoten); //Gán L bằng độ dài của chuỗi
    int Vitri = ViTriTen(Hoten); //Xác định vị trí tên
    for (int i=Vitri; i<L; i++) //Duyệt các ký tự của phần tên
        printf("%c", Hoten[i]); //In tên

    printf(" "); //In thêm 1 ký tự khoảng trắng
    for (int i=0; i<Vitri; i++) //Duyệt các ký tự họ và họ lót
        printf("%c", Hoten[i]); //In họ và họ lót

    getch();
}
```



Hình 8.5: Kết quả in đảo chuỗi

VII. Bài tập

- Câu 1: Viết chương trình nhập họ tên của một người rồi in ra màn hình
- Câu 2: Viết chương trình nhập vào một chuỗi, rồi in các ký tự của chuỗi ra màn hình (mỗi ký tự là 1 hàng)
- Câu 3: Viết chương trình nhập một chuỗi, hãy cho biết chuỗi đó có tất cả bao nhiêu ký tự chữ cái, bao nhiêu ký số, bao nhiêu khoảng trắng và bao nhiêu ký tự còn lại.
- Câu 4: Nhập vào một chuỗi, hãy in ngược chuỗi đã nhập
- Câu 5: Nhập họ tên sinh viên rồi in ra tên viết tắt của sinh viên đó. Ví dụ nhập : Nguyen Thi Lan Anh → NTLA
- Câu 6: Nhập họ tên của giáo viên rồi in ra địa chỉ email của giáo viên đó. Ví dụ nhập : Nguyen Van Hung → hung.nguyenvan@gmail.com.vn
- Câu 7: Viết chương trình nhập họ tên, mã SV. In ra địa chỉ email của mình như trường Đại học Hoa sen đã cấp cho mình
- Câu 8: Nhập S1, S2, kiểm tra xem chuỗi S2 có phải là tập con của chuỗi S1 không.
- Câu 9: Nhập một chuỗi bất kỳ. Hãy loại bỏ các ký tự khoảng trắng rồi sắp xếp tăng dần chuỗi đó.
- Câu 10: Hãy viết hàm chuẩn hóa một chuỗi. Nghĩa là cắt bỏ các ký tự khoảng trắng thừa ở đầu và cuối, ở giữa,...

Câu hỏi trắc nghiệm

- Câu 1: Để khai báo một chuỗi S có độ dài tối đa 100 ký tự, ta làm:

a/ `char S[100];`

b/ `char 100[S];`

c/ `char S = 100;`

d/ `char [100] S;`

Câu 2: Để biết độ dài của chuỗi S ta làm:

a/ `int L = len(S);`

b/ `int L = strlen(S)`

c/ `int L = length(S)`

d/ `int L = stringlen(S)`

Câu 3: Hãy cho biết kết quả của K là bao nhiêu khi thực hiện lệnh sau:

`int K = strcmp("ABCD", "ACBD");`

a/ Bằng không

b/ Lớn hơn không

c/ Nhỏ hơn không

d/ Giá trị NULL

Câu 4: Hàm nào sau đây đổi chuỗi thành số nguyên?

a/ `atof(...)`

b/ `atoi(...)`

c/ `toint(...)`

d/ `stringtoint(...)`

Câu 5: Muốn đổi 1 ký tự thành chữ viết hoa dùng hàm nào?

a/ `toupper(...)`

b/ `tolower(...)`

c/ `islower(...)`

d/ isupper(...)

Câu 6: Câu lệnh nào sau đây hợp lệ?

a/ char S[20] = "Hoa Sen";

b/ char *S = "Hoa Sen";

c/ Cả a và b đều hợp lệ

d/ Cả a và b đều không hợp lệ

Câu 7: Kết quả của chương trình sau là gì?

```
void main()
{
    char S[20];
    S= "Hoa Sen";
    printf(S);
}
```

a/ Hoa Sen

b/ S

c/ HoaSen

d/ Phép gán S = "Hoa Sen" không chấp nhận → lỗi chương trình

Câu 8: Kết quả của chương trình sau là gì?

```
void main()
{
    char S[4] = "ABC";
    int i=0;
    while (S[i]!=NULL)
        printf("%d",S[i++]);
}
```

a/ ABC

b/ ABCABCABC

c/ 656667

d/ Một kết quả khác

Câu 9: Hãy điền vào khoảng trống ở đoạn mã sau:

```
void main()  
{  
    char S[30] = "Nam 2014 la 2 ty dong";  
    //Đếm số ký tự là số trong chuỗi S  
    int dem=0, i=0;  
    while (S[i]!=NULL)  
        if (.....) dem++;  
    printf("%d",dem) ;  
}
```

a/ isdigit(S[++i])

b/ isdigit(S[i])

a/ isalpha([i])

b/ isdigit(S[i++])

Câu 10: Kết quả của chương trình sau là gì?

```
void main()  
{  
    char S[4] = "ABC";  
    int i=0;  
    while (S[i]!=NULL)  
        printf("%d",S[i++]);  
}
```

a/ ABC

b/ ABCABCABC

c/ 656667

d/ Một kết quả khác

BÀI 10: CON TRỎ – POINTER

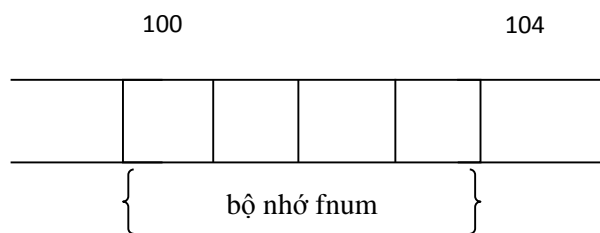
I. Khái niệm con trỏ

I.1. Địa chỉ (address)

Khi khai báo một biến như sau:

```
float fnum = 3.14;
```

Máy tính sẽ cấp phát một ô nhớ 4 byte để chứa dữ liệu cho biến fnum. Địa chỉ của ô nhớ này là một số nguyên, số thứ tự byte đầu tiên của ô nhớ chứa dữ liệu cho fnum.



Hình 10.1 Minh họa bộ nhớ biến fnum

Với hình vẽ trên thì địa chỉ của biến fnum là 100. Để lấy được địa chỉ của 1 biến ta dùng dấu & (ví dụ: &fnum)

I.2. Con trỏ (pointer)

Con trỏ là một biến chứa địa chỉ của biến khác. Có nhiều loại dữ liệu khác nhau nên kích thước các ô nhớ cũng khác nhau, nên cũng có nhiều loại con trỏ khác nhau cho các loại dữ liệu. Chẳng hạn **con trỏ int chỉ được lưu trữ địa chỉ cho biến có kiểu int, tương tự như con trỏ float, con trỏ double...**

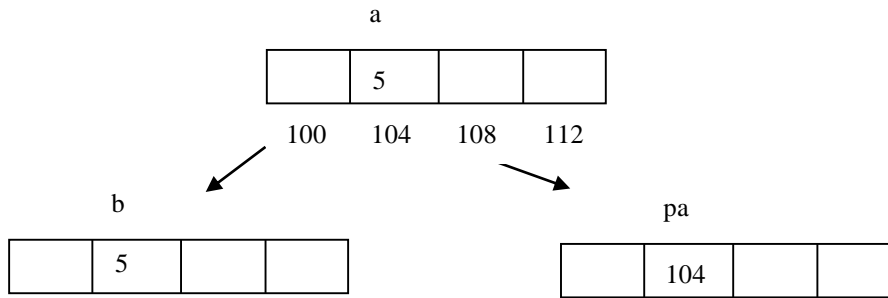
Cú pháp khai báo:

```
kiểu_dữ_liệu *Tên_con_trỏ;
```

Ví dụ:

```
char *pc;  
float *pf;  
double *pd;  
int *pa;  
int a=5;  
int b=a;
```

`pa = &a;` // *pa* là con trỏ, lưu trữ địa chỉ của biến *a*,
 // chúng ta có thể nói rằng *pa* là con trỏ chỉ đến biến *a*.



Hình 10.2: Minh họa con trỏ *pa*

- Toán tử `&`: lấy địa chỉ của biến
- Toán tử `*`: lấy giá trị của ô nhớ mà con trỏ này trỏ tới.
- Ví dụ: *pa* lưu trữ địa chỉ của *a* nên `*pa` có giá trị là 5.

```
#include <stdio.h>
void main()
{
    int a=5, b=7;
    int *pa, *pb;

    pa = &a; //con trỏ pa trỏ vào địa chỉ của biến
    pb = &b;

    *pa = 10; //gán 10 cho ô nhớ mà con trỏ pa trỏ tới
              //nghĩa là biến a có giá trị 10
    *pb = *pa;

    b = 20;
    *pb = 30;

    printf("Gia tri a la: %d\n ", a);
    printf("Gia tri b la: %d\n", b);

    printf("Dia chi của a %u, gia tri cua pa %u\n", &a, pa);
    printf("Dia chi của b %u, gia tri cua pb %u\n", &b, pb);
}
```

II. Con trỏ và mảng

Mảng có quan hệ rất gần với con trỏ. Thực ra tên mảng là một định danh (identifier) lưu trữ địa chỉ ô nhớ đầu tiên, do đó chúng có chung khái niệm.

Chẳng hạn ta có ví dụ sau:

```
int arr[10];  
int *pa;
```

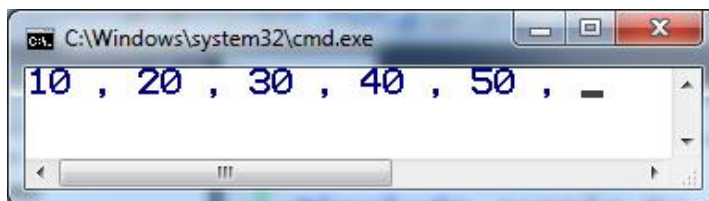
ta có thể gán:

```
pa = arr;
```

Như vậy pa và arr có cùng tính chất, ta có thể thao tác trên pa giống như mảng arr. Sự khác biệt duy nhất giữa pa và arr là giá trị pa có thể thay đổi được, nhưng arr thì không thay đổi được, nó luôn lưu trữ địa chỉ ô nhớ đầu của mảng.

Chẳng hạn ta không thể thực hiện lệnh sau: `arr = pa;`

