**BÀI 1- GIỚI THIỆU NỘI DUNG CHÍNH**

CHƯƠNG 1: CÁC KHÁI NHIỆM CƠ BẢN

* Tài liệu tham khảo:

+ C how to programming

+ C programming tutotial

+ Google

* Chương trình thường gồm 3 phần:

Phần 1: khai báo thư viện

Phần 2: khai báo hằng, biến

Phần 3: các hàm và phương thức

* Cách ghi chú trong C

// 🡪 ghi chú trên 1 dòng

/\* \*/ 🡪ghi chú trên nhiều dòng

-Cách Compile và chạy chương trình trên dev c++

Cách 1: F11

Cách 2: Execute\ Compile&Run

**BÀI 2: KIỂU DỮ LIỆU SỐ TRONG LẬP TRÌNH C**

Lập trình là giải quyết những bài toán.

Các bước thường là: nhập dữ liệu\ xử lý dữ liệu\ xuất dữ liệu.

CÁC KIỂU DỮ LIỆU:

1: Không có dữ liệu: void

2: Ký tự: char

3: Số nguyên:

1. Có dấu: short, int (-32k đến 32k), long (-2 tỷ đến 2 tỷ)
2. Không dấu: unsigned short, unsigned int (0 đến 64k), unsigned long (0 đến 4 tỷ)

4: Số thực: vd: 3,14 or 6,52… float, double, long double

CÁCH KHAI BÁO BIẾN:

tên\_kiểu\_dữ\_liệu tên\_biến;

hoặc tên\_kiểu\_dữ\_liệu tên\_biến= giá trị;

VD: #include <studio.h>

int main () {

unsigned short tuoi;

int so\_tien\_co = 30000;

unsigned long dan\_so\_the\_gioi = 7 000 000 000;

char c= ‘A’;

double dien\_tich;

float x1, x2; }

XUẤT DỮ LIỆU TRONG LẬP TRÌNH C

Sử dụng thư viện: stdio.h (standard input output).

Sử dụng hàm: printf (“format string’’, agument\_list).

Format string có thể là: %d (số nguyên), %c (ký tự), %s (chuỗi), %f (float).

Để hiển thị số thực ta sử dụng: %.wf trong đó w là một số tự nhiên và la số ký tự sau dấu chấm chúng ta cần, ví dụ %.2f xuất ra một số thực có phần thập phân chỉ lấy đến 2 chữ số.

Chú ý: /n xuống dòng

VD: \\ khai báo thư viện

void main () {

\\ in ra màn hình

printf ("Bai 003 - xuat du lieu ra man hinh \n");

[\\ khai](file:///\\khai) báo biến kiểu số nguyên

int x = 5;

\\xuất ra màn hình giá trị của x

printf ("Gia tri cua x la: %d", x);

[\\khai](file:///\\khai) báo biến kiểu số thực

float x1= 3.1415;

printf ( " \nGia tri x1 la: %f ", x1);

printf (" \nGia tri x1 la: %.2f", x1);

float x2= 2.2634;

float x3= 2.2876;

printf ( "\n x2= %.2f , x3=%.2f", x2 ,x3);

char c= 'A';

printf ("\nc= %c", c);}

NHẬP DỮ LIỆU TỪ BÀN PHÍM

Sử dụng thư viện stdio.h (standard input output)

Sử dụng hàm scanf để nhập dữ liệu từ bàn phím qua bảng điều khiển (console)

Cú pháp của hàm scanf:

scanf (“format string”, agument\_address\_list):

địa chỉ của biến là: & tên biến

format string có thể là: %d (số nguyên), %f (số thực), %s (chuỗi ), %c (kí tự )

VD: #include <stdio.h>

void main (){

int x;

float y;

char c;

// nhap vao ky tu

printf ("\nnhap vao ky tu:");

scanf ("%c", &c);

printf ("\ngia tri c vua nhap la: %c",c );

// nhap vao so nguyen

printf (" \nNhap vao gia tri cua x: ");

scanf ( "%d", &x );

printf ("\ngia tri cua x la: %d", x);

// nhap vao so thuc

printf (" \nnhap vao gia tri cua y: ");

scanf ("%f", &y);

print// nhap nhieu du lieu cung mot hang

float x1, x2, x3 ;

printf ("nhap vao gia tri cua x1, x2, x3:");

scanf ("%f%f%f", &x1, &x2, &x3);

printf ("\nx1=%f, x2=%f, x3=%f", x1, x2, x3);}

**CÁC PHÉP TOÁN CƠ BẢN**

Toán tử Miêu tả

+ Cộng hai toán hạng

- Trừ hai toán hạng

\* Nhân hai toán hạng

/ Chia hai toán hạng

% Chia lấy phần dư

++ Tăng giá trị toán hạng thêm một đơn vị

\_\_ Giảm giá trị toán hạng một đơn vị

Lưu ý: phép ++ và – có thể đặt trước hoặc sau toán hạng

VD: #include <stdio.h>

void main (){

float a,b;

printf ("Nhap vao gia tri cua a= ");

scanf ("%f", &a);

printf ("\nNhap vao gia tri cua b= ");

scanf ("%f", &b);

float tong= a+b;

printf ("\n%.2f + %.2f= %.2f", a, b,tong);

float hieu= a-b;

printf ("\n%.2f - %.2f= %.2f", a, b, hieu);

float tich=a\*b;

printf ("\n %.2f \* %.2f= %.2f", a, b, tich);

float thuong=a/b;

printf ("\n%.2f /%.2f= %.2f", a, b, thuong);

int r= ((int) a) % ((int) b);

printf (“\n %.2f chia lay du %.2f=%d ”, a, b, r);

a++;

printf ("\na++= %.2f", a);

b--;

printf ("\nb--=%.2f", b);}

**LT06 -** **CÁC PHÉP TOÁN ++ VÀ – TRONG C**

++ tăng lên một giá trị

-- giảm đi một giá trị

a++ 🡪 a= a+1 // giá trị của a thay đổi sau khi kết thúc câu lệnh

++a 🡪 a= a+1 // giá trị của a thay đổi trước khi kết thúc câu lệnh

b-- 🡪 b= b-1 // giá trị của b thay đổi sau khi kết thúc câu lệnh ư

++b 🡪 b= b-1 // giá trị của b thay đổi trước khi kết thúc câu lệnh

#include <stdio.h>

void main (){

int a=5 ;

printf ("%d", a++);

printf ("\n%d", a);

printf ("\n%d", ++a);

int b=7;

printf ("\n%d", b--);

printf ("\n%d", b);

printf ("\n%d", --b);}

**LT07 - CÁC TOÁN TỬ SO SÁNH TRONG LẬP TRÌNH C**

Toán tử ý nghĩa ví dụ

== so sánh bằng 4=3 cho kết quả là False

> so sánh lớn hơn 4>3 cho kết quả là True

< so sánh nhỏ hơn 6<5 cho kết quả là false

!= so sánh khác 6!=9 cho kết quà là True

>= so sánh lớn hơn hoặc bằng 5>=3 cho kết quả là True

<= so sánh bé hơn hoặc bằng 7<=9 cho kết quả là False

Ý nghĩa của việc so sánh

0 🡪 False

1 ( hoặc khác 0) 🡪 True

VD:

