# BỘ BÀI TẬP THỰC HÀNH

# *KỸ THUẬT LẬP TRÌNH C*

## LỜI NÓI ĐẦU

Những năm gần đây, các học phần về kỹ thuật lập trình tại trường Đại học Giao thông vận tải đều được tổ chức thi trên máy. Vì thế, để đạt được kết quả cao trong học tập, sinh viên không chỉ học lý thuyết mà cần phải rèn luyện khả năng lập trình tốt.

Bộ bài tập này được Bộ môn Công nghệ phần mêm, khoa Công nghệ thông tin biên soạn nhằm giúp sinh viên hệ thống lại những nội dung cơ bản của ngôn ngữ lập trình C, đồng thời cung cấp những chỉ dẫn và bài tập để các em có thể luyện tập, từng bước nâng cao kỹ năng lập trình.

Bộ bài tập được biên soạn theo hướng thực hành, gồm năm bài, tương ứng với năm chủ đề chính trong học phần Kỹ thuật lập trình C:

- Bài 1: Các khái niệm cơ bản và các lệnh điều khiển
- Bài 2: Mảng 1 chiều
- Bài 3: Mảng 2 chiều
- Bài 4: Hàm
- Bài 5: Cấu trúc

Nội dung mỗi bài có bố cục tương tự nhau, đầu tiên là tóm tắt lý thuyết, tiếp theo là một số bài tập mẫu (có sẵn chương trình), các bài tập bắt buộc sinh viên cần hoàn thành và cuối cùng là một số bài tập làm thêm.

Mặc dù đã cố gắng để có được kết quả tốt nhất, nhưng chắc chắn Bộ bài tập này vẫn còn có những hạn chế, khiếm khuyết. Chúng tôi rất mong muốn trong quá trình sử dụng, các thầy cô và các em sinh viên có những ý kiến phản hồi để chúng tôi tiếp tục chỉnh sửa, hoàn thiện.

Các ý kiến đóng góp xin gửi về: Bộ môn Công nghệ phần mềm (phòng 310 nhà A9, trường ĐH GTVT), Email: cuonggt@gmail.com .

# Bài 1: Các khái niệm cơ bản và các lệnh điều khiển

## Mục đích bài thực hành

Giúp sinh viên làm quen với công cụ lập trình DevC++, thử nghiệm viết các chương trình đơn giản, qua đó nắm vững các kiến thức cơ bản như:

1. Khái niệm biến, hằng,kiểu dữ liệu. Biết cách khai báo và sử dụng trong lập trình giải các bài toán.

2. Các lệnh điều khiển trong C như điều khiển rẽ nhánh if, if…else, các lệnh điều khiển cho phép tổ chức các vòng lặp để lặp lại các lệnh như for, while, do…while và vận dụng để giải các bài toán.

## Tóm tắt lý thuyết

### 1. Biến, hằng, các kiểu dữ liệu:

#### *1.1 Biến*

Biến là đối tượng lưu giá trị và có thể thay đổi giá trị trong chương trình. Trong C trước khi sử dụng biến phải khai báo để chương trình dịch cấp phát bộ nhớ cho biến. Biến có thể được khai báo ở đầu chương trình (sau #include, #define) hay khai báo ở đầu của các hàm hay khối lệnh. Trong chương trình có thể sử dụng các lệnh gán giá trị cho biến hay nhập giá trị từ bàn phím, từ file. Cú pháp khai báo như sau:

```
Kiểu  danh_sách_biến;
```

Trong đó Kiểu là bất kỳ kiểu dữ liệu nào có sẵn của C hay do người dùng định nghĩa.
Ví dụ:

```
int i, j, m, n; /* Khai báo 4 biến kiểu nguyên */
float x, y, z; /* Khai báo 3 biến kiểu thực */
char ch1, ch2; /* Khai báo 2 biến kiểu kí tự */
```

#### *1.2 Hằng*

Hằng là đại lượng có giá trị không thay đổi trong chương trình C, có thể khai báo hằng trước khi sử dụng để sử dụng hằng theo tên hằng trong chương trình hoặc sử dụng giá trị hằng trực tiếp trong biểu thức khi cần.
Ví dụ khai báo hằng:

```
const int Max = 100;          /* Khai báo hằng có tên Max giá trị 100 */
```

#### *1.3 Các kiểu dữ liệu cơ bản trong C*

C đã định nghĩa sẵn một số kiểu dữ liệu cơ bản là kiểu số nguyên và kiểu số thực, người dùng có thể khai báo biến theo các kiểu này khi viết chương trình. Đi với các kiểu dữ liệu cũng sẽ có các phép toán tương ứng.