```
#include <stdio.h>  
void main ()  
{  
    int listnum[5];  
    int * p;  
    p = listnum;  
    *p = 10; //p chỉ đến phần tử thứ 0 của mảng  
    p++;  
    *p = 20; //p chỉ đến phần tử thứ 1 của mảng  
    p = &listnum[2];  
    *p = 30; //p chỉ đến phần tử thứ 2 của mảng  
    p = listnum + 3;  
    *p = 40; //p chỉ đến phần tử thứ 3 của mảng  
    p = listnum;  
    *(p+4) = 50; //p chỉ đến phần tử thứ 4 của mảng  
    for (int i=0; i<5; i++)  
        printf("%d , ", listnum[i] );  
}
```



Hình 10.3: Kết quả chương trình minh họa dùng con trỏ truy xuất phần tử của mảng

III. Khởi tạo con trỏ

Khi khai báo con trỏ, muốn xác định ô nhớ (biến) mà con trỏ đó chỉ vào. Ta khai báo như sau:

Cách 1:

```
int num=5;  
int *pnum = &num;
```

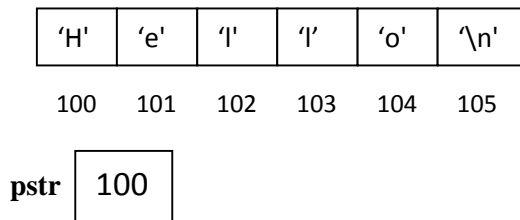
Cách 2:

```
int num=5;  
int *pnum;  
pnum = &num;
```

Cách 1 và 2 tương đương nhau.

Khai báo con trỏ như một chuỗi các ký tự: `const char *pstr = "Hello";`

Trong trường hợp này con trỏ `pstr` chỉ vào địa chỉ của ô nhớ chứa ký tự đầu tiên



`pstr` có giá trị là 100, chính là địa chỉ ô nhớ chứa ký tự 'H'.

IV. Các phép toán trên con trỏ

Các phép toán số học trên con trỏ khác với các phép toán trên số nguyên. Con trỏ chỉ thực hiện được phép cộng và phép trừ. Tuy nhiên, ngay cả phép cộng, trừ cũng khác nhau khi thực hiện trên các loại con trỏ khác nhau. Chúng ta đã biết kích thước của mỗi kiểu dữ liệu khác nhau, chẳng hạn `char` chiếm 1 byte, `int` chiếm 4 byte, `double` chiếm 8 byte bộ nhớ, nên việc tính toán trên con trỏ cũng khác nhau.

Ví dụ:

```
char * ptrChar;  
int *ptrInt;  
double *ptrDouble;
```

Thực hiện phép cộng trên các con trỏ trên.

`ptrChar++;`

`ptrInt ++;`

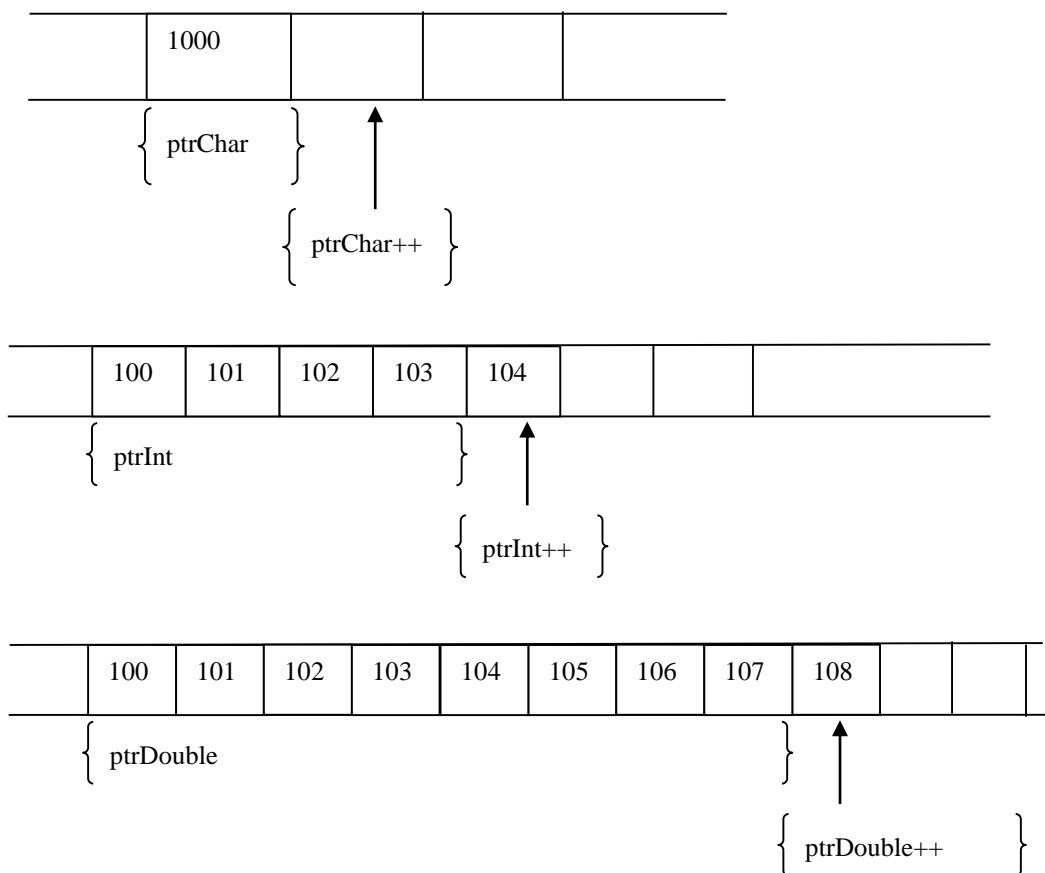
`ptrDouble++;`

hoặc

`ptrChar = ptrChar+1;`

`ptrInt = ptrInt+1;`

`ptrDouble = ptrDouble+1;`



Hình 10.5: Minh họa phép dịch chuyển con trỏ

`ptrInt` có giá trị là 100, khi cộng 1 sẽ là 104, `ptrDouble` có giá trị là 100, khi cộng 1 sẽ là 108.

V. Bộ nhớ động (Dynamic Memory)

Khi khai báo mảng, chúng ta phải xác định sẵn số lượng phần tử tối đa. Số lượng này không được thay đổi khi thực thi chương trình. Nhưng nếu cần một mảng

mà kích thước của nó được xác định khi thực thi chương trình thì chúng ta phải khai báo bộ nhớ động. Sử dụng từ khóa **new** để cấp phát bộ nhớ động như sau:

- Cấp phát một ô nhớ:
con_trỏ = new kiểu_dữ_liệu;
- Cấp phát một số các ô nhớ:
con_trỏ = new kiểu_dữ_liệu[số_phần_tử];
- Giải phóng bộ nhớ: Sau khi sử dụng xong con trỏ ta phải giải phóng bộ nhớ để máy tính dùng bộ nhớ cho các biến khác.
delete con_trỏ;
delete []con_trỏ; // giải phóng một nhóm các ô nhớ

Ví dụ:

```
int *ptr = new int;  
int *arr = new int[10];
```

arr được sử dụng giống như mảng một chiều, còn được gọi là mảng động. Mảng động giúp giảm thiểu bộ nhớ, chỉ cần cấp phát đủ bộ nhớ muốn dùng.

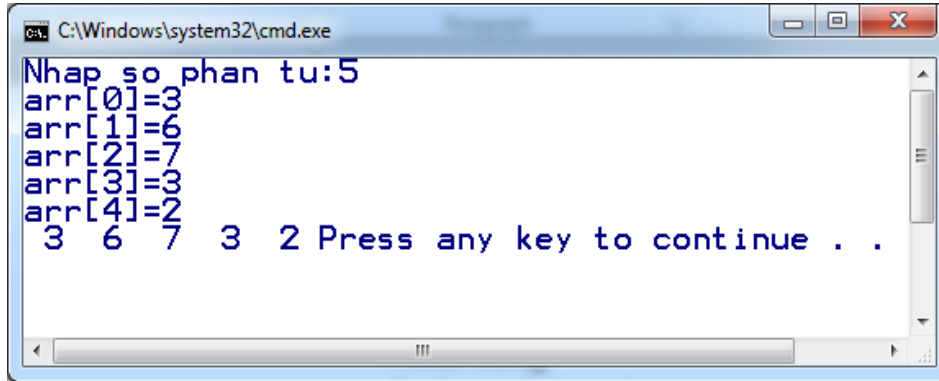
Ví dụ:

Nhập mảng các số nguyên. Xuất mảng ra màn hình (dùng mảng động)

```
#include <stdio.h>  
void main()  
{  
    int *arr, n;  
    do{  
        printf("Nhap so phan tu:"); scanf("%d", &n);  
    }while(n<=0 || n>100); // n trong khoảng 1 đến 100  
  
    //cấp phát bộ nhớ động  
    arr = new int[n];  
  
    //nhập các số nguyên vào mảng  
    for(int i=0; i<n; i++)  
    {  
        printf("arr[%d]=", i);  
        scanf("%d", &arr[i]);  
    }  
}
```



```
//xuất mảng
for(int i=0; i<n; i++)
{
    printf(" %d ", arr[i]);
}
}
```



Hình 10.6: Kết quả chương trình nhập xuất với mảng động

VI. Bài tập minh họa

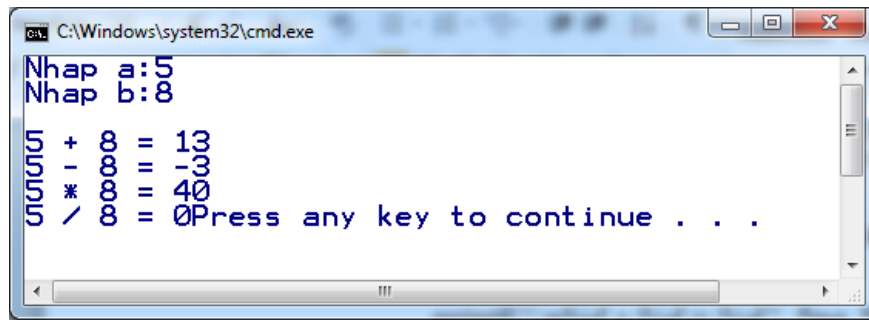
Bài 1: Viết chương trình khai báo 2 con trỏ số nguyên a, b. Xuất ra màn hình tổng, hiệu, tích thương.