#include <stdio.h>

void main (){

int a,b;

printf (" Nhap vao gia tri cua a: ");

scanf ("%d", &a);

printf (" \nNhap vao gia tri cua b: ");

scanf ("%d", &b);

printf ("\n%d == %d là %d", a,b, a==b);

printf ("\n%d > %d là %d", a, b, a>b);

printf ("\n%d < %d là %d", a, b, a<b);

printf ("\n%d >= %d là %d", a, b, a>=b);

printf ("\n%d <= %d là %d", a, b, a<=b);

printf ("\n%d != %d là %d", a, b, a!=b);}

**LT08 -TOÁN TỬ ĐIỀU LIỆN TRONG LẬP TRÌNH C**

Cấu trúc:

Bieu\_thuc\_1? bieu\_thuc\_2: bieu\_thuc\_3;

ở đây bieu\_thuc\_1, bieu\_thuc\_2 ,bieu\_thuc\_3 là các biểu thức.

Kết quả được quyết định như sau:

+ Nếu bieu\_thuc\_1 là True thì bieu\_thuc\_2 được tính toán và trở thành giá trị của toàn bộ biểu thức.

+ Nếu bieu\_thuc\_1 là False thì bieu\_thuc\_3 được tính toán và trở thành giá trị của toàn bộ biểu thức.

VD: x= (5>=3)?1:0; 🡪 x=1

#include <stdio.h>

void main(){

int a;

printf ("Nhap vao gia tri cua a= ");

scanf ("%d",&a);

printf("%d la %s",a,(a%2==0)?"CHAN":"LE");

}

**LT09 – BT TÌM MIN MAX GIỮA HAI SỐ a VÀ b**

Nhập vào màn hình hai số nguyên a và b thông qua cửa sổ console.

Hãy xuất ra màn hình giá trị nhỏ nhất và giá trị lớn nhất giữa hai số a và b.

#include <stdio.h>

void main(){

int a,b;

printf ("Nhap vao so a:");

scanf ("%d",&a);

printf ("Nhap vao so b:");

scanf ("%d",&b);

printf ("\nMIN là %d",(a<=b)?a:b);

printf ("\nMAX là %d",(a>=b)?a:b);

}

**LT10 – GIẢI PHƯƠNG TRÌNH ax+b=0**

**LT11 – ÉP KIỂU GIỮ LIỆU TRONG C**

**Cú pháp:** **(type) value;**

**Loại 1: Nới rộng (widening)**

**Short 🡪 int 🡪 long 🡪 float 🡪 double**

Là quá trình làm tròn số từ kiểu dữ liệu có kích thước nhỏ hơn sang kiểu dữ liệu có kích thước lớn hơn. Kiểu biến đổi này không làm mất thông tin . Ví dụ chuyển từ int sang float. Chuyển kiểu loại này có thể thực hiện ngầm định bởi trình biên dịch.

Ví dụ: **long** x= 5; // khong yeu cau chi dinh ep kieu

**float** y= x; // khong yeu cau chi dinh ep kieu

**Loại 2: Thu hẹp (narrowwing)**

**Double🡪 float🡪 long🡪 int🡪 short**

Là quá trình làm tròn số từ kiểu dữ liệu có kích thước lớn hơn sang kiểu dữ liệu có kích thước nhỏ hơn. Kiểu biến đổi này có thể làm mất thông tin. Chuyển kiểu loại này không thể thực hiện ngầm định bởi trình biên dịch, người dùng phải thực hiện chuyển kiểu tường trình

Ví dụ: **float** x = 5.3; // yeu cau chi dinh ep kieu

**long** y = **(long)x**; //yeu cau chi dinh ep kieu

**LT13 - CÁC HÀM TOÁN HỌC TRONG LẬP TRÌNH C**

Thư viện **“math.h”**

1. **ceil(number )** làm tròn lên một số đã cho. Nó trả về giá trị số

nguyên lớn hơn hoặc bằng với số đã cho.

2. **floor(number )** làm tròn xuống một số đã cho. Nó trả về giá trị

số nguyên nhỏ hơn hoặc bằng số đã cho.

3. **sqrt(number )** trả về căn bậc hai của số đã cho.

4. **pow(base, exponent )** trả về số mũ của số đã cho

5. **abs(number )**  trả về giá trị tuyệt đối của số đã cho

**CÂU LỆNH ĐIỀU KIỆN - CÂU LỆNH IF ELSE TRONG C**

**Cấu trúc 1:**

**if** ( điều kiện ) {

// đoạn code được thực hiện nếu điều kiện đúng

}

**Cấu trúc 2:**

**if**  (điều kiện) {

//đoạn code được thực hiện nếu điều kiện đúng

} **else** {

// đoạn code được thực hiện nếu điều kiện sai

}

**Cấu trúc 3:**

**if** (điều\_kiện\_1) {

//đoạn code được thực hiện nếu điều kiện 1 đúng

} **else if** (điều\_kiện\_2) {

//đoạn code được thực hiện neeuss điều kiện 1 sai và điều kiện 2 đúng

} **else** {

//đoạn code được thực hiện nếu điều kiện 1 và 2 đều sai

}

Ví dụ: Nhập vào một con số và cho biết nó có nhỏ hơn 20 hay không?

Nhập vào một con số và xuất ra nó là số chẵn, số lẻ?

Nhập vào giờ và cho biết nó là ngày hay đêm?

**LT 14 - GIẢI PHƯƠNG TRÌNH BẬC NHẤT ax+b=0**

Số a được phép bằng 0

Hướng dẫn : bài toán gồm 2 TH

+TH1: a=0 Nếu b != 0 🡪 phương trình vô nghiệm

Nếu b=0 🡪 phương trình vô số nghiệm

+TH2: a# 0 🡪 x= -b/a

**LT15 – TOÁN TỬ LOGIC TRONG LẬP TRÌNH C**

**Toán tử logic là một toán tử trong C. Toán tử logic bao gồm các toán tử sau:**

Toán tử && 🡪 là toán tử AND, trả về true khi và chỉ khi tất cả các toán hạng đều đúng.

Toán tử || 🡪 là toán tử OR, trả về true khi có ít nhất một toán hạng đúng.

Toán tử ! 🡪 là toán tử NOT, phủ định giá trị của toán hạng.