Các kiểu số nguyên

| Kiểu | Phạm vi biểu diễn | Kích thước |
|---|---|---|
| char | -128 → 127 | 1 byte |
| int | -32768 → 32767 | 2 byte |
| long hoặc long int | -2147483648→2147483647 | 4 byte |

Các kiểu số thực

| Kiểu | Phạm vi biểu diễn | Kích thước |
|---|---|---|
| float | 3.4E-38 → 3.4E+38 | 4 byte |
| double | 1.7E-308 → 1.7E+308 | 8 byte |

| | | |
|---|---|---|
| long double | 3.4E-4932 → 1.1E4932 | 10 byte |

**Một số ví dụ minh họa:**

**Ví dụ 1:** Nhập bán kính hình tròn. In ra màn hình chu vi và diện tích hình tròn.

```
#include<stdio.h>
int main()
{
   const float PI=3.14; /* Khai báo hằng*/
   float r;
   printf("Nhap ban kinh duong tron:");
   scanf("%f",&r);
   printf("Chu vi hinh tron=%f Dien tich hinh tron= %f ",
               2*r*PI,r*r*PI);
}
```

**Ví dụ 2:** Viết chương trình C nhập từ bàn phím 3 số nguyên, in ra màn hình tổng, tích và trung bình cộng của các số này.

```
#include<stdio.h>
int main()
{
  int a, b, c, tong, tich;
  printf("Nhap 3 so nguyen:");
  scanf("%d%d%d",&a,&b,&c);
  tong=a+b+c;
  tich=a*b*c;
  printf("Tong 3 so = %d \nTich 3 so = %d\n",tong, tich);
  printf("Trung binh cong 3 so = %f", (float)tong/3);
}
```

**Bài tập:**

1. Viết chương trình nhập vào một số đo nhiệt độ theo độ F (Fahrenheit) và xuất ra nhiệt độ tương đương của nó theo độ C (Celsius), sử dụng công thức chuyển đổi: C = (F – 32)*5/9

2. Viết chương trình nhập vào giờ, phút và giây, hãy đổi sang giây và in kết quả ra màn hình.

3. Viết chương trình nhập vào thời gian của một công việc nào đó tính bằng giây. Hãy chuyển đổi và in ra màn hình thời gian trên dưới dạng bao nhiêu giờ, bao nhiêu phút, bao nhiêu giây

## 2. Các câu lệnh điều khiển

### *2.1 Các câu lệnh điều khiển rẽ nhánh if, if …else*

Lệnh if cho phép lựa chọn để thực hiện một trong hai nhánh tùy thuộc vào điều kiện đúng (khác không) hay sai (bằng không) của một biểu thức, gọi là biểu thức điều kiện. Mỗi nhánh của lệnh if là một khối lệnh khác nhau. Lệnh if có 2 dạng như sau:
Dạng 1:

if (<biểu thức điều kiện>)
<khối lệnh>

Nếu <biểu thức điều kiện> có giá trị khác 0, khối lệnh sẽ được thực hiện, ngược lại, các lệnh sau <khối lệnh> được thực hiện.

**Ví dụ 1:** Nhập 3 số nguyên a, b, c từ bàn phím. Tìm và in giá trị lớn nhất và giá trị nhỏ nhất trong 3 số.

```
#include<stdio.h>
int main(){
     int a, b, c, max, min;
     printf("Nhap 3 so nguyen:");
     scanf("%d%d%d",&a,&b,&c);
     max=min=a;
     if(max<b)
        max=b;
     if(max<c)
       max=c;
     if(min>b)
       min=b;
     if(min>c)
      min=c;
     printf("So lon nhat la:%d, so nho nhat la:%d",max, min);
}
```

Dạng 2:

if (<biểu thức điều kiện>)
<khối lệnh 1>
else
<khối lệnh 2>

Nếu <biểu thức điều kiện> có giá trị khác 0, <khối lệnh 1> sẽ được thực hiện, ngược lại, <khối lệnh 2> được thực hiện.

**Ví dụ 2:** Giải phương trình ax+b=0. Trong đó a, b là số thực nhập từ bàn phím.

```
#include<stdio.h>
int main()
{    float a, b;
     printf("Nhap a, b:");
     scanf("%f%f",&a,&b);
     if(a!=0)
        printf("Phuong trinh co nghiem duy nhat x=%f", -b/a);
     else   /*a=0*/
       if(b==0)
          printf("Phuong trinh nghiem dung voi moi x");
       else   /*b!=0*/
         printf("Phuong trinh vo nghiem");
}
```