```
#include <stdio.h>
void main()
{
    int *pa = new int;
    int *pb = new int;

    printf("Nhap a:");
    scanf("%d", pa); // lưu ý không có dấu &

    printf("Nhap b:");
    scanf("%d", pb); // lưu ý không có dấu &

    printf("\n%d + %d = %d", *pa, *pb, (*pa)+(*pb));
    printf("\n%d - %d = %d", *pa, *pb, (*pa)-(*pb));
    printf("\n%d * %d = %d", *pa, *pb, (*pa)*(*pb));
    if( (*pb)!=0)
        printf("\n%d /%d = %d", *pa, *pb, (*pa)/(*pb));
}
```



Hình 10.7: Kết quả chương trình minh họa dùng con trỏ truy xuất giá trị của biến

Bài 2: Viết chương trình nhập mảng các số nguyên. Tính và xuất ra màn hình tổng mảng.

Cách 1: truy xuất phần tử mảng thông qua chỉ số

```
#include <stdio.h>
void main()
{
    int *arr = new int[5];
    //nhập các số nguyên vào mảng
    for(int i=0; i<5; i++)
    {
        printf("arr[%d]=", i);
        scanf("%d", &arr[i]);
    }

    //xuất mảng
    for(int i=0; i<5; i++)
    {
        printf(" %d ", arr[i]);
    }
}
```

Cách 2: truy xuất phần tử mảng thông qua con trỏ

```
#include <stdio.h>
void main()
{
    int *arr = new int[5];
    //nhập các số nguyên vào mảng.
    for(int i=0; i<5; i++)
    {
```

```
        printf("arr[%d]=", i);
        scanf("%d", arr+i);
    }

    //xuất mảng
    for(int i=0; i<5; i++)
    {
        printf(" %d ", *(arr+i));
    }
}
```

- Giải thích: *arr* là con trỏ chỉ đến phần tử đầu tiên của mảng, *arr+1* là con trỏ chỉ đến phần tử kế tiếp của mảng..., **arr* là giá trị phần tử đầu tiên, **(arr+1)* là giá trị phần tử kế tiếp.

VII. Bài tập tự làm

- Bài 1: Viết chương trình nhập 2 số nguyên a, b là hệ số phương trình $ax+b=0$. Giải phương trình tìm nghiệm x. Dùng con trỏ truy xuất đến 2 số nguyên.
- Bài 2: Viết chương trình nhập mảng các số nguyên, tính tổng các số nguyên trong mảng. Dùng con trỏ truy xuất đến các phần tử của mảng
- Bài 3: Viết chương trình tạo một mảng động có N phần tử số nguyên ngẫu nhiên. Các phần tử nằm trong khoảng từ -100 đến 100. Tìm vị trí của 2 phần tử liên nhau có tổng giá trị tuyệt đối nhỏ nhất.
- Bài 4: Viết chương trình tạo một mảng động có N phần tử số nguyên ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1 đến 1000. Tính tổng các phần tử là số nguyên tố.
- Bài 5: Viết chương trình tạo một mảng động có 1000 phần tử số thực ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1.0 đến 5.0. Hãy tạo ra một mảng khác và kiểm tra mảng mới này có phải là mảng con của mảng trên không.
- Bài 6: Cho mảng 2 chiều động MT có r dòng và c cột. Viết các hàm sau:
- Nhập mảng
 - Xuất mảng
 - Tính tổng dòng k của mảng

- d. Tính tổng cột k của mảng
- e. Tính tổng đường chéo chính
- f. Tìm cột có tổng lớn nhất
- g. Sắp xếp cột k tăng dần.

Bài 7: Viết chương trình tạo một mảng 2 chiều động có $M \times N$ phần tử số nguyên ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1 đến 1000. Tính tổng các phần tử nằm trên đường viền.

Bài 8: Viết chương trình tạo một mảng 2 chiều động có $M \times N$ phần tử số nguyên ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1 đến 1000. Hãy chuyển mảng từ dạng $M \times N$ sang $N \times M$ (Hàng thành cột, cột thành hàng)

Câu hỏi trắc nghiệm

Câu 1: Con trỏ là:

- a/ Một biến chứa giá trị của một biến khác
- b/ Một biến chứa phép tính trị cho một biến khác
- c/ Một biến chứa địa chỉ của một biến khác
- d/ Cả a, b, c đều đúng

Câu 2: Toán tử truy xuất địa chỉ của một biến là toán tử

- a/ &
- b/ *
- c/ new
- d/ delete

Câu 3: Cú pháp khai báo một biến con trỏ p chỉ đến kiểu double:

- a/ double b[]
- b/ double &p
- c/ double *p
- d/ cả a,b,c đều đúng

Câu 4: Toán tử truy xuất trị thông qua con trỏ là toán tử

- a/ &
- b/ *
- c/ new
- d/ delete

Câu 5: Có thể gán 2 con trỏ khác kiểu nhau

- a/ Đúng

b/ Sai

Câu 6: Có thể gán 1 số nguyên vào một con trỏ.

a/ Đúng

b/ Sai

Câu 7: Chọn các phát biểu sai

a/ Toán tử cộng có thể thao tác trên con trỏ

b/ Toán tử trừ có thể thao tác trên con trỏ

c/ Toán tử nhân có thể thao tác trên con trỏ

d/ Toán tử chia có thể thao tác trên con trỏ

Câu 8: Giả sử con trỏ p chỉ đến kiểu int, p đang chỉ đến ô nhớ 1000. p+3 sẽ chỉ đến ô nhớ

a/ 900

b/ 995

c/ 960

d/ một trị khác

Câu 9: Cách khai báo và cấp phát bộ nhớ một mảng động nào sau đây là đúng:

a/ `int *a = new int[10];`

b/ `int a* = new int[10];`

c/ `int a = new int*[10];`

d/ `int *a = new int*[10];`

Câu 10: Một mảng động khác mảng tĩnh ở chỗ nào

a/ Mảng động và mảng tĩnh không có gì khác nhau ngoại trừ cách khai báo

b/ Kích thước mảng động được xác định trước khi chương trình thực thi, còn kích thước mảng tĩnh xác định khi chương trình đang thực thi.

c/ Kích thước tĩnh được xác định trước khi chương trình thực thi, còn kích thước mảng động xác định khi chương trình đang thực thi.

d/ Cả 3 câu trên đều đúng

BÀI 11: STRUCT – KIỂU CẤU TRÚC

I. Khái niệm kiểu cấu trúc

Các bài trước đã nói về các kiểu dữ liệu cơ bản như kiểu số nguyên (int, long), kiểu ký tự (char), kiểu số thực (float, double), kiểu mảng (array),... Tuy nhiên, trong thực tế các kiểu dữ liệu này không lưu trữ được các dạng thông tin phức tạp hơn...Hãy xem ví dụ sau:

DANH SÁCH SINH VIÊN

Họ và tên	Năm sinh	Điểm trung bình
Nguyễn Thị Mân	1993	7.2
Trần Ai Châu	1992	5.9
Võ Văn Bình	1994	8.6

Nhìn vào danh sách sinh viên trên, không có kiểu dữ liệu đã học nào có thể lưu trữ được thông tin sinh viên như vậy. Vì mỗi sinh viên có 3 thông tin bao gồm họ và tên (xâu ký tự), năm sinh (kiểu số nguyên), điểm trung bình (kiểu số thực).

Trong ngôn ngữ C cung cấp cho người lập trình một kiểu dữ liệu cấu trúc (struct) cho phép người dùng gom các kiểu dữ liệu không cùng loại vào một. Kiểu struct thường được dùng để quản lý các dữ liệu dạng bảng (table), trong đó mỗi cột của table là một kiểu dữ liệu cơ bản đã học.

II. Khai báo kiểu cấu trúc (struct)

- Mẫu khai báo:

```
struct struct_name
{
    Khai báo các biến thành phần;
};
```

- Khi khai báo một kiểu struct, cần phải phân tích cấu trúc của struct đó một cách rõ ràng bao gồm:
 - o Tên struct : đặt tên như quy ước đặt tên biến, tên hàm

- Các thành phần của struct: là các thành phần dữ liệu để mô tả thông tin của một đối tượng. Chẳng hạn như sinh viên có mã số, tên, ngày sinh, điểm,...
- Luôn có cặp dấu { và }, kết thúc việc khai báo struct là dấu chấm phẩy (;). Những người mới học lập trình thường hay quên dấu chấm phẩy.

Ví dụ: khai báo một kiểu struct để quản lý sinh viên gồm các thông tin họ và tên, năm sinh, điểm trung bình, chúng ta làm như sau:

```
struct Sinhvien    //Tên struct là Sinhvien
{
    char Hoten[30]; //Khai báo họ tên kiểu chuỗi ký tự
    int Namsinh;    //Khai báo năm sinh kiểu nguyên
    float DiemTB;   //Khai báo điểm TB kiểu thực
}; // dấu chấm phẩy ";" để kết thúc khai báo struct
```

- Các biến thành phần có thể sử dụng tất cả các kiểu dữ liệu kể cả mảng (array) hay một kiểu struct khác. Xem ví dụ sau:

```
struct Ngaysinh //Tên struct là Ngaysinh
{
    int Ngay;
    int Thang;
    int Nam;
};

struct Sinhvien    //Tên struct là Sinhvien
{
    char Hoten[30]; //Khai báo họ tên kiểu chuỗi ký tự
    Ngaysinh NS; //Khai ngày sinh (NS) kiểu Ngaysinh đã định nghĩa ở trên
    float DiemTB;   //Khai báo điểm TB kiểu thực
};
```

III. Truy xuất các thành phần của struct

Cú pháp truy xuất các thành phần của struct như sau:

Tênbiếnkiểustruct.Tênbiếnthànhthầncủastruct

- Giả sử có một định nghĩa struct Sinhvien như sau:

```
struct Sinhvien
{
    char Hoten[30];
    int Namsinh;
    float DiemTB;
};
```

- Để truy xuất các biến Hoten, Namsinh, DiemTB làm như sau:

```
Sinhvien SV; //Khai báo một biến SV kiểu Sinhvien
SV.Hoten; //Truy xuất biến họ tên
SV.Namsinh; //Truy xuất biến Namsinh
SV.DiemTB; //Truy xuất biến DiemTB
```

Nếu khai báo kiểu con trỏ thì cú pháp truy xuất như sau:

Tênbiếnkiểustruckieukontro→Tênbiếnthànhthànhcủastruct

Ví dụ:

```
Sinhvien *SV; //Khai báo một biến SV kiểu Sinhvien
SV->Hoten; //Truy xuất biến họ tên
SV->Namsinh; //Truy xuất biến Namsinh
SV->DiemTB; //Truy xuất biến DiemTB
```

IV. Mảng struct

Khi quản lý sinh viên, thì thông thường là nhiều sinh viên (danh sách sinh viên). Khi quản lý hàng hóa thì thường là nhiều hàng hóa (danh sách hàng hóa),...Như vậy để quản lý một danh sách thì phải dùng mảng để quản lý. Mỗi phần tử của mảng là một biến kiểu struct do chúng ta định nghĩa.

Ví dụ 1: Để quản lý một danh sách hàng hóa, mỗi mặt hàng có các thông tin: Mã hàng, tên hàng, số lượng, đơn giá. Chúng ta khai báo như sau:

```
struct Hanghoa
{
    char Mahang[10], Tenhang[30];
```



```
float Soluong, Dongia;  
};  
//Khai báo một mảng HH có 100 phần tử kiểu Hanghoa  
Hanghoa HH[100];  
//Hoặc sử dụng con trỏ như sau  
Hanghoa *HH;
```

Ví dụ 2: Để quản lý một danh sách sinh viên, mỗi sinh viên có các thông tin: mã SV, họ tên SV, điểm TB, xếp loại. Chúng ta khai báo như sau:

```
struct Sinhvien  
{  
    char MaSV[10], Hoten[30], Xeploai[20];  
    float DiemTB;  
};  
//Khai báo một mảng SV có 100 phần tử kiểu Sinhvien  
Sinhvien SV[100];  
//Hoặc sử dụng con trỏ như sau  
Sinhvien *SV;
```

V. Các ví dụ về struct

Ví dụ 1:

Viết chương trình quản lý sách, mỗi cuốn sách có các thông tin: mã sách, tên sách, năm xuất bản, giá bán. Chương trình có các thao tác: Nhập thông tin sách, in danh sách ra màn hình mỗi cuốn một hàng.

//Phân tích:

- Tên struct: QuanLySach
- Các biến thành phần:
 - o MaSach: mã sách kiểu chuỗi có 10 ký tự
 - o TenSach: tên sách kiểu chuỗi có 50 ký tự
 - o NamXB: năm xuất bản kiểu số nguyên (int)
 - o GiaBan: Giá bán kiểu số thực (float)
- Khai báo mảng Sach[100] có 100 phần tử

- Viết hàm NhapSach()
- Viết hàm XuatSach()
- Viết hàm main() để chạy chương trình

//Bài giải gợi ý viết bằng C:

```
#include "stdio.h"
#include "conio.h"
#include "string.h"

//Khai báo struct
struct QuanLySach
{
    char MaSach[10];
    char SenSach[50];
    int NamXB;
    float GiaBan;
};

//Khai báo các hàm nguyên mẫu
void NhapSach(QuanLySach S[], int &N);
void XuatSach(QuanLySach S[], int N);

void main()
{
    int N;
    QuanLySach Sach[100]; //Khai báo mảng struct
    NhapSach(Sach,N); //gọi hàm để nhập sách
    printf("\nDanh uc sach:\n");
    XuatSach(Sach,N); //Gọi hàm để in sách ra màn hình
    getch();
}

//Xây dựng hàm nhập sách
void NhapSach(QuanLySach S[], int &N)
{
    float DG;
    printf("Nhap so luong sach :");
    scanf("%d",&N);
    for (int i=0; i<N; i++)
    {
        fflush(stdin);
        printf("Nhap cuon sach thu %d:\n",i+1);
```

```
printf("Ma sach:"); gets(S[i].MaSach);
printf("Ten sach:"); gets(S[i].SenSach);
fflush(stdin);
printf("Nam XB:"); scanf("%d",&S[i].NamXB);
printf("Don gia:"); scanf("%f",&DG);
S[i].GiaBan = DG;
//Luu ý: đối với kiểu float thì không nhập trực tiếp được trong
//mảng struct mà phải nhập qua một biến trung gian sau đó
//sử dụng phép gán
}
}
//Xây dựng hàm in danh mục sách ra màn hình
void XuatSach(QuanLySach S[], int N)
{
    for (int i=0; i<N; i++)
    {
        printf("%s\t%s\t ",S[i].MaSach,S[i].SenSach);
        printf("%d\t%6.2f\n",S[i].NamXB,S[i].GiaBan);
    }
}
```

Ví dụ 2:

Viết chương trình quản sinh viên, mỗi sinh viên có các thông tin: mã SV, họ tên SV, ngày sinh, điểm TB. Chương trình có các thao tác:

- Nhập thông sinh viên
- In danh sách ra màn hình theo dạng sau:

Mã SV	Họ tên	Ngày sinh	Điểm TB	Xếp loại
0001	Nguyen Van A	10/11/1993	6.3	Trung bình
0002	Tran Thi B	03/10/1994	9.2	Gioi
...
...

- In danh sách sinh viên có sinh năm vào năm 1994
- Tìm sinh viên theo mã số (cho phép nhập mã số, tìm nếu có thì in ra màn hình thông tin SV, ngược lại thì thông báo không tìm thấy).
- Tạo một menu có các chức năng trên để quản lý chương trình

//Phân tích:

- Tên struct: QuanLySinhVien
- Các biến thành phần:
 - o MaSV: mã sinh viên kiểu kiểu có 10 ký tự
 - o HoTen: họ tên SV kiểu chuỗi có 30 ký tự
 - o Ngày sinh: được định nghĩa thành một cấu trúc con có tên NgaySinh với các biến là: Ngay kiểu int, Thang kiểu int và Nam kiểu int
 - o DiemTB: điểm trung bình kiểu số thực (float)
 - o XeLoai: kiểu chuỗi có 20 ký tự, được tự động tính khi nhập điểm trung bình.
- Khai báo mảng SinhVien[100] có 100 phần tử
- Viết hàm NhapSV()
- Viết hàm XuatSV()
- Viết hàm TimSV(): hàm này nếu tìm thấy thì trả về vị trí tìm thấy, nếu không tìm thấy sẽ trả về giá trị âm một (-1).
- Viết hàm main() để chạy chương trình. Trong hàm main() tạo menu để chọn từng chức năng chương trình. Menu có dạng như sau:

MENU

1) Nhap sinh vien

(2) In danh sach SV

(3) In danh sach SV sinh nam 1994

(4) Tim SV

(5) Thoat

Chon chuc nang:

//Bài giải gợi ý viết bằng C:

```
#include "stdio.h"
#include "conio.h"
#include "string.h"
#include "windows.h" // chứa hàm xóa màn hình system("CLS")
//Khai báo struct
struct NgaySinh
{
    int Ngay;
    int Thang;
    int Nam;
};
struct QuanLySinhVien
{
    char MaSV[10];
```

```
char HoTen[30];
NgaySinh NS;
float DiemTB;
char XepLoai[20];
};

//Khai báo các hàm nguyên mẫu
void NhapSV(QuanLySinhVien SV[], int &N);
void XuatSV(QuanLySinhVien SV[], int N);
int TimSV(QuanLySinhVien SV[], int N, char Maso[]);

void main()
{
    int N=0, ok;
    char Maso[10];
    QuanLySinhVien SV[100]; //Khai báo mảng struct
    do
    {
        system("CLS"); //Xóa màn hình
        //Tạo menu
        printf("1.Nhap sinh vien\n");
        printf("2.In danh sach SV\n");
        printf("3.Inh danh sach SV sinh nam 1994\n");
        printf("4.Tim SV\n");
        printf("5.Thoat\n");
        printf("Chon chuc nang:");
        scanf("%d",&ok);
        switch(ok)
        {
            case 1: NhapSV(SV,N);
                    break;
            case 2: printf("Danh sach SV:\n");
                    for (int i=0; i<N; i++)
                        XuatSV(SV,i);
                    break;
            case 3: printf("Sinh vien sinh nam 1994:\n");
                    for (int i=0; i<N; i++)
                        if (SV[i].NS.Nam == 1994)
                            XuatSV(SV,i);
                    break;
            case 4: printf("Nhap ma SV can tim:");
                    fflush(stdin);
                    gets(Maso);
                    int kq = TimSV(SV,N,Maso);

```

```

        if (kq>=0)
            XuatSV(SV,kq);
        else
            printf("Khong tim thay!!!");
        break;
    }
    getch();
}
while (ok!=5);
}

//Xây dựng các hàm
void NhapSV(QuanLySinhVien SV[], int &N)
{
    float DTB;
    fflush(stdin);
    printf("Ma SV:"); gets(SV[N].MaSV);
    printf("Ho ten SV:"); gets(SV[N].HoTen);
    fflush(stdin);
    printf("Ngay sinh:\n");
    printf("\t-Ngay:"); scanf("%d",&SV[N].NS.Ngay);
    printf("\t-Thang:"); scanf("%d",&SV[N].NS.Thang);
    printf("\t-Nam:"); scanf("%d",&SV[N].NS.Nam);
    printf("Diem TB:"); scanf("%f",&DTB);
    SV[N].DiemTB = DTB;
    //Xử lý xếp loại SV
    if (DTB>=8)
        strcpy(SV[N].XepLoai,"Gioi");
    else if (DTB>=7)
        strcpy(SV[N].XepLoai,"Kha");
    else if (DTB>=5)
        strcpy(SV[N].XepLoai,"Trung Binh");
    else
        strcpy(SV[N].XepLoai,"Yeu");
    N++;
}

void XuatSV(QuanLySinhVien SV[], int i)
{
    printf("%s\t",SV[i].MaSV);
    printf("%s\t",SV[i].HoTen);
    printf("%d/%d/%d\t",SV[i].NS.Ngay,SV[i].NS.Thang,
                                                    SV[i].NS.Nam);
    printf("%.1f\t",SV[i].DiemTB);
    printf("%s\n",SV[i].XepLoai);
}

```