**LT16 – GIAI PHUONG TRINH BAC HAI ax^2 +bx +c =0**

#include <stdio.h>

#include <math.h>

void main (){

// giai phuong trinh bac hai ax^2 +bx +c= 0

float a,b,c,x1,x2,delta;

printf ("Nhap vao he so a,b,c cua phuong trinh ");

scanf ("%f %f %f", &a, &b, &c);

// xu ly

if (a==0) { printf ("a phai khac 0");}

else { delta = b\*b - 4\*a\*c ;

if (delta > 0) { x1= (-b-sqrt(delta))/(2\*a);

x2= (-b+sqrt(delta))/(2\*a);

printf ("phuong trinh co hai nghiem la x1 = %.2f va x2 = %.2f", x1,x2);}

else if ( delta = 0) { printf ("phuong trinh co nghiem kep x= %.2f", -b/(2\*a));

} else { printf ("Phuong trinh vo nghiem");}}

}

**LT17 - CÂU LỆNH SWICH CASE (rẽ nhánh) TRONG LT C**

Cấu trúc:

swich (case\_value) {

case 1: //excute your code

break;

case n: //excute your code

break;

default: // excute your code // trái lại

break;

}

Chú ý:

1. Biểu thức trong câu lệnh switch phải là kiểu dữ liệu số nguyên, ký tự.
2. case\_valua phải là hằng số nguyên hoặc ký tự.
3. case\_valua có thể sử dụng trong lệnh switch.
4. *Lệnh break trong switch không bắt buộc*

**Lệnh break:**

**Câu lệnh break trong C có hai cách sử dụng như sau:**

Khi gặp câu lệnh break trong một vòng lặp, vòng lặp bị kết thúc ngay lập tức và câu lệnh kế tiếp trong vòng lặp được thực thi.

Lệnh break có thể được sử dụng để kết thúc một case trong câu lệnh switch.

**Bài tập:**

Nhập vào một số từ 1 🡪 7 và hiển thị của thứ tương ứng trong một tuần ( Chủ nhật 🡪 thứ bảy )

**LT18 – BT TÌM SỐ NGÀY CỦA THÁNG DÙNG SWITCH CASE**

**Bài toán:**

Viết chương trình nhập tháng năm.

Hãy tìm só ngày trong tháng đó và xuất ra màn hình.

**Lưu ý :**

Tháng 2 của năm nhuận có 29 ngày.

Tháng 2 của năm không nhuận có 28 ngày.

Các tháng 1 3 5 7 8 10 12 có 31 ngày.

Các tháng 4 6 9 11 có 30 ngày

**Cách kiểm tra năm nhuận:**

1: Chia hết cho 400.

2: chia hết cho 4 nhưng không chia hết cho 100.

/\*Bài toán:

Viet chuong trình nhap tháng nam.

Hãy tìm só ngày trong tháng dó và xuat ra màn hình \*/

#include <stdio.h>

void main () {

int a,b ;

printf ("Nhap thang nam can tim ");

printf ("\nthang = ”);

scanf ("%d", &a);

printf ("\nam =");

scanf ("%d", &b);

switch (a) {

case 1:

case 3:

case 5:

case 7:

case 8:

case 10:

case 12:

printf ("\nCo 31 ngay");

break;

case 4:

case 6:

case 9:

case 11:

printf ("\nCo 30 ngay");

break;

case 2: if (b%400==0 || (b%4==0 && b%100!=0)) {

printf ("\nCo 29 ngay");

} else printf ("\nCo 28 ngay");

break; }}

LT19 – TÍNH CHU VI VÀ DIỆN TÍCH TAM GIÁC TRONG LT C

Bài 1: Viết chương trình nhập vào tọa độ 3 đỉnh của tam giác ABC. Hãy thực hiện các yêu cầu sau đây

1. Kiểm tra 3 điểm A,B,C có tạo thành một tam giác hay không

Nếu ABC tạo thành một tam giác tiếp tục thực hiện các yêu cầu dưới đây:

1. Kiểm tra tam giác ABC có cân hay không?
2. Tính và xuất ra màn hình chu vi tam giác ABC
3. Tinh và xuất ra màn hình diện tích tam giác ABC

#include <stdio.h>

#include <math.h>

void main (){

//khai bao

float xA,yA,xB,yB,xC,yC;

float AB, AC, BC;

float chuvi, nuachuvi, dientich;

//Nhap du lieu

printf ("Nhap vao toa do dinh A ");

scanf ("%f,%f",&xA,&yA);

printf ("\nNhap vao toa do dinh B ");

scanf ("%f,%f",&xB,&yB);

printf ("\nNhap vao toa do dinh C ");

scanf ("%f,%f",&xC, &yC);

// Xu ly

AB = sqrt(pow(xA-xB,2)+ pow(yA-yB,2));

AC = sqrt(pow(xA-xC,2)+ pow(yA-yC,2));

BC = sqrt(pow(xB-xC,2)+ pow(yB-yC,2));

printf ("\nAB=%.2f",AB);

printf ("\nAC=%.2f",AC);

printf ("\nBC=%.2f",BC);

if ((AB+AC)>BC && (AB+BC)>AC && (AC+BC)>AB){ printf ("\nABC co tao tanh mot tam giac");

} else {printf ("\nABC khong tao thanh tam giac");

}

//kiem tra xem ABC co can hay khong?

if ((AB=AC)||(AB=BC)||(AC=BC)) {printf("\nABC la tam giac can");

} else { printf ("\nABC khong la tam giac can");

}

//tinh chu vi tam giac ABC

chuvi= AB+AC+BC;

printf ("\nChu vi =%.2f ", chuvi);

nuachuvi = chuvi/2;

dientich = sqrt(nuachuvi\*(nuachuvi-AB)\*(nuachuvi-AC)\*(nuachuvi-BC));

printf ("\ndien tich = %.2f", dientich);

**LT 20 – CÂU LỆNH LẶP TRONG C**

**Câu lệnh: for**

**Cấu trúc:** for (expr1; expr2; expr3) { statement }

**Trong đó:**

Biểu thức 1: biểu thức khởi đầu.

Biểu thức 2: Biểu thức điều kiện – điều kiện lặp.

Biểu thức 3: bước nhảy – thường dùng với ý nghĩa là thay đổi bước nhảy.

Cả ba biểu thức này đều là tùy chọn, chúng có thể vắng mặt trong câu lệnh cụ thể nhưng các dấu chấm phẩy vẫn phải có.

Sự hoạt động của vòng lặp for:

Bước 1: Thực hện biểu thức khởi đầu – biểu thức 1.

Bước 2: Tính giá trị biểu thức 2 để xác định điều kiện lặp.

-Nếu biểu thức 2 có giá trị ‘sai’ (==0) thì ra khỏi vòng lặp.

-Ngược lại nếu biểu thức có giá trị đúng (khác 0) thì chuyển tới bước 3.

Bước 3: Thực hiện khối lệnh trong thân của for, chuyển tới bước 4.

Bước 4: thực hiện biểu thức 3, rồi quay về bước 2.