**Ví dụ 3:** Viết chương trình tính tiền điện với chỉ số mới và chỉ số cũ được nhập vào từ bàn phím. In ra màn hình chỉ số cũ, chỉ số mới và số tiền phải trả. Biết rằng 100 kWh đầu giá 1000, từ kWh 101 – 150 giá 1200, từ kWh 151 – 200 giá 2000, từ 201 trở lên giá 2500.

```
#include<stdio.h>
int main() {
     int  csc, csm, dntt;   float tiendien;
```

```
      nhapdulieu:
      printf("Nhap csc, csm:");
      scanf("%d%d",&csc,&csm);
      if(csm < csc)
      {
         printf("Nhap du lieu sai, nhap lai!\n");
         goto nhapdulieu;
      }
      dntt=csm-csc;
      tiendien=0;
      if (dntt<=100)
            tiendien=dntt*1000;
      else
         if(dntt<=150)
            tiendien=100*1000+(dntt-100)*1200;
         else
              if(dntt<=200)
                tiendien=100*1000+50*1200+(dntt-150)*2000;
              else
                tiendien=100*1000+50*1200+50*2000+(dntt-200)*2500;
      printf("\nDien nang tieu thu:%d kwh. ", dntt);
      printf("\nSo tien dien phai tra la:%f dong", tiendien);
}
```

**Bài tập:**

1. Viết chương trình nhập vào số nguyên dương, in ra số chẵn hay lẻ.
2. Cho ba số a, b, c đọc vào từ bàn phím. Hãy tìm giá trị lớn nhất của ba số trên và in ra kết quả.
3 Viết chương trình nhập vào 4 số thực a, b, c, d. Tìm và in ra số lớn nhất trong 4 số đó
4. Cho ba số a, b, c đọc vào từ bàn phím. Hãy in ra màn hình theo thứ tự tăng dần các số.
5. Viết chương trình nhập vào một số nguyên n gồm ba chữ số. xuất ra màn hình chữ số lớn nhất ở vị trí nào?
6. Viết chương trình nhập vào số nguyên n gồm ba chữ số. Xuất ra màn hình theo thứ tự tăng dần của các chữ số.
7. Nhập vào ngày, tháng, năm. Kiểm tra xem ngày, tháng, năm đó có hợp lệ không? In kết quả ra màn hình.
8. Viết chương trình nhập điểm thi từ bàn phím và hiển thị kết quả : kém nếu điểm từ 0 đến 3; Yếu nếu điểm là 4; Trung bình nếu điểm từ 5 đến 6; Khá nếu điểm từ 7 đến 8; Giỏi nếu điểm từ 9 đến 10.
9. Viết chương trình giải phương trình bậc 2: ax2 + bx + c = 0 (a, b, c nhập từ bàn phím). Xét tất cả các trường hợp có thể.
10. Viết chương trình nhập vào các hệ số thực, giải và biện luận về nghiệm của hệ phương trình bậc nhất

$$\begin{cases} a_1x + b_1y = c_1 \\ a_2x + b_2y = c_2 \end{cases}$$

11. Nhập ngày và tháng năm 2014 tìm ngày tiếp theo ngay sau, Ví dụ ngày 2/9 thì tiếp theo là 3/9 còn 28/2 thì tiếp theo là 1/3; ngày 31/12 thì tiếp theo là 1/1. Biết rằng năm 2014 tháng 1,3,5,7,8,10,12 có 31 ngày tháng 2 có 28 ngày, các tháng còn lại có 30 ngày.
12. Viết chương trình nhập vào điểm 3 môn thi: Toán, Lý, Hóa của học sinh. Nếu tổng điểm >= 15 và không có môn nào dưới 4 thì in kết quả đậu. Nếu đậu mà các môn đều lớn hơn 5 thì in ra lời phê "Học đều các môn", ngược lại in ra "Học chưa đều các môn", các trường hợp khác là "Thi hỏng".

13. Nhập vào một số nguyên dương là một năm, kiểm tra năm đó có phải năm nhuận hay không? Thuật toán: năm nhuận là năm chia hết cho 400 hoặc chia hết cho 4 nhưng không chia hết cho 100
14. Nhập r, a, b. Tìm giao điểm của vòng tròn tâm tại gốc O(0,0), bán kính r và đường thẳng y = ax + b.

15. Viết chương trình nhập vào một số nguyên n gồm ba chữ số. (Giả sử 3 chữ số của n khác nhau) Xuất ra màn hình chữ số lớn nhất ở vị trí nào?

*Ví dụ: n=291. Chữ số lớn nhất nằm ở hàng chục (chữ số 9).*

16. Viết chương trình tính tiền cước TAXI. Biết rằng:

- km đầu tiên là 13000đ.
- Mỗi km tiếp theo là 12000đ.
- Nếu lớn hơn 30km thì mỗi km thêm sẽ là 11000đ.

Hãy nhập số km sau đó in ra số tiền phải trả.