```
}  
int TimSV(QuanLySinhVien SV[], int N, char Maso[])  
{  
    for (int i=0; i<N; i++)  
    {  
        //Sử dụng hàm so sánh chuỗi  
        if (strcmp(SV[i].MaSV,Maso)==0)  
            return i;  
    }  
    return -1;  
}
```

VI. Bài tập

Bài 1: Viết chương trình quản lý sinh viên gồm các thông tin: mã sv, họ và tên, ngày sinh, nơi sinh, lớp học, điểm thi toán, điểm thi triết, điểm thi anh văn, điểm trung bình là điểm bình quân của các môn học, xếp loại dựa vào điểm trung bình. Chương trình cho phép thực hiện các chức năng sau:

- Nhập thông tin của sinh viên.
- In danh sách sinh viên
- Xem thông tin của 1 sinh viên (dựa vào mã SV)
- Xem thông tin trích ngang của 1 lớp bất kỳ (do người dùng nhập vào lớp học)
- Xem danh sách sv theo xếp loại (ví dụ nhập loại trung bình thì in danh sách SV xếp loại trung bình).

Bài 2: Viết chương trình quản lý bán hàng gồm các thông tin sau: mã hàng, tên hàng, đơn vị tính, số lượng, đơn giá bán, thành tiền=số lượng * đơn giá, chiết khấu = 5% * thành tiền, doanh thu = thành tiền – chiết khấu. Chương trình cho phép thực hiện các công việc sau:

- Nhập thông tin bán hàng (gồm các thông tin trên)
- In liệt kê chi tiết bán hàng
- In ra 2 mặt hàng có số lượng bán lớn nhất
- In ra 2 mặt hàng có doanh thu bán nhỏ nhất
- Sắp xếp thứ tự giảm dần theo doanh thu bán hàng
- Hãy lập bảng thống kê doanh thu bán hàng như bảng sau:

Tên hàng	Doanh thu
Mặt hàng A	?
Mặt hàng B	?
Tổng cộng	?

Bài 3: Viết chương trình tạo cấu trúc phân số có tử và mẫu. Hãy thực hiện các thao tác trên 2 phân số gồm:

- Nhập 2 phân số
- Nhân 2 phân số
- Chia 2 phân số
- Cộng 2 phân số
- Trừ 2 phân số
- Rút gọn phân số
- Tạo một menu để thực các chức năng

Câu hỏi trắc nghiệm

Câu 1: Kết thúc khai báo một struct là?

- a/ Dấu hai chấm
- b/ Dấu chấm phẩy**
- c/ Dấu chấm
- d/ Dấu ngoặc móc đóng

Câu 2: Các kiểu dữ liệu nào sau đây không được dùng để khai báo các biến thành viên trong struct?

- a/ Kiểu int
- b/ Kiểu long
- c/ Kiểu char và float
- d/ Tất cả đều sai**

Câu 3: Giả sử đã khai báo một struct có tên là HangHoa, hay cho biết câu lệnh nào sau đây là không đúng?

- a/ HangHoa HH;
- b/ HangHoa HH[100];
- c/ HangHoa *HH;
- d/ HangHoa.Ten = "Khoa Mi";**

Câu 4: Khai báo struct sau, sai ở chỗ nào?

```
STRUCT EXAM{ int x=y=1;};
```

a/ Không được nằm trên 1 hàng

b/ Chữ struct không được viết hoa

c/ int x=y=1;sai chỗ này

d/ Chữ EXAM không được viết hoa

Câu 5: struct Phanso { int tu, mau;};

Câu nào sau đây hợp lệ?

a/ PHANSO ps1;

b/ Phanso ps1.tu = 1;

c/ Phanso *ps1 = new Phanso;

d/ Phanso ps1.mau =1;

Câu 6: struct Phanso { int tu, mau;};

Để gán các biến tu và mau bằng 1 chúng ta làm như sau:

a/ Phanso PS; PS.tu = PS.mau = 1;

b/ Phanso *PS = new Phanso; PS->tu = PS->mau = 1;

c/ Phanso PS; PS.tu = 1; PS.mau = 1;

d/ Tất cả đều đúng

Câu 7: Khai báo sau sai ở chỗ nào?

```
struct Nhanvien  
{  
    char Hoten[30];  
    struct Ngaysinh{int ngay, thang, nam;};  
};
```

a/ Không được khai báo struct lồng vào trong struct

b/ Cuối dòng struct bên trong không được có dấu chấm phẩy

c/ Họ tên của nhân viên chỉ 30 ký tự là quá ngắn

d/ Tất cả a,b,c đều sai

Câu 8: Hãy điền vào chỗ trống sau:

```
Struct .....{ char Hoten[30]; int tuoi;};
```

a/ Sinh vien;

b/ Sinh-vien;

c/ Con Nguoi;

d/ ABC

Câu 9: Hãy điền vào các chỗ trống sau một cách hợp lý:

```
struct HocSinh  
{  
    ..... Hoten[30];  
    ..... Tuoi;  
    ..... DiemTB  
};
```

a/ string, int, float;

b/ char, long, double

c/ char, int, float

d/ char, float, int;

Câu 10: Hãy điền vào các chỗ trống sau một cách hợp lý:

```
struct HangHoa  
{  
    ..... Tenhang[100];  
    ..... Donvitinh[10];  
    ..... Dongia;  
    ..... Khoiluong  
    ..... Chietkhau;  
    ..... Thanh tien;  
};
```

a/ char, char, float, float, float, float

b/ char, char, int, int, int, float

c/ char, char, float, float, int , float

d/ char, int, float, float, float, float

BÀI 12: LẬP TRÌNH VỚI TẬP TIN – FILE

I. Khái niệm file

File là một tập hợp thông tin chẳng hạn như chương trình, một tập dữ liệu được tạo ra bởi một chương trình hoặc bởi người dùng. Tên file thường có 2 phần: phần tên và phần mở rộng, trong đó phần mở rộng dùng để xác định loại file. Ví dụ: abc.txt, pro.exe,...

Người ta chia ra làm hai loại file cơ bản:

- File văn bản – File text

File text lưu trữ thông tin dưới dạng ký tự với kích thước mỗi phần tử là 1 byte.

Đối với chuỗi ký tự: để lưu từ “ABC” sẽ phải mất 3 byte. Dựa vào bảng mã ASCII, ‘A’ có mã 65, ‘B’ có mã 66, ‘C’ có mã 67. Chuyển các mã này thành các mã nhị phân, mỗi mã nhị phân được lưu 8 bit. Chuỗi “ABC” thành “01000001 01000010 01000011”. Dãy nhị phân này sẽ được lưu xuống file. Khi đọc file lên máy tính sẽ làm ngược lại, đọc thành từ byte và dựa vào bảng mã ASCII để hiển thị ký tự.

Đối với số nguyên: để lưu số 12, máy tính sẽ chuyển 12 thành chuỗi ký tự “12” sau đó chuyển sang chuỗi nhị phân dựa trên mã ASCII của các ký tự. Chuỗi ký tự “12” sẽ chuyển thành “00110001 00110010”. Sau đó chuỗi nhị phân này được lưu xuống file theo từng byte.

- File nhị phân – File binary

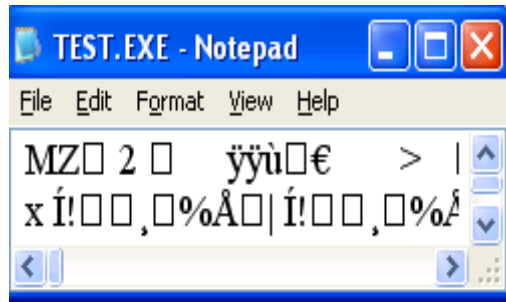
Đối với chuỗi ký tự: được lưu giống dạng text.

Đối với số nguyên: được chuyển sang mã nhị phân sau đó lưu chuỗi nhị phân này xuống file.

Chẳng hạn số 12 có mã nhị phân(4 byte) là:

0000 0000 0000 0000 0000 0000 0000 1100

Dãy nhị phân này sẽ được lưu xuống file. Khi đọc file này bằng notepad chương trình sẽ đọc từng byte và tra bảng mã ASCII để hiển thị ký tự. Điều này dẫn đến việc file đọc lên hiển thị nội dung không đúng.



Hình 12.1: Minh họa file nhị phân

II. Cấu trúc FILE

Để xử lý trên file, ta phải dùng cấu trúc FILE để lưu thông tin, nội dung file muốn đọc cũng như dùng cấu trúc này để ghi file.

typedef struct

```
{    int                level;           /* fill/empty level of buffer */
    unsigned    flags;           /* File status flags */
    char        fd;              /* File descriptor */
    unsigned char    hold;        /* Ungetc char if no buffer */
    int         bsize;           /* Buffer size */
    unsigned char    _FAR *buffer; /* Data transfer buffer */
    unsigned char    _FAR *curp; /* Current active pointer */
    unsigned         istemp;      /* Temporary file indicator */
    short            token;       /* Used for validity checking */
}    FILE;                       /* This is the FILE object */
```

Cấu trúc FILE nằm trong thư viện stdio.h

III. Thao tác trên file text – Đọc File – Ghi File

Các bước thao tác trên file

Bước 1: Khai báo biến con trỏ kiểu FILE

Bước 2: Mở file để đọc hoặc ghi

Bước 3: Đọc/ Ghi file

Bước 4: Đóng file

Các hàm xử lý file

a) Mở file

FILE * fopen (const char* path, const char* mode)

Trong đó:

path: đường dẫn file

mode: định dạng mở file (w – ghi file, r - đọc file, a – thêm nội dung)

Hàm fopen trả về một con trỏ file để quản lý việc đọc-ghi file. Nếu không mở được file, hàm sẽ trả về giá trị NULL

Ví dụ:

```
FILE * f; //khai báo con trỏ file
f = fopen("input.txt", "r"); //mở file để đọc
if (f==NULL)
{
    printf("không mở được file");
    exit(1); //thoát chương trình
}
...
...
```

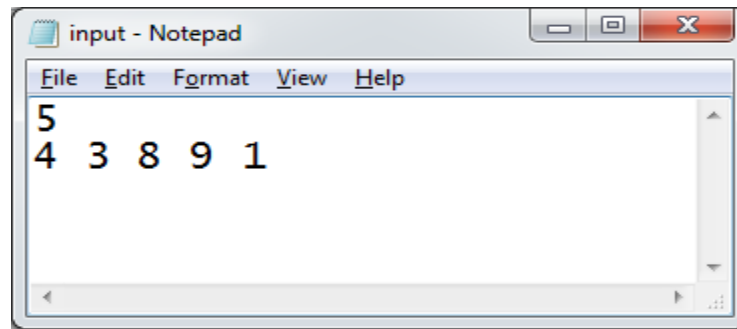
b) Đọc nội dung file

Việc đọc file cũng giống như nhập dữ liệu từ bàn phím. Dùng các hàm fgetc, fgets, fscanf

- char ch = fgetc(FILE *f) //đọc một ký tự
- fgets(char *s, int nbytes, FILE *f) // đọc một chuỗi ký tự
- fscanf(FILE *f, "Định dạng", &Biến) //đọc dữ liệu từ file

Ví dụ:

Cho file input.txt, hàng đầu tiên là một số nguyên N, hàng thứ 2 là 1 dãy N số nguyên.



Hình 12.2: File input.txt minh họa đọc file

//Đọc file input.txt, xuất nội dung ra màn hình

```
#include <stdio.h>
void main()
{
    // lưu file input.txt ở thư mục hiện hành
    FILE *f = fopen("input.txt", "r");
    if(f==NULL)
    {
        printf("không mở được file");
        exit(1); //thoát chương trình
    }
    int n, x;
    //đọc phần tử ở hàng đầu tiên
    fscanf(f, "%d", &n);

    //đọc n phần tử tiếp theo ở hàng thứ 2
    for(int i=0; i<n; i++)
    {
        fscanf(f, "%d", &x);
        printf("%d", x );
    }
    fclose(f); //đóng file
}
```

c) Ghi nội dung vào file

Tương tự như việc xuất giá trị ra màn hình. Dùng các hàm fputc, fputs, fprintf để lưu nội dung vào file.

- int fputc(char ch, FILE *f) //ghi một ký tự vào file
- fputs(char *s, FILE *f) //ghi một chuỗi ký tự vào file

- `fprintf(FILE *f, "Định dạng", biến,...)` //ghi dữ liệu vào file

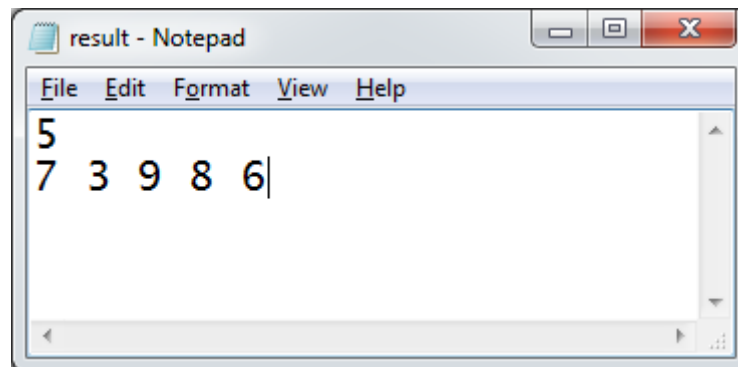
Ví dụ:

Ghi các giá trị từ mảng số nguyên ra file result.txt

```
#include <stdio.h>
void main()
{
    int a[5] = {7, 3, 9, 8, 6};
    // lưu file input.txt ở thư mục hiện hành
    FILE *f = fopen("result.txt", "w"); //tạo file để ghi
    if(f==NULL)
    {
        printf("Không mở được file");
        exit(1); //thoát chương trình
    }
    //ghi phần tử đầu tiên
    fprintf(f, "%d \n", 5);

    //ghi các phần tử trong mảng
    for(int i=0; i<5; i++)
        fprintf(f, "%d ", a[i]);

    fclose(f); //đóng file
}
```



Hình 12.3: File result.txt minh họa ghi file

Một số các hàm xử lý file khác:

- **int fclose()** : hàm dùng để đóng file, trả về -1 nếu đóng file thất bại, trả về 0 nếu đóng file thành công

- **long ftell(FILE *f):** hàm trả về số byte tính từ vị trí hiện hành của file trở về đầu.
- **void rewind(FILE * f) :** đưa con trỏ về đầu file
- **int fseek(FILE * f, long offset, int whence):** dùng để di chuyển từ 1 vị trí nào đó trong file tới 1 vị trí khác

Trong đó:

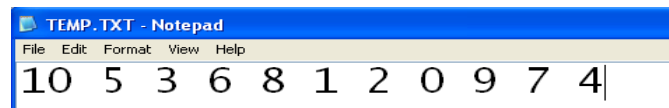
f: con trỏ file

offset: di chuyển với số byte

whence: vị trí bắt đầu di chuyển (0 – Vị trí đầu file; 1 – Vị trí hiện hành; 2–Vị trí cuối file)

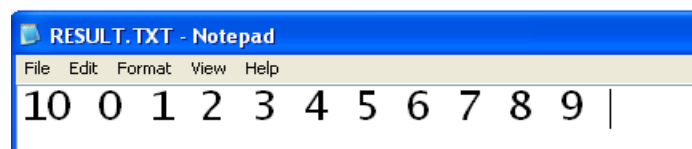
IV. Bài tập minh họa đọc-ghi file text

Tạo một file TEMP.txt có dạng sau, số đầu tiên là số lượng các số nguyên.



Hình 12.4: File temp.txt cho bài minh họa đọc-ghi file

Viết chương trình đọc file temp.txt, đưa các số nguyên vào mảng sau đó sắp xếp và ghi kết quả dãy các số nguyên đã được sắp xếp xuống file Result.txt.



Hình 12.5: File result.txt cho bài minh họa đọc ghi file

```
#include<stdio.h>
#include<conio.h>
void Read(int*&a, int &n, char* fname) {
    FILE* f= fopen(fname, "r");
    fscanf(f,"%d",&n);
    a= new int[n];
    for(int i=0; i<n; i++) fscanf(f,"%d",&a[i]);
    fclose(f);
}
void Write(int *a, int n, char* fname) {
    FILE* f=fopen(fname,"w");
```

```
fprintf(f,"%d ", n);
for(int i=0; i<n; i++) fprintf(f,"%3d", a[i]);
fclose(f);
}
void Sort(int *a, int n) {
    for(int i=0; i<n-1; i++)
        for(int j=n-1; j>i; j--)
            if(a[j]<a[j-1])
            {
                int t=a[j];
                a[j]=a[j-1];
                a[j-1]=t;
            }
}
void OutPut(int *a, int n) {
    for(int i=0; i<n; i++)
        printf("%4d", a[i]);
}
void main() {
    int *a,n;
    Read(a,n,"Temp.txt");
    Sort(a,n);
    Output(a,n);
    Write(a,n,"Result.txt");
}
```

V. Các thao tác trên file nhị phân

File nhị phân lưu các dạng dữ liệu phức tạp như mảng, cấu trúc, mảng cấu trúc

Các hàm sử dụng

fread(&Biến, sizeof(Kiểu),int n , FILE *f)

fwrite(&Biến, sizeof(Kiểu), int n, FILE *f)

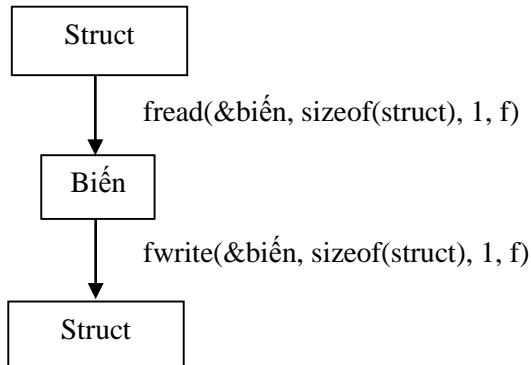
Trong đó:

- Biến: biến lưu dữ liệu cần ghi hay đọc. Biến có thể là 1 phần tử cũng có thể là một nhóm phần tử.

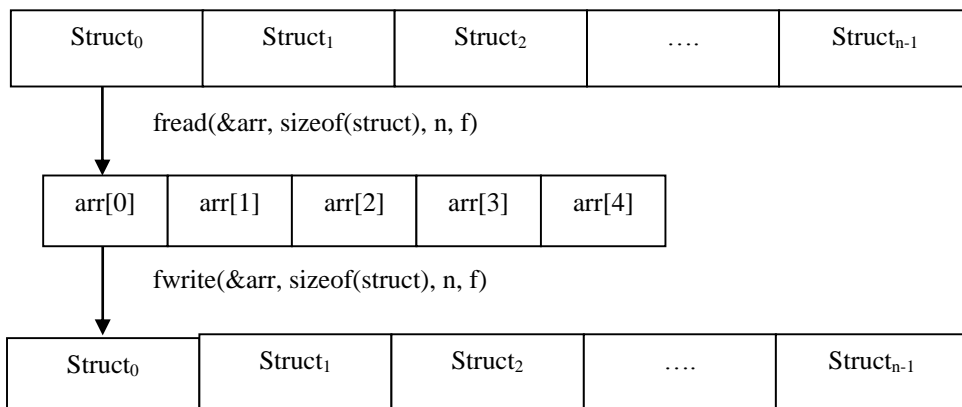
- Kiểu: kiểu dữ liệu của một phần tử

- n: số lượng phần tử

- f: con trỏ FILE



Hình 12.6: Đọc/ghi 1 phần tử



Hình 12.7: Đọc/ghi n phần tử

VI. Bài tập minh họa đọc ghi file nhị phân

Cho thông tin nhân viên < mã số (int) , tên (50 ký tự) , lương (double)>

Viết chương trình:

Nhập danh sách nhân viên từ bàn phím rồi lưu vào file “nhanvien.dat”

Xuất danh sách nhân viên trong file ra màn hình theo mã số tăng dần.