VD1: xuất ra các số từ 0 đến 9 ra màn hình

#include <stdio.h>

void main () {

int i;

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

printf ("\n%d",i);

}

int n,i;

printf ("Nhap vao gia tri cua n = ");

scanf ("%d",&n);

for (i=0;i<=n;i++) { printf ("\n%d",i);

}

}

**LT 21 – BẢNG MÃ ASCII VÀ BẢNG CHỮ CÁI ALPHABET TRONG LT C**

**Bài tập:** viết chương trình in ra bảng chữ cái alphabet

Bảng mã ASCII:

**#include <stdio.h>**

**void main () {**

**char kytu;**

**for (kytu=65; kytu<=90; kytu++) { printf ("\n%d - %c", kytu,**

**kytu);}**

**}**

**LT 22 – CÂU LỆNH LẶP TRONG C**

**Câu lệnh: while**

while (expression-điều kiện ) { statement-câu lệnh }

**Sự hoạt động của vòng lặp while:**

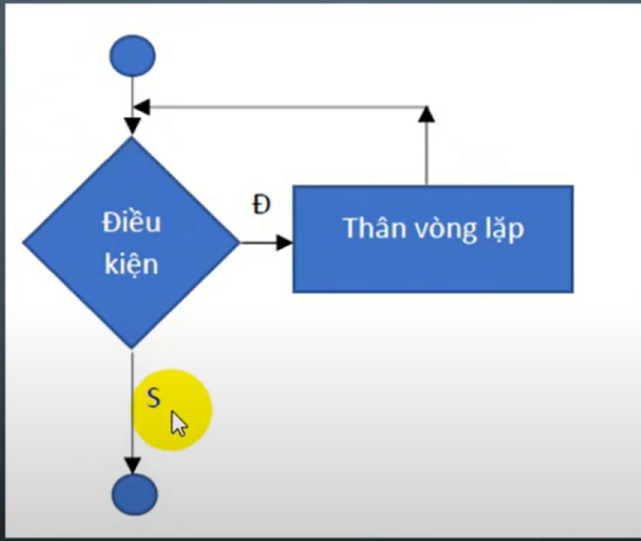
Bước 1: tính giá trị của (biểu thức) sau while.

Bước 2: nếu giá trị được tính của (biểu thức) là ‘sai’ (==0) thì kết thúc vòng lặp while.

Bước 3: nếu giá trị của (biểu thức) là ‘đúng’ (!=0) thì thực hiện khối lệnh sau while.

Bước 4: quay lại bước 1

* **Câu lệnh for sử dụng khi biết điểm bắt đầu và kết thúc**
* **Câu lệnh while sử dụng khi chưa biết điểm kết thúc**



**Ví dụ:** nhập n từ bàn phím (n>0), tính tổng các con số từ 1 đến n và xuất kết quả ra màn hình.

#include <stdio.h>

void main (){

int n,i=1, tong=0;

printf ("Nhap vao so n =");

scanf ("%d",&n);

while (i<=n) {

tong = tong + i;

i++;

}

printf ("tong =%d",tong);

}

**Ví dụ :** sử dụng vòng lặp while in ra các gái trị từ 1 đến 10

#include <stdio.h>

void main () {

int count = 0;

while (count<=10) {

printf (“\nGia tri count= %d”,count”);

count++;

}

**LT 23- TÌM ƯỚC SỐ CHUNG LỚN NHẤT GIỮA HAI SỐ**

Bài toán:

Nhập vào hai số nguyên a,b tìm ước số chung lớn nhất giữa hai số a,b và xuất kết quả ra màn hình.

Thuật toán tìm ước số chung lớn nhất

**BÀI TẬP**

ax^2+ bx +c =0

Th1: a=0

Nếu a=0, b=0, c != 0 🡪 phuong trinh vo nghiem

Nếu a=0, b=0, c= 0 🡪 phuong trinh vo so nghiem

Nếu a=0, b!=0 🡪 x= -c/b

Th2: a khac 0

Neu b!=0, c!=0 🡪

Delta = b^2 – 4\*a\*c

Neu delta>0 🡪 phuong trinh co hai nghiem phan biet la:

x1 = (-b- sqrt delta)/(2\*a)

x2= (-b+ sqrt delta)/(2\*a)

Neu delta = 0🡪 phuong trinh co nghiem kep x= -b/2a

Neu delta <0🡪 phuong trinh vo nghiem

**LAB 3\_BAI 3:**

Neu sodien>= 0 && sodien<= 50 🡪 T1=sodien \*1.678

**LT 24 – VÒNG LẶP do – while**

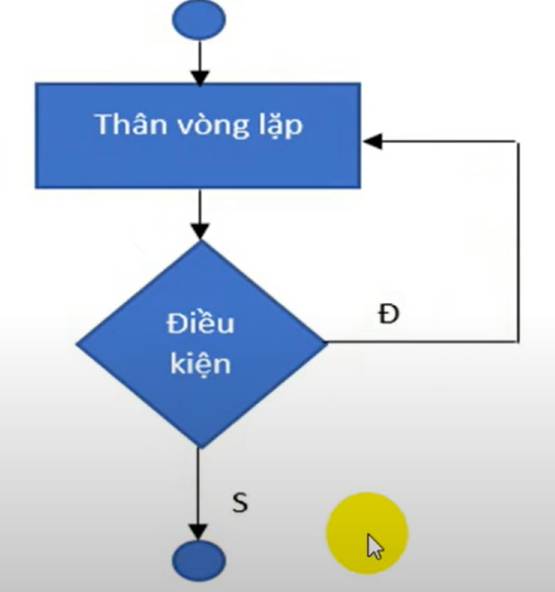
**Cấu trúc:** do {statement

} while (expresion)

Sự hoạt động của vòng lặp:

Bước 1: Thực hiện khối lệnh sau do

Bước 2: kiểm tra giá trị biểu thức <bieu\_thuc> sau while, nếu có giá trị đúng (!=0) thì lặp lại bước 1 ,nếu giá trị sai (=0) thì kết thúc vòng lặp.



**Ví dụ:** Viết chương trình hiển thị một menu sau đây lên màn hình

-Nhập sô 0 để thoát

-Nhập bất kì một ký tự nào khác để tiếp tục

#include <stdio.h>

void main () {

int n;

do {

printf (“MENU”);

printf (“\nNhap vao so 0 để thoát”);

printf (“\nNhap vao ky tu bat ki de tiep tuc”);

scanf (“%d”,&n);

}while (n!=0);

}

BÀI 1: TÍNH TRUNG BÌNH TỔNG CỦA CÁC SỐ TỰ NHIÊN CHIA HẾT CHO 2Input: nhập từ bàn phím 2 giá trị min, max  
Output: Trung bình tổng của các số tự nhiên chia hết cho 2 từ min tới max

(Đã hoàn thành)

**BÀI 2: TÍNH GIAI THỪA SỐ n (n>=0, n được nhập từ bàn phím)**

(demo25-đã ht)

do {

} while ();

**LT 26: LỆNH goto; VÀ continue;**

**LỆNH goto;**

Lệnh goto trong C cung cấp một bước nhảy vô điều kiện từ ‘goto’ dến một câu lệnh có nhãn trong cùng một hàm.

**LỆNH continue;**

Lệnh continue là lệnh mà khi gặp nó thì chương trình sẽ bỏ qua những câu lệnh phía dưới nó (trong cùng một câu lệnh lặp) để thực hiện một vòng lặp mới.