## 2.2 Các câu lệnh lặp

Các câu lệnh lặp cho phép thực hiện lặp lại một khối lệnh một số lần, giúp chương trình ngắn gọn. Trong C có một số lệnh để tổ chức vòng lặp đó là lệnh for, while và do … while.

### *2.2.1 Lệnh for*

Lệnh for cho phép thực hiện một khối lệnh một số lần xác định. Lệnh for thường dùng để giải các bài toán có tính chu trình, ví dụ như các bài toán về dãy số, về ma trận

Lệnh for có dạng sau:

```
for (<biểu thức khởi tạo>; <biểu thức kiểm tra>; <biểu thức tăng>)
        <khối lệnh>
```

Lệnh for làm việc theo các bước sau:
- Bước 1. Tính giá trị biểu thức khởi tạo.
- Bước 2. Tính giá trị của biểu thức kiểm tra. Nếu đúng, tới bước 3, ngược lại thoát khỏi lệnh for.
- Bước 3. Thực hiện khối lệnh.
- Bước 4. Tính giá trị biểu thức tăng, sau đó quay trở lại bước 2

**Ví dụ:** Tính s = 1+2+….+n. Với n là số nguyên dương nhập từ bàn phím

```
#include<stdio.h>
int main()
{     int  i, n, s;
      printf("Nhap n:");
      scanf("%d",&n);
      s = 0;
      for(i=1;i<=n;i++)
         s = s+i;
      printf("S = %d", s);
}
```

### *2.2.2 Lệnh while*

Lệnh while cho phép thực hiện một khối lệnh nhiều lần. Thông thường while sử dụng trong các trường hợp mà số lần lặp không xác định trước

Cú pháp của while có dạng:

```
while (<biểu thức kiểm tra>)
        <khối lệnh>
```

Trong cú pháp trên:

- while: từ khoá của lệnh while.
- Biểu thức kiểm tra: bắt buộc phải đặt trong cặp ngoặc tròn.
- Khối lệnh, còn gọi là phần thân của while, là khối lệnh cần thực hiện nhiều lần

**Ví dụ 1:** Chương trình dưới đây sẽ tìm số nguyên dương n nhỏ nhất sao cho 1 + 2 + ... + n > 10000

```
#include <stdio.h>
int  main() {
     float s;
     int n;
     s = 0;
     n = 0;
     while (s <= 1000)
     {
           n++;
           s=s+n;
     }
     printf("\nn = %d", n);
}
```

**Ví dụ 2:** Nhập số nguyên N, hãy in ra N là số nguyên tố hay hợp số.

```
#include<stdio.h>
#include<math.h>
int main()
{
     int i, n, nguyento;
     printf("Nhap so nguyen duong n=");
     scanf("%d",&n);
     nguyento=1; // ban dau gia su n la so nguyen to
     for(i=2;i<=int(sqrt(n));i++)
          if(n%i==0)
            nguyento=0;   // n khong phai la so nguyen to

     if(nguyento)
        printf("%d la so nguyen to",n);
     else
        printf("%d la hop so",n);
}
```

### *2.2.3 Lệnh do … while*

Trong các lệnh while và for, việc kiểm tra điều kiện kết thúc đặt ở đầu chu trình. khác với hai lệnh trên, trong chu trình do while việc kiểm tra điều kiện kết thúc đặt cuối chu trình. Như vậy thân của chu trình bao giờ cũng được thực hiện ít nhất một lần.

Lệnh do - while có dạng sau:

```
do
      <khối lệnh>
while (<biểu thức kiểm tra>);
```

Trong đó
- do, while là các từ khoá của lệnh do – while.

- Khối lệnh, còn gọi là phần thân của do – while là khối lệnh cần thực hiện nhiều lần.
- Biểu thức kiểm tra phải đặt trong cặp ngoặc tròn

**Ví dụ 1:** Lập chương trình tính $e^x$ theo công thức xấp xỉ:

$$e^x = 1 + \frac{x}{1} + \frac{x^2}{2!} + \cdots + \frac{x^n}{n!}$$

với độ chính xác 0.00001. Tức là n cần chọn sao cho

$$|\frac{x^n}{n!}| < 0.00001$$

```
#include <stdio.h>
#include<math.h>
int main(){
      float ex, x, eps;
      int n;
      printf("Nhap x:");
      scanf("%f",&x);
      ex=1;
      eps=1;
      n = 0;
      do {
            n++;
            eps = eps*x/n;
            ex = ex+eps;
      }
      while (fabs(eps)>=1e-5);
      printf("\nE luy thua %.2f = %f", x,ex);
}
```