```
#include<stdio.h>
```

```
#include<conio.h>
#include<stdlib.h>
struct Employee
{
    int MaSo;
    char Ten[51];
    double Luong;
};
void Input(Employee &e)
{
    printf("Nhap ma so");
    scanf("%d", &e.MaSo);
    fflush(stdin);
    gets(e.Ten);
    double x;
    printf("Nhap luong");
    scanf("%lf", &x);
    e.Luong = x;
}
void WriteFile(char* fName)
{
    FILE* f= fopen(fName, "wb");
    if(!f) exit(0);
    Employee e;
    char ans;
    do{
        Input(e);
        fwrite(&e, sizeof(e), 1, f);
        printf("Nhap nua Y/N? :");
        ans = toupper(getche());
    }while(ans!='N');
    fclose(f);
}
void ReadFile(char* fName, Employee* &a, long &n)
{
    FILE *f = fopen(fName, "rb");
    if(!f) exit(0);
    fseek(f,0, SEEK_END);
    n = ftell(f)/sizeof(Employee); // lay so nhan vien
    a = new Employee[n];
    rewind(f); // quay ve dau file
    fread(a, sizeof(Employee), n, f); // doc file ra mang
    fclose(f);
}
```

```
void Sort(Employee *arr, int n)
{
    for(long i=0; i<n-1; i++)
        for(long j=n-1; j>1; j--)
            if(arr[j].MaSo<arr[j-1].MaSo)
            {
                Employee e = arr[j];
                arr[j] = arr[j-1];
                arr[j-1] = e;
            }
}

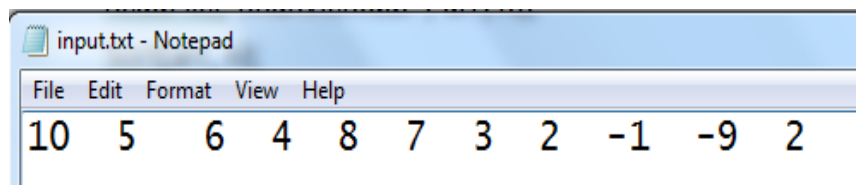
void Output(Employee *arr, long n)
{
    for(long i=0; i<n; i++)
        printf("\n%4d %-50s%.2lf",
            arr[i].MaSo, arr[i].Ten, arr[i].Luong);
}

void main()
{
    WriteFile("nhanvien.dat");
    Employee * arr ; long n ;
    ReadFile("nhanvien.dat", arr, n);
    Sort(arr, n);
    Output(arr, n);
}
```

VII. Bài tập tự làm

Bài 1: Viết chương trình nhập một dãy các số nguyên dương, sắp xếp theo thứ tự tăng dần rồi lưu vào file array.txt

Bài 2: Tạo ra một file có tên input.txt với số nguyên đầu tiên là N số nguyên, Tiếp theo là dãy N số nguyên. Ví dụ dãy số ở dưới đây, số đầu tiên là 10, và có 10 số nguyên tiếp theo.



Hình 12.8: File input.txt cho bài tập 2

Hãy viết chương trình thực hiện các chức năng sau:

- Đọc nội dung các số nguyên đưa vào mảng với số đầu tiên trong dãy là số phần tử của mảng
- Sắp xếp các phần tử.
- Tính tổng các phần tử trong mảng
- Nhập vào số nguyên x, tìm xem x có nằm trong mảng không
- Ghi nội các số nguyên đã sắp xếp xuống file result.txt

Bài 3: Tìm cặp số nguyên kế nhau lớn nhất

Yêu cầu: Cho một dãy số nguyên, hãy tìm cặp số nguyên kế nhau có tổng lớn nhất.

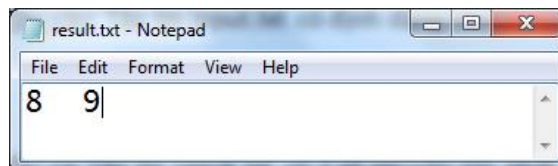
Dữ liệu đầu vào: Tập tin input.txt có định dạng như sau:

- Phần tử đầu có giá trị từ 1 đến 100, là số lượng các số nguyên
- Dãy các số nguyên tiếp theo



Hình 12.9: File input.txt cho bài tập 3

Dữ liệu đầu ra: Tập tin result.txt có 2 số nguyên lớn nhất.



Hình 12.10: File result.txt cho bài tập 3

Bài 4: Tìm hình vuông lớn nhất – Đề thi OLP

Cho một ma trận nhị phân (các phần tử trong ma trận có giá trị 0 hoặc 1)

Tìm số lượng các hình vuông thuộc ma trận trên sao cho các phần tử trong hình vuông đều có giá trị khác 0

Dữ liệu đầu vào: Từ file “SQUARE.INP” có định dạng như sau:

- + Dòng đầu tiên ghi một số NTEST là số lượng ma trận ($0 < \text{NTEST} \leq 1000$)
- + Tiếp theo là NTEST bộ dữ liệu trong đó: Dòng đầu gồm 3 số: số hàng, số cột và kích cỡ của hình vuông cần tìm (giá trị không quá 100). Các giá trị của ma trận tương ứng

Đầu ra: Ghi ra file văn bản “SQUARE.OUT”

Gồm NTEST dòng, mỗi dòng hiện số lượng hình vuông thỏa mãn yêu cầu ứng với từng ma trận và kích cỡ hình vuông đã cho.

Ví dụ:

SQUARE.INP	SQUARE.OUT
2	3
2 2 1	
0 1	4
1 1	
4 5 3	
1 1 1 1 0	
1 1 1 1 0	
1 1 1 1 0	
1 1 1 1 0	

Bảng 12.1 Dữ liệu minh họa cho bài tập 4

Bài 5: cho một dãy các số nguyên

Yêu cầu: Hãy viết chương trình cho phép sắp xếp từng nhóm k phần tử theo thứ tự tăng và k phần tử theo thứ tự giảm.

Input: File text SORT.INP có 2 dòng:

- Dòng đầu có 2 số nguyên, một số nguyên n có giá trị từ 10 đến 100 là số lượng các số nguyên cần sắp xếp, một số nguyên k là độ dài đoạn số nguyên cần sắp xếp.
- Dòng thứ 2 là dãy n số nguyên.

Output: File text SORT.OUT có 2 dòng

- Dòng đầu là n
- Dòng thứ 2 là dãy các số nguyên đã được sắp xếp theo nhóm k.

Ví dụ:

SORT.INP

11 3

9 5 6 7 4 8 10 2 3 1 13

SORT.OUT

11

5 6 9 8 7 4 2 3 10 13 1

Bảng 12.2 Dữ liệu minh họa cho bài tập 5

Bài 6: cho một ma trận vuông.

Yêu cầu: Hãy viết chương trình sắp xếp các phần tử trên đường chéo chính theo thứ tự tăng dần.

Input: File text SQUARE.INP có định dạng sau:

- Dòng đầu có số nguyên là kích thước ma trận vuông
- Các dòng tiếp theo là ma trận vuông

Output: File text SQUARE.OUT có 2 dòng

- Dòng đầu là kích thước ma trận vuông
- Các dòng tiếp theo là ma trận vuông với đường chéo chính đã được sắp xếp

Ví dụ:

SQUARE.INP

5

9 5 6 7 4

8 10 9 1 2

2 3 1 13 10

4 8 7 3 2

0 9 10 1 7

SQUARE.OUT

5

1 5 6 7 4

8 3 9 1 2

2 3 7 13 10

4 8 7 9 2

0 9 10 1 10

Bảng 12.3 Dữ liệu minh họa cho bài tập 6

Bài 7: viết chương trình tạo một chuỗi đối xứng dài nhất từ một chuỗi cho trước bằng cách loại bỏ các ký tự thừa. Chuỗi đối xứng là chuỗi cho kết quả giống nhau khi đọc xuôi hay đọc ngược.

Input: File text STRING.INP có một dòng duy nhất chứa một chuỗi toàn chữ in hoa

Output: File text STRING.OUT có 1 dòng duy nhất là chuỗi đối xứng thu được.

Ví dụ:

STRING.INP	STRING.OUT
AHOADAWOHN	HOADAOH

Bảng 12.4 Dữ liệu minh họa cho bài tập 7

Bài 8: rút gọn phân số

Input: File text PHANSO.INP

Dòng 1 chứa một số nguyên cho biết có bao nhiêu phân số

Các dòng tiếp theo chứa các phân số

Output: File text PHANSO.OUT

Dòng 1 chứa một số nguyên cho biết có bao nhiêu phân số đã được rút gọn

Các dòng tiếp theo chứa các phân số đã được rút gọn

Ví dụ:

PHANSO.INP	PHANSO.OUT
7	3
2/4 3/5 7/11 15/25 1/4 10/8 9/17	1/2 3/5 7/11 3/5 1/4 5/4 9/17

Bảng 12.5 Dữ liệu minh họa cho bài tập 8

Câu hỏi trắc nghiệm

Câu 1: Trong C, biến int n=123; khi được lưu lên file văn bản thì

a/ giá trị nhị phân của n được ghi thẳng lên file

b/ giá trị nhị phân của n được đổi ra chuỗi số rồi mới ghi lên file

- c/ ghi số 123 trực tiếp lên file
- d/ tất cả đều sai

Câu 2: Trong C, biến char S[10] = "Hello"; Khi được lưu lên file văn bản thì

- a/ giá trị nhị phân của các ký tự trong S được ghi thẳng lên file
- b/ giá trị nhị phân của các ký tự được đổi ra chuỗi số hệ 10 rồi mới ghi lên file
- c/ ghi trực tiếp từ "Hello" lên file.
- d/ tất cả a, b, c đều sai

Câu 3: Trong ngôn ngữ C, biến int n = 123; biến n được lưu trữ trong 4 byte của bộ nhớ chương trình. Khi lưu trữ n dạng file văn bản, giá trị của n sẽ lưu vớibyte(s)

- a/ 1
- b/ 2**
- c/ 3
- d/ 4

Câu 4: Trong ngôn ngữ C, biến char c = "123"; khi lưu trữ c dạng file nhị phân, giá trị của c sẽ lưu vớibyte(s)

- a/ 1
- b/ 2
- c/ 3**
- d/ 4

Câu 5: Chọn đường dẫn tên file đúng cách trong chương trình C

- a/ "D:/TEST/TEST/data.txt"
- b/ "D:\\TEST\\TEST\\data.txt"
- c/ "D://TEST//TEST//data.txt"
- d/ tất cả đều sai.**

Câu 6: Khai báo một biến file trong C

- a/ FILE f;
- b/ FILE * f;
- c/ File f;
- d File *f;**

Câu 7: Xem xét các bước xử lý file sau. Hãy cho biết thứ tự hợp lệ của các bước này

- (1) Khai báo con trỏ FILE
- (2) Mở file với tên file
- (3) Đọc dữ liệu
- (4) Đóng file

- a/ 1 2 3 4**
- b/ 2 1 3 4
- c/ 2 1 3 4

d/ 1 2 3 4

Câu 8: Xem xét các bước xử lý file sau. Hãy cho biết thứ tự hợp lệ của các bước này

- (1) Mở file với tên file
- (2) Khai báo con trỏ FILE
- (3) Ghi dữ liệu
- (4) Đóng file

a/ 1 2 3 4

b/ 2 1 3 4

c/ 2 1 3 4

d/ 1 2 3 4

Câu 9: Trường hợp nào sau đây không mở được file để ghi

- a/ File đang được mở bởi một chương trình khác
- b/ File không tồn tại
- c/ Đường dẫn file chính xác
- d/ Tất cả đều sai

Câu 10: Trường hợp nào sau đây là không mở được file để đọc

- a/ File đang tồn tại
- b/ Đường dẫn file không chính xác
- c/ kích thước file quá nhỏ
- d/ Tất cả đều đúng.

BÀI TẬP LÀM THÊM

Bài 1: Nhập N là số nguyên, hãy tính biểu thức sau:

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

Bài 2: Nhập N là số nguyên, hãy tính biểu thức sau:

$$Z = -1 + \frac{1}{-1+2} - \frac{1}{-1+2-3} + \frac{1}{-1+2-3+4} - \dots + \frac{1}{-1+2-3+4\dots+N(-1)^N}$$

Bài 3: Nhập n là số nguyên, x là số thực, hãy tính biểu thức sau:

$$Z = \frac{1+2+3+\dots+n}{!n - \sqrt[n]{x}}$$

Bài 4: Nhập n là số nguyên, hãy tính biểu thức sau:

$$Z = \frac{1+2^2+3^3+\dots+n^n}{1+\sqrt{2}+\sqrt[3]{3}+\dots+\sqrt[n]{n}}$$

Bài 5: Nhập cạnh của hình vuông, ký tự vẽ hãy vẽ các hình sau: (ví dụ cạnh =4, ký tự vẽ là dấu sao *)