**Cú pháp:** **continue;**

**Ví dụ:**

Viết chương trình nhập vào số n (n>0), và xuất ra các số lẻ từ 1 đến n, trong đó có sử dụng lệnh **goto** và **continue;**

(demo26-ht)

**LT 27: HÀM TRONG LẬP TRÌNH C**

**-**Hàm là 1 đoạn chương trình có tên, đầu vào, đầu ra.

- Hàm có chức năng giải quyết một số vấn đề chuyên biệt cho chương trình chính.

-Hàm được gọi nhiều lần với các tham số khác nhau.

-Hàm được sử dụng khi có nhu cầu:

+Tái sử dụng.

+Sửa lỗi và cải tiến.

-Hàm có thể được xây dựng bưởi lập trình viên hoặc hàm có sẵn trong ngôn ngữ lập trình. Ví dụ trong lập trình C, chúng ta có thể liệt kê các hàm có sẵn như scanf, printf, pow, sqrt….

**Cú pháp:**

<kiểu trả về> <tên hàm> ([danh sách tham số])

{ <các câu lệnh>

[return <giá trị>;]

}

-Trong đó

+<kiểu trả về.: kiểu bất kì của c (char, int, long, float,…). Nếu không trả về thì là void.

+<tên hàm>: theo quy tắc đặt tên định danh

+<danh sách tham số>: tham số hình thức đầu vào giống khai báo biến, cách nhau bằng dấu ,

+ <giá trị> : trả về cho hàm qua lệnh return.

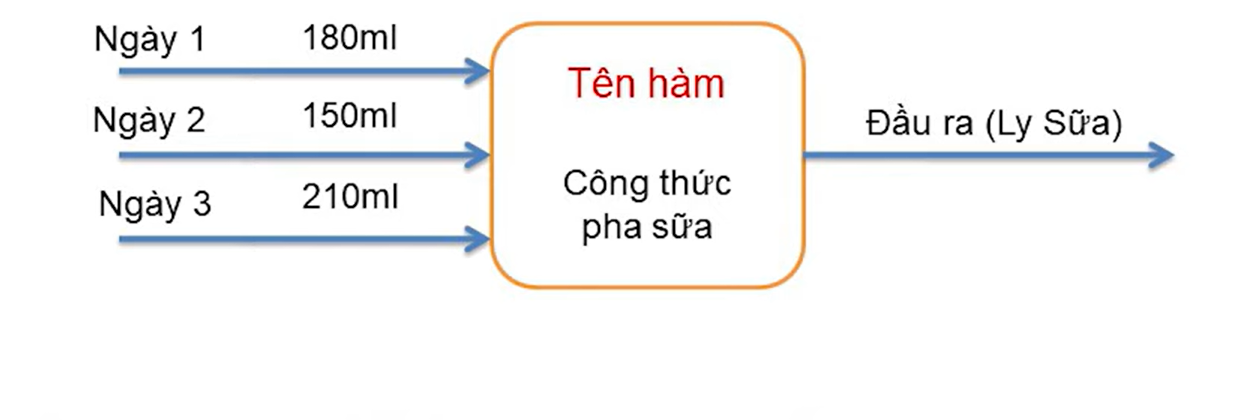
*Ví dụ:* Mẹ gọi Tí lại và bảo: Mẹ viết sẵn công thức pha sữa. Khi mẹ gọi pha sữa, con phải biết pha 1 ly sữa theo dung tích mẹ gọi cho em con uống:

-Hàm: pha sữa

- giá trị truyền vào : dung tích

- kiểu dữ liệu trả về: ly sữa

-developer: mẹ



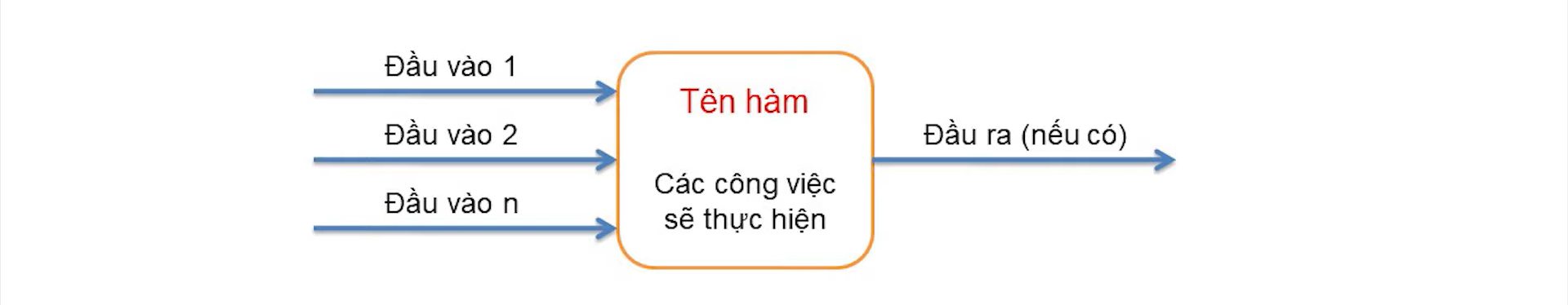
**-Các bước xây dựng hàm:**

+ Tên hàm.

+Hàm sẽ thực hiện công việc j

+Các đầu vào ( nếu có)

+Đầu ra ( nếu có)



*Ví dụ 1:* Hàm tính tổng hai số nguyên.

-tên hàm: tinhTong1

-công việc: tính và xuất tổng hai số nguyên

-Đầu vào : không có

-Đầu ra: không có

//xay dung ham:

void tinhTong1(){

int a,b,tong;

printf (“a= “);

scanf(“%d”,&a);

printf (“%d”);

scanf (“%d”,&b);

tong=a+b;

printf (“Tong hai so la %d”,tong);

}

*Ví dụ 2:* Hàm nhận tham số đầu vào và không trả về giá trị.

-Tên hàm: tinhTong2

-Công việc: tính và xuất tổng hai số nguyên

-Đầu vào : hai số a,b

-Đầu ra không có

//xay dung ham

void tinhTong2(int a, int b){

int tong;

tong=a+b;

printf (“Tông hai so la %d”,tong);

}

*Ví dụ 3:* Hàm trả về giá trị và không nhận tham số đầu vào.