**Bài tập về vòng lặp**

**Bài tập**

1. Cho số thực x và số nguyên dương n. Tính biểu thức:
   F = x + $(1+x)^3$ + $(2+x)^3$ + … + $(n+x)^3$
2. Cho số thực x và số nguyên dương n. Tính biểu thức:
   F = 2005 + (x+3)/9 + (x+5)/11 + ... + (x + 2n +1)/(2n+7)
3. Tính giai thừa của số nguyên dương n.
4. Cho một số nguyên dương n. Tính:
   S = 1.3.5.7...n nếu n lẻ và
   S = 2.4.6.8...n nếu n chẵn
5. Cho số nguyên dương n. Tính:
   S = $(1^2+2^2+3^2+...+n^2)/n^2$
6. Lập chương trình tính sin(x) với độ chính xác 0.0001 theo công thức:

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

7. Cần có tổng số 200000đ từ 3 loại giấy bạc 1000đ, 2000đ và 5000đ. Lập chương trình để tìm tất cả các phương án có thể.

**Bài làm thêm**

8. Viết chương trình tìm ước chung lớn nhất của 2 số nguyên dương nhập từ bàn phím.
9. Viết chương trình in ra n số nguyên tố đầu tiên (n nhập từ bàn phím)
10. Tìm số Fibonacci thứ n (n < 40), dùng vòng lặp.
Số Fibonacci thứ n:

$$F(n) = \begin{cases} 1 & n = 0,1 \\ F(n-2) + F(n-1) \end{cases}$$

11. Viết chương trình nhập vào số nguyên dương n và hai số thực x, y sau đó tính tổng

$$S = \frac{1+x}{y^2+1} + \frac{1+x^2}{y^4+1} + ... + \frac{1+x^n}{y^{2n}+1}$$

12. Viết chương trình nhập vào số nguyên dương n số thực x sau đó tính tổng

$$S = \frac{x + x(x+1) + x(x+1)(x+2) ... + x(x+1)...(x+n)}{1+2+...+n}$$

13. Viết chương trình nhập vào số nguyên dương n số thực x sau đó tính tổng

$$S = \sqrt{nx + \sqrt{(n-1)x + \sqrt{.... + \sqrt{1x + \sqrt{1}}}}}$$

15. Viết chương trình tìm và in ra các số (<=1000) thỏa tính chất: số bằng tổng các ước số của nó.

Ví dụ: 6 = 1 + 2 + 3

# Bài 2: Mảng 1 chiều

## Các khái niệm cơ bản

1. ***Khái niệm mảng***: Mảng có thể hiểu là một tập hợp nhiều phần tử có cùng một kiểu dữ liệu và có chung một tên. Mỗi phần tử mảng có vai trò như một biến và chứa được một giá trị.

2. ***Khai báo mảng một chiều:***

   <kiểu dữ liệu>  <tên mảng>[so_phan_tu];

trong đó:
- Kiểu dữ liệu của mảng là kiểu dữ liệu của các phần tử thuộc mảng đó, có thể là các kiểu: int, float…
- Tên mảng phải được đặt theo đúng quy định đặt tên trong ngôn ngữ C
- Số phần tử của mảng là một số nguyên dương được đặt trong 1 cặp ngoặc vuông, thường có thể chọn các số đủ lớn như 50, 100…

*Ví dụ về khai báo mảng*:

```
int a[50];
float x[100],y[100];
```

Trong ví dụ trên, câu lệnh thứ nhất khai báo 1 mảng một chiều có tên là a gồm 50 phần tử kiểu nguyên (int), mỗi phần tử có thể xem như một biến được đặt tên mặc định là a[0], a[1],…, a[49] . Câu lệnh thứ hai khai báo 2 mảng một chiều, mảng thứ nhất có tên là x gồm 100 phần tử kiểu thực (float), mỗi phần tử có thể xem như một biến được đặt tên mặc định là x[0], x[1],…, x[99], mảng thứ 2 tên là y cũng có các tính chất tương tự mảng x.

3. ***Lấy địa chỉ phần tử mảng một chiều:***

Để lấy địa chỉ của phần tử có chỉ số j trong mảng a ta sử dụng cú pháp: &a[j]