```

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

```

```

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

```

Bài 6: Nhập chiều cao của tam giác vuông cân, ký tự vẽ hãy vẽ các hình sau: (ví dụ chiều cao =4, ký tự vẽ là dấu sao *)

```

*
* *
* * *
* * * *
H1

```

```

*
* *
*   *
* * * *
H2

```

```

          *
        * *
      *   *
    * * * *
H3

```

```

          *
        * *
      * * *
    * * * *
H4

```

Bài 7: Nhập chiều cao của tam giác cân, ký tự vẽ, hãy vẽ các hình sau: (ví dụ chiều cao =4, ký tự vẽ là dấu sao *)

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

H1

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

H2

Bài 8: Viết chương trình nhập một số nguyên n ($n \geq 5$), in ra các hình gồm các dấu sao như sau: Ví dụ nhập n = 5:

Hình 1

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

Hình 2:

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

Hình 3:

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

Bài 9: Viết chương trình nhập một số nguyên n ($n > 5$), in ra hình gồm các dấu sao như sau, đồng thời lưu nội hình vẽ này vào file có tên là Vehinh.txt
Ví dụ nhập n = 7:

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

Bài 10: Viết chương trình nhập một số nguyên n ($n > 5$), in ra hình gồm các dấu sao như sau: Ví dụ nhập n = 7:

```

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

```

Bài 12: Viết chương trình nhập một số nguyên n ($n > 5$), in ra hình gồm các dấu sao như sau: Ví dụ nhập $n = 7$:

```

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

```

Bài 13: Viết chương trình nhập một số nguyên n ($n > 5$), in ra hình gồm các dấu sao như sau: Ví dụ nhập $n = 7$:

```

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

```

Bài 14: Viết chương trình nhập một số nguyên n ($n > 5$), in ra hình gồm các dấu sao như sau: Ví dụ nhập $n = 5$:

```

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

```


Bài 18: Viết chương trình nhập một số nguyên n ($n > 5$), in ra các hình gồm các dấu sao như sau: Ví dụ nhập $n = 7$:

Hình 1:

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

Hình 2:

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

Bài 19: Viết chương trình tạo một mảng có N phần tử số nguyên ngẫu nhiên. Các phần tử nằm trong khoảng từ 0 đến 20. Xuất các phần tử của mảng ra màn hình. Tính tổng các phần tử của mảng. Sắp xếp mảng tăng dần.

Bài 20: Tính đa thức bậc n

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 \quad \text{với } n \text{ nguyên và mảng thực } a.$$

Bài 21: Viết chương trình tạo một mảng có N phần tử số nguyên ngẫu nhiên. Các phần tử nằm trong khoảng từ 1 đến 200. Xuất các phần tử của mảng ra màn hình. Tìm Phần tử Max, Phần tử Min. Tính tổng các phần tử chẵn, tổng các phần tử lẻ. Sắp xếp mảng giảm dần.

Bài 22: Viết chương trình tạo một mảng có N phần tử số nguyên ngẫu nhiên. Các phần tử nằm trong khoảng từ -100 đến 100. Tìm vị trí của 2 phần tử liên nhau có tổng giá trị tuyệt đối nhỏ nhất.

Bài 23: Viết chương trình tạo một mảng có N phần tử số nguyên ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1 đến 1000. Tính tổng các phần tử là số nguyên tố.

Bài 24: Viết chương trình tạo một mảng có 100 phần tử số thực ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1.0 đến 5.0. Hãy tạo ra một

mảng khác và kiểm tra mảng mới này có phải là mảng con của mảng trên không.

Bài 25: Viết chương trình tạo một mảng 2 chiều có $M \times N$ phần tử số nguyên ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1 đến 1000. Tính tổng các phần tử nằm trên đường chéo.

Bài 26: Viết chương trình tạo một mảng 2 chiều có $M \times N$ phần tử số nguyên ngẫu nhiên không trùng nhau. Các phần tử nằm trong khoảng từ 1 đến 1000. Hãy chuyển mảng từ dạng $M \times N$ sang $N \times M$ (Hàng thành cột, cột thành hàng)

Bài 27: Viết chương trình nhân 2 ma trận

Bài 28: Viết chương trình quản lý bãi giữ xe gắn máy tại Đại học Hoa Sen. Mô tả thông tin giữ xe như sau:

BienSo: Biển số xe máy – kiểu chuỗi có 4 ký tự (chứa 4 số cuối của xe)
LoaiXe: Loại xe – kiểu chuỗi có 2 ký tự (GA = Xe tay ga, SO = Xe số)
GV_SV: Giảng viên Sinh viên – kiểu chuỗi có 2 ký tự (GV = Giảng viên, SV = Sinh viên)

- Sử dụng kiểu struct để khai báo dữ liệu như mô tả trên (1đ)
- Viết hàm nhập thông tin gửi xe
- Viết hàm xuất danh sách xe
- Viết hàm tính tổng số tiền giữ xe tay ga, biết rằng xe tay ga giá 3.000đ, xe số giá 2.000đ

Bài 29: Viết chương trình quản lý sinh viên gồm các thông tin: mã sv, họ và tên, ngày sinh, nơi sinh, lớp học, điểm thi toán, điểm thi triết, điểm thi anh văn, điểm trung bình là điểm bình quân của các môn học, xếp loại dựa vào điểm trung bình. Chương trình cho phép thực hiện các chức năng sau:

- Nhập thông tin của sinh viên.
- In danh sách sinh viên
- Xem thông tin của 1 sinh viên (dựa vào mã SV)
- Xem thông tin trích ngang của 1 lớp bất kỳ
- Xem danh sách sv theo xếp loại (ví dụ nhập loại trung bình thì in danh sách SV xếp loại trung bình).

Bài 30: Viết chương trình quản lý bán hàng gồm các thông tin sau: mã hàng, tên hàng, đơn vị tính, số lượng, đơn giá bán, thành tiền = số lượng * đơn giá, chiết khấu = 5% * thành tiền, doanh thu = thành tiền – chiết khấu. Chương trình cho phép thực hiện các công việc sau:

- Nhập thông tin bán hàng (gồm các thông tin trên)
- In liệt kê chi tiết bán hàng
- In ra 2 mặt hàng có số lượng bán lớn nhất
- In ra 2 mặt hàng có doanh thu bán nhỏ nhất

- Sắp xếp thứ tự giảm dần theo doanh thu bán hàng
- Hãy lập bảng thống kê doanh thu bán hàng như bảng sau:

Tên hàng	Doanh thu
Mặt hàng A	?
Mặt hàng B	?
Tổng cộng	?

TÀI LIỆU THAM KHẢO

- [1] Phạm Văn Át. Kỹ thuật lập trình C. NXB Giao Thông Vận Tải, 2006
- [2] Java tập 1. Phương Lan, Hoàng Đức Hải. NXB Lao Động Xã Hội, 2003
- [3] Stephen Prata. C Primer Plus. Fifth Edition, Sams, 2004
- [4] Brian W.Kernighan, Dennis M.Ritchie. The C programming Language, Prentice Hall, 1988
- [5] Byron S. Gottfried. Programming with C. McGraw-Hill, 1999.
- [6] Peter Aitken, Bradley L. Jones. Teach Yourself C in 21 Days, Sams.Net