+tên hàm: tinhTong3

+công việc: tính và xuất tổng 2 số nguyên

+Đầu vào: không có

+Đầu ra: một số nguyên có giá trị a+b

//xay dung ham

void tinhTong3(){

int a,b,tong;

printf (“nhap so thu nhat a= “);

scanf (“%d”,&a);

printf (“nhap so thu hai b=”);

scanf (“&d”,&b);

tong=a+b;

//ket thuc ham va tra ve ket qua

return tong;

}

*Ví dụ 4:* Hàm trả về giá trị và nhận tham số đầu vào

+Tên hàm: tinhTong4

+Công việc: tính và xuất tổng hai số nguyên

+Đầu vào: hai số nguyên a và b

+Đầu ra: một số nguyên có gái trị a+b

//xay dung ham:

int tinhTong4(inta, intb) {

int tong;

tong=a+b;

//ket thuc ham tra ve ket qua

return tong;

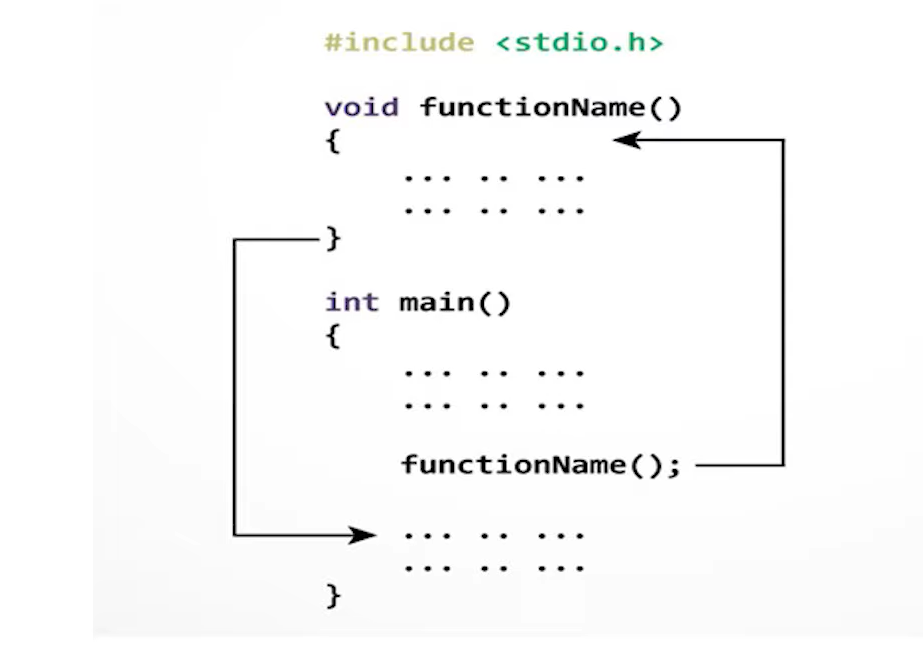
}

**Cách gọi hàm trong C**

-sau khi xây dựng hàm muốn sử dụng hàm ta phải tiến hành gọi hàm

-gọi hàm thoogn qua tên hàm và truyền tham số đàu vào haowcj xử lý kết quả trả về nếu có.

-Hàm trong lập trình C hoạt động như thế nào? Hình ảnh sau mô tả cách gọi một hàm do người dùng định nghĩa bên trong hàm main()



Gọi hàm cho các ví dụ trên:

-Gọi hàm tinhTong1 – Hàm không trả về giá trị không nhận tham số đầu vào.

void main(){

tinhTong1();

}

-Gọi hàm tinhTong2 – Hàm nhận tham số đầu vào và không trả về giá trị

void main(){

tinhTong2(10,20);

}

-Gọi hàm tinhTong3 – Hàm trả về giá trị và không nhận tham số đầu vào.

void main(){

int ketqua;

ketqua= tinhTong3();

}

\*lưu ý: khi hàm có giá trị trả về ta phải khai báo một biến sau đó gán biến đó bằng giá trị trả về của hàm.

-Gọi hàm tinhTong4 – Hàm trả về gái trị và nhận tham số đầu vào.

void main(){

int ketqua;

ketqua = tinhTong4(10,20);

}

-Một số lỗi thường gặp khi làm việc với hàm

+gọi sai tên hàm

+Gọi hàm không phù hợp với định nghĩa

+ Hàm có nhận tham số đầu vào nhưng khi gọi thì không truyền tham số đầu vào

+Truyền không dúng số lượng tham số

+Truyền tham số không phù hợp với kiểu dữ liệu lúc định nghĩa hàm

**Phạm vi của biến trong C**

+biến toàn cục-bien dia phuong: Khai báo ngoài tất cả các hàm (bao gồm cả hàm main) và có tác dụng lên toàn bộ chương trình.

+biến cục bộ: khai báo trong hàm hoặc khối {} và chỉ có tác dụng trong bản thân hàm hoặc khối đó (kể cả khối con của nó). Biến cục bộ sẽ bị xóa khỏi bộ nhớ khi kết thúc khối khai báo nó.

**Các cách truyền đối số trong C**

+ call by value-gọi hàm theo giá trị:

* Truyền đối số cho hàm ở dạng giá trị
* Có thể truyền hằng, biến, biểu thức nhưng hàm chỉ sẽ nhận giá trị.
* Được sử dụng khi không có nhu cầu thay đổi giá trị của tham số sau khi thực hiện hàm.

Ví dụ:

//xay dung ham hoan vi

void hoanDoi (int x, int y){  
 int temp;

temp=x;

x=y;

y=temp;

}

void main(){  
int a=100;

int b=200;

printf (“Truoc hoan doi, gia tri cua a: %d”,a);

printf (“truoc hoan doi, gia tri cua b: %d”,b);

//goi ham de hoan doi cac gia tri

hoanDoi (a,b);

printf (“sau hoan doi, gia tri a=%d”,a);

printf (“sau hoan doi, gia tri b=%d”,b);

}  
// kết quả a trước hoán đổi =100

//kết quả b trước hoán đổi =200

// kết quả a sau hoán đổi vẫn bằng 100

// kết quả b sau hoán đổi vẫn bằng 200

+ call by refernce = gọi hàm theo tham chiếu ngẫu nhiên

* Truyền đối số cho hàm ở dạng địa chỉ ( con trỏ)
* Không được truyền giá trị cho tham số này
* Được sử dụng khi có nhu cầu thay đổi giá trị của tham số sau khi thực hiện hàm

//xay dung ham hoan vi:

void hoanVi ( int \*x, int \*y){

int temp;

temp = \*x;

\*x=\*y;

\*y= temp

}

void main(){  
int a=100;

int b=200;

printf (“Truoc hoan doi, gia tri cua a: %d”,a);

printf (“truoc hoan doi, gia tri cua b: %d”,b);

//goi ham de hoan doi cac gia tri

hoanDoi (&a,&b);

printf (“sau hoan doi, gia tri a=%d”,a);

printf (“sau hoan doi, gia tri b=%d”,b);