## Một số ví dụ minh họa

**Ví dụ 1:** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số thực $a_1, a_2, ..., a_n$ . In dãy số vừa nhập ra màn hình trên 1 dòng
b. Tìm  rồi in ra màn hình giá trị và vị trí của phần tử lớn nhất trong dãy.
c. Sắp xếp dãy số đó theo thứ tự tăng dần rồi in dãy số sau khi đã sắp xếp ra màn hình.

```
#include<stdio.h>
main()
{
     int n,i,j,vt;
     float a[100],max,tam;
     /*Nhap vao so phan tu n và day so thuc a1,a2...*/
     printf("Nhap so phan tu n = ");
     scanf("%d",&n);
     printf("Nhap day so thuc:");
     for (i=1;i<=n;i++)
          {
               printf("a[%d] = ",i);
```

```
            scanf("%f",&a[i]);
        }
    /*In day so vua nhap ra man hinh tren mot dong*/
    printf("Day so thuc vua nhap la:\n");
    for (i=1;i<=n;i++) printf("%.2f   ",a[i]);
    /*Tim va in ra man hinh gia tri va vi tri phan tu lon nhat*/
    max=a[1];vt=1;
    for (i=2;i<=n;i++)
        if (a[i]>max)
        {
            max=a[i];vt=i;
        }
    printf("\nPhan tu lon nhat cua day so thuc la %.2f ",max);
    printf("\nPhan tu lon nhat o vi tri thu %d trong day",vt);
    /*Sap xep day so theo thu tu tang dan sau do in ra man hinh*/
    for (i=1;i<n;i++)
        for (j=i+1;j<=n;j++)
            if (a[i]>a[j])
                {
                    tam=a[i];
                    a[i]=a[j];
                    a[j]=tam;
                }
    printf("\nDay so sau khi sap xep tang dan:\n");
    for (i=1;i<=n;i++) printf("%.2f   ",a[i]);
}
```

**Ví dụ 2:** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập một dãy các điểm trong mặt phẳng xOy. In dãy các điểm vừa nhập ra màn hình trên 1 dòng theo mẫu: (x1, y1), (x2, y2), ... (xn, yn).
b. Trong số các điểm đã nhập, in ra màn hình tọa độ của 1 điểm cách xa tâm O(0, 0) nhất và khoảng cách xa nhất đó.
**c.** Nhập số thực dương r. Đếm và in ra màn hình số đoạn thẳng tạo bởi 2 trong số các điểm đã nhập mà 1 đầu mút nằm trong, còn đầu mút kia nằm ngoài đường tròn tâm O(0, 0) bán kính r.

```
#include<stdio.h>
#include<math.h>
main(){
    int i,n,td,dem1,dem2;
    float x[50],y[50],kc,kcmax,r;
    /*Nhap mot day cac diem trong mat phang xOy*/
    printf("Nhap so diem n = ");
    scanf("%d",&n);
    printf("Nhap toa do day diem: \n");
    for (i=1;i<=n;i++)
        {
            printf("x[%d],y[%d] = ",i,i);
            scanf("%f%f",&x[i],&y[i]);
        }
    /*In day diem vua nhap theo mau (x1,y1),(x2,y2)...*/
    printf("Day diem vua nhap la:\n");
    for (i=1;i<=n;i++)
        printf("(%.2f,%.2f),  ",x[i],y[i]);
    /*In ra toa do diem cach xa tam O nhat va khoang cach do*/
    kcmax=sqrt(pow(x[1],2)+pow(y[1],2));
    td=1;
```

```
      for (i=2;i<=n;i++)
            {
                  kc=sqrt(pow(x[i],2)+pow(y[i],2));
                  if (kc>kcmax) { kcmax=kc; td=i; }
            }
      printf("\nDiem cach xa tam O nhat la (x[%d],y[%d])",td,td);
      printf("\nKhoang cach diem xa tam O nhat la %.2f",kcmax);
      /*Nhap so thuc duong r*/
      printf("\nNhap so thuc duong r = ");
      scanf("%f",&r);
      /*In ra so doan thang ... */
      dem1=dem2=0;
      for (i=1;i<=n;i++)
            {
                  kc=sqrt(pow(x[i],2)+pow(y[i],2));
                  if (kc>r) dem1++;
                  else dem2++;
            }
      printf("So doan thang ... la %d",dem1*dem2);
}
```

**Bài tập**

**1.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số thực $a_1, a_2, ..., a_n$ và xuất ra màn hình trên 1 dòng

b. Tìm giá trị lớn nhất của $\frac{a_1}{n}, \frac{a_2}{n-1}, ..., \frac{a_n}{1}$

c. Tính tổng $(a_1 + a_2) + (a_2 + a_3) + (a_3 + a_4) + ... + (a_{n-1} + a_n)$

**2.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số nguyên $a_1, a_2, ..., a_n$ và xuất ra màn hình trên 1 dòng

b. Đếm số bộ 2 số liên tiếp $(a_i, a_{i+1})$ với $1 \le i < n$ thỏa mãn số sau chia hết cho số trước

c. Tính tổng $a_1 + (a_1 + a_2) + (a_1 + a_2 + a_3) + ... + (a_1 + a_2 + ... + a_n)$

**3.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số nguyên $a_1, a_2, ..., a_n$ đã thỏa mãn tính chất không có bất kỳ hai số nào trùng nhau. In dãy số ra màn hình trên một dòng.