}  
// kết quả a trước hoán đổi =100

//kết quả b trước hoán đổi =200

// kết quả a sau hoán đổi = 200

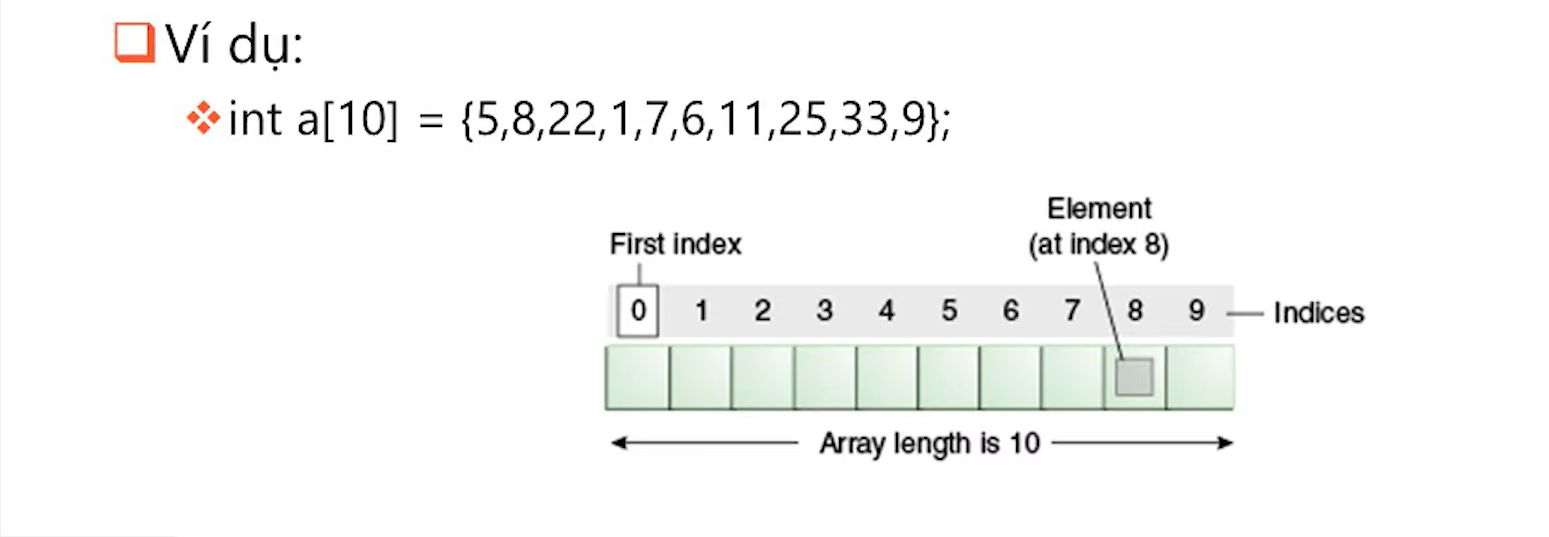
// kết quả b sau hoán đổi =100

**BÀI 6: MẢNG**

**-**mảng là tập hợp các phần tử cùng kiểu

**-**mảng có số lượng phần tử cố định và được cấp phát vùng nhớ liên tục.

**-** truy xuất các phần tử mảng bằng chỉ số, bắt đầu là 0.

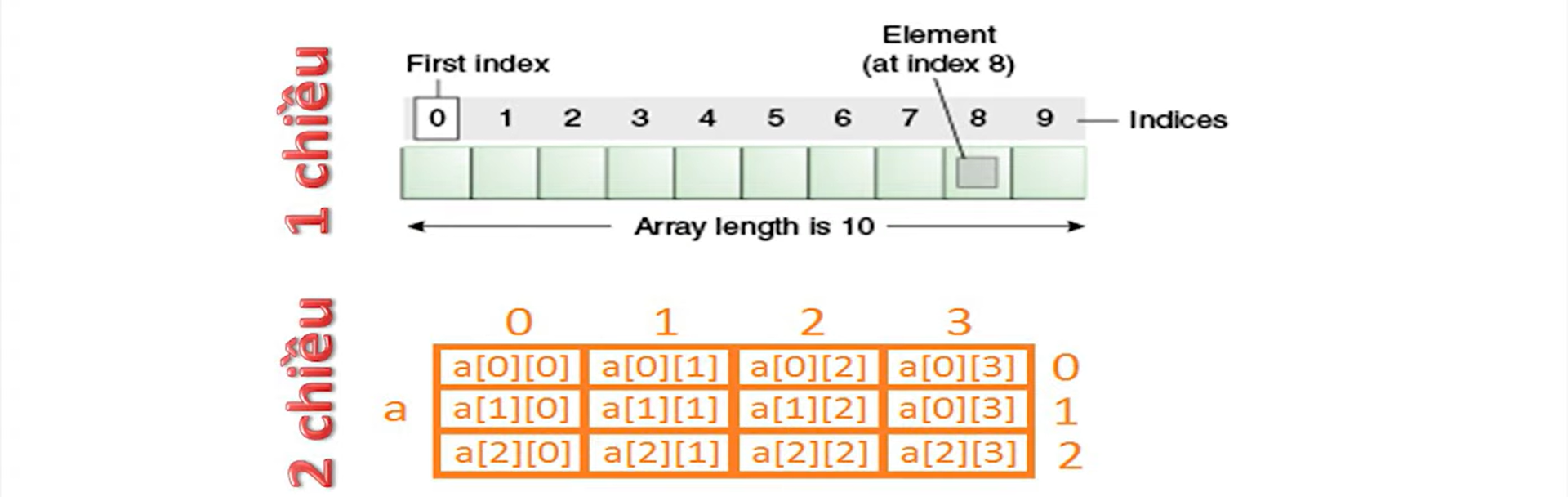
**Ví dụ:** 

**Phân loại mảng:**

+ mảng một chiều

+mảng hai chiều

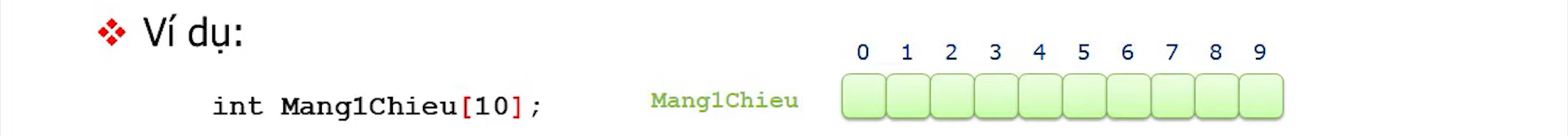
-chiều của mảng được xác định bằng số chỉ số cần thiết để định danh duy nhất mỗi phần tử.



**Khai báo một mảng:**

-Cú pháp: <kiểu\_dữ\_liệu> <tên\_mảng> <[biểu thức kích thước]>

* Tất cả các phần tử của mảng có cùng một kiểu. Một mảng được khai báo kiểu int không thể chứa các phần tử có kiểu khác.
* Kích thước mảng là một hằng số có giá trị nguyên dương.



-Không thể gán trực tiếp mảng này cho mảng khác mà thay vì thế ta sẽ gán từng phần tử của mảng này với từng phần tử của mảng khác.