b. Kiểm tra xem dãy có phải đã sắp xếp tăng dần không?

c. Tính tổng $S = \frac{a_1a_2 + a_2a_3 + ... + a_{n-1}a_n + a_na_1}{1 + 2 + ... + n}$

**4.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập một dãy số thực $a_1$, $a_2$ ... $a_n$ và in dãy số ra màn hình trên một dòng.

b. Nhập x là số nguyên từ bàn phím. Liệt kê các số trong dãy số trên mà giá trị tuyệt đối của nó lớn hơn x

c. Đếm các cặp 2 phần tử bất kỳ mà tích của chúng bằng tổng của chúng

d. Xuất ra màn hình hai số bất kỳ trong dãy sao cho hiệu trị tuyệt đối $|a_i| - |a_j|$ lớn nhất

**5.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập một dãy các điểm trong mặt phẳng xOy. In dãy các điểm vừa nhập ra màn hình trên 1 dòng theo mẫu: (x1, y1), (x2, y2), ... (xn, yn).

b. Nhập 2 số thực a và b. Đếm rồi in ra màn hình số điểm nằm phía trên đường thẳng y = ax + b.

c. Trong số các đoạn thẳng tạo bởi 2 trong số các điểm đã nhập, tính rồi in ra màn hình độ dài của đoạn thẳng dài nhất.

**6.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập một dãy các điểm trong mặt phẳng xOy. In dãy các điểm vừa nhập ra màn hình trên một dòng theo mẫu: $(x_1, y_1)$, $(x_2, y_2)$, ... $(x_n, y_n)$.
b. Tính độ dài đường gấp khúc đi qua các điểm này theo thứ tự nhập vào $(x_1, y_1)$ => $(x_2, y_2)$ => ... => $(x_{n-1}, y_{n-1})$ => $(x_n, y_n)$.
c. Tính diện tích đường tròn tâm O(0,0) có bán kính nhỏ nhất chứa tất cả các điểm thuộc góc phần tư thứ nhất. Đếm xem có bao nhiêu điểm không thuộc đường tròn trên.

**7.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập đa thức P(x) bậc n và in các hệ số của đa thức trên 1 dòng.
b. Đếm rồi in ra màn hình số các hệ số có giá trị bằng 0.
c. Nhập số thực $x_0$ rồi tính và in ra màn hình $P'(x_0)$
d. Xác định 2 hệ số lớn nhất của đa thức trên

**Bài tập làm thêm**

**1.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số nguyên $a_1, a_2, ..., a_n$ xuất ra màn hình trên một dòng
b. Nhập số nguyên dương k, tính trung bình cộng những số chia hết cho k trong dãy
c. Tính tổng:

$$S = \frac{a_1}{a_2} + \frac{a_2}{a_3} + ... + \frac{a_{n-1}}{a_n}$$

**2.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số thực $a_1, a_2, ..., a_n$ và xuất ra màn hình trên 1 dòng
b. Tìm giá trị nhỏ nhất của

$$\frac{a_1}{n}, \frac{a_2}{n-1}, ..., \frac{a_n}{1}$$

c. Tính tổng

$$S = \frac{a_1 + a_2 + a_3}{a_1 + a_3} + \frac{a_2 + a_3 + a_4}{a_2 + a_4} + ... + \frac{a_{n-2} + a_{n-1} + a_n}{a_{n-2} + a_n}$$

**3.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số nguyên $a_1, a_2, ..., a_n$ đã thỏa mãn tính chất âm dương xen kẽ tức là $a_1, a_3, a_5, \cdots$ là dãy dương còn $a_2, a_4, a_6, \cdots$ là dãy âm sau đó xuất dãy ra màn hình
b. Tìm giá trị nhỏ nhất của các phần tử âm trong dãy $a_2, a_4, a_6, ...$
c. Nhập số nguyên dương k $(1 \le k \le n/3)$ tính tích những phần tử ở các vị trí chia hết cho k trong dãy gồm

$$a_k \times a_{2k} \times a_{3k} \times ... \times a_{mk}$$

**4.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số nguyên $a_1, a_2, ..., a_n$ đã thỏa mãn tính chất cứ hai số lẻ đến một số chẵn tức là $a_3, a_6, a_9, \cdots$ là số chẵn còn lại là lẻ sau đó xuất dãy ra màn hình
b. Tìm giá trị lớn nhất các phần tử chẵn trong dãy $a_3, a_6, a_9, ...$
c. Tính tổng

$$S = \frac{a_1 + a_2}{a_1 - a_2} + \frac{a_2 + a_3}{a_2 - a_3} + ... + \frac{a_{n-1} + a_n}{a_{n-1} - a_n}$$