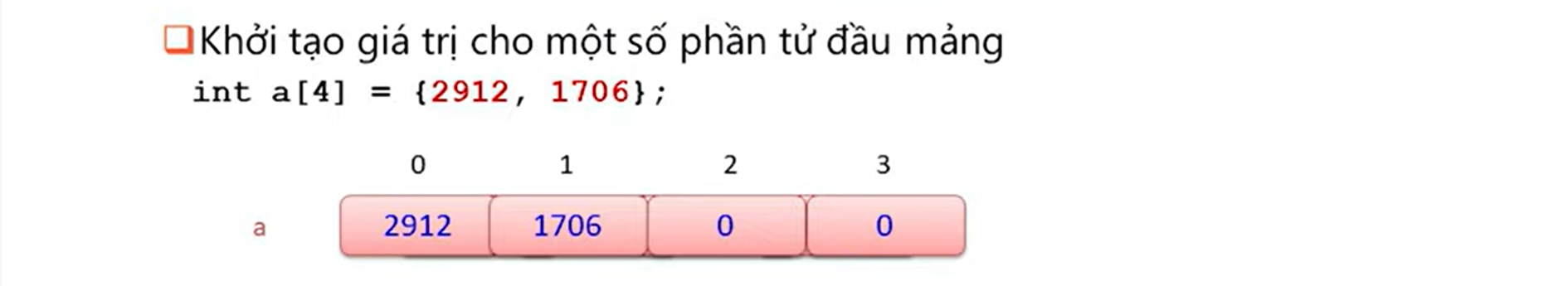
-Cấu trúc for là thao tác lý tưởng để thao tác các mảng.

**Khởi tạo mảng** gồm các cách sau:

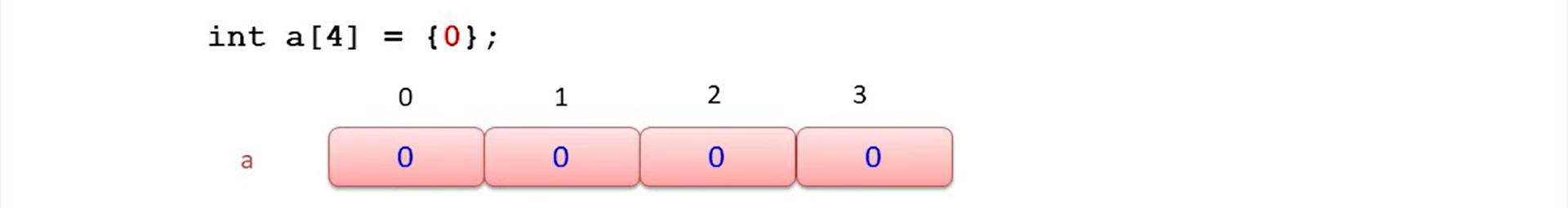
-Khởi tạo giá trị cho mọi phần tử của mảng

int a[4] = {1235, 4568, 4785, 2548}

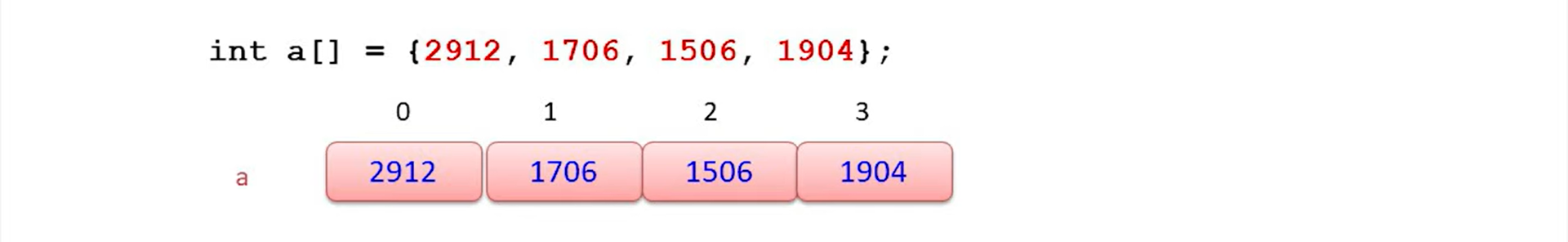
-Khởi tạo giá trị cho 1 số phần tử đầu mảng



-Khởi tạo giá trị 0 cho mọi phần tử của mảng

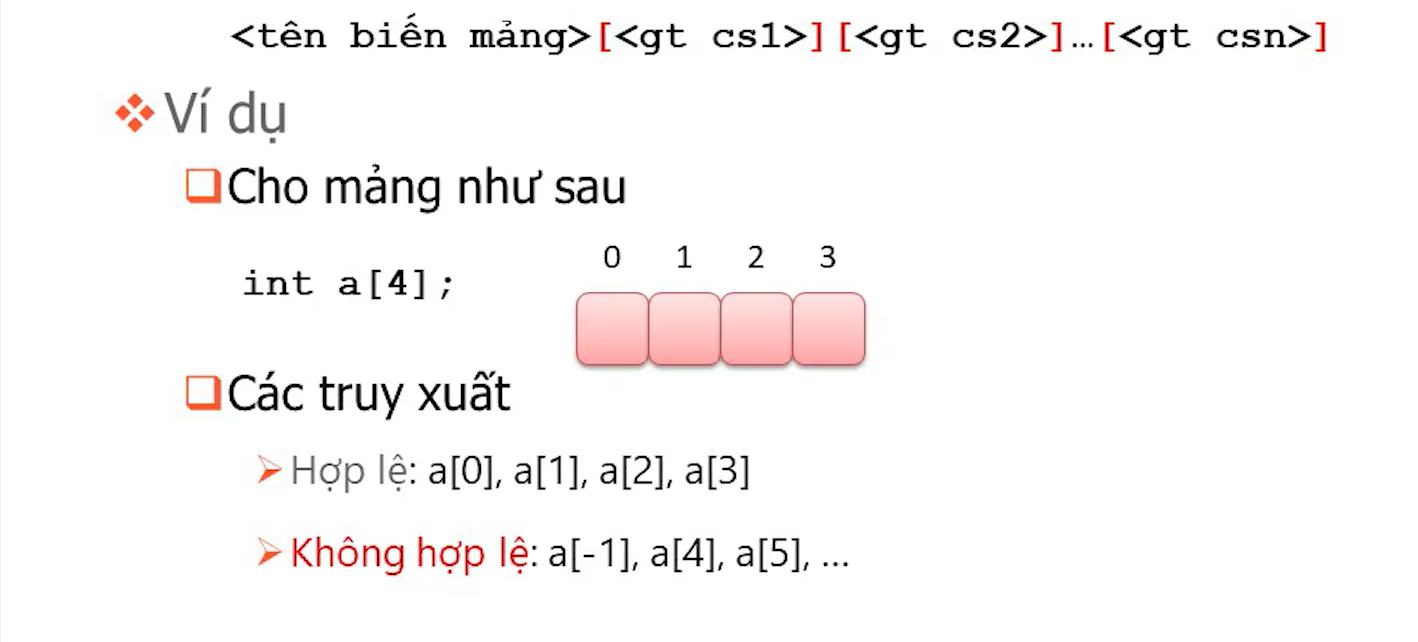


-tự động xác định số lượng phần tử



**Truy xuất mảng:**

-truy xuất mảng thông qua chỉ số:



**SẮP XẾP MẢNG & MẢNG HAI CHIỀU**

**CHUỖI VÀ THAO TÁC VỚI CHUỖI**