**5.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số nguyên $a_1, a_2, ..., a_n$ và xuất ra màn hình trên 1 dòng
b. Đếm số bộ 3 số liên tiếp $(a_{i-1}, a_i, a_{i+1})$ với $1 < i < n$ thỏa mãn $a_i - a_{i-1} = a_{i+1} - a_i$
c. Tính tổng

$$S = \frac{a_1}{a_1 + a_2} + \frac{a_2}{a_2 + a_3} + ... + \frac{a_{n-1}}{a_{n-1} + a_n}$$

**6.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số nguyên $a_1, a_2, ..., a_n$ và xuất ra màn hình trên 1 dòng
b. Đếm số bộ 2 số liên tiếp $(a_i, a_{i+1})$ với $1 \le i < n$ thỏa mãn số sau chia hết cho số trước
c. Tính tổng $a_1 + (a_1 + a_2) + (a_1 + a_2 + a_3) + ... + (a_1 + a_2 + ... + a_n)$

**7.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy số nguyên $a_1$, $a_2$ ... $a_n$ và in dãy số ra màn hình trên một dòng.
b. Kiểm tra xem tổng các phần tử chẵn có bằng tổng các phần tử lẻ không xuất ra thông báo những trường hợp dãy không có phần tử chẵn hoặc không có phần tử lẻ
c. Tính giá trị hiệu max – min của dãy

**8.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy số nguyên $a_1$, $a_2$ ... $a_n$ (giả sử các số khác nhau tưng đôi một) và in dãy số ra màn hình trên một dòng.
b. Tìm 2 giá trị lớn nhất của dãy xuất ra màn hình
c. Tính trung bình cộng các số chia 11 dư 8.

**9.** Viết chương trình bằng ngôn ngữ C để giải quyết bài toán sau:
a. Nhập một dãy các điểm trong mặt phẳng xOy. In dãy các điểm vừa nhập ra màn hình trên một dòng theo mẫu: ($x_1$, $y_1$), ($x_2$, $y_2$), ... ($x_n$, $y_n$).
b. Xác định và in ra màn hình tọa độ tất cả các điểm xa tâm O(0,0) nhất.
c. Tính độ dài đường gấp khúc đi qua những điểm ở vị trí lẻ.

**10.** Viết chương trình bằng ngôn ngữ C để giải quyết bài toán sau:
a. Nhập một dãy số thực $a_1$, $a_2$ ... $a_n$ và in dãy số ra màn hình trên một dòng.
b. Đếm những cặp 3 số liên tiếp $a_{i-1}, a_i, a_{i+1}$ nếu sắp xếp chúng tăng dần chúng sẽ là cấp số cộng
c. Sắp xếp dãy số giảm dần và xuất ra màn hình

**11.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập vào số phần tử n và dãy số nguyên $a_1, a_2, ..., a_n$ đã thỏa mãn tính chất không có bất kỳ hai số nào trùng nhau
b. Tìm tất cả các cặp 2 số ở vị trí bất kỳ trong dãy mà tổng của chúng bằng 10
c. Tính trung bình cộng các số chia hết cho 3 trong dãy

**12.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập một dãy các điểm trong mặt phẳng xOy. In dãy các điểm vừa nhập ra màn hình trên 1 dòng theo mẫu: (x1, y1), (x2, y2), ... (xn, yn).
b. Nhập 2 số thực a và b. Đếm rồi in ra màn hình số điểm nằm phía trên đường thẳng y = ax + b.
c. Trong số các đoạn thẳng tạo bởi 2 trong số các điểm đã nhập, tính rồi in ra màn hình độ dài của đoạn thẳng dài nhất.

**13.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy các điểm trong mặt phẳng xOy. In dãy các điểm vừa nhập ra màn hình trên 1 dòng theo mẫu: ($x_1$, $y_1$), ($x_2$, $y_2$), ... ($x_n$, $y_n$).
b. Đếm số điểm nằm phía trên đường phân giác của góc phần tư thứ nhất và in kết quả ra màn hình.
c. Trong số các đoạn thẳng tạo bởi 2 trong số các điểm đã nhập, đếm rồi in ra màn hình số đoạn thẳng cắt trục hoành

**14.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:

a. Nhập một dãy các điểm trong mặt phẳng xOy. In dãy các điểm vừa nhập ra màn hình trên 1 dòng theo mẫu: $(x_1, y_1)$, $(x_2, y_2)$, ... $(x_n, y_n)$.
b. Đếm số điểm nằm phía trong đường tròn tâm O(0,0), bán kính r (với r là số thực nhập từ bàn phím).
c. Xác định 3 điểm có khoảng cách gần trục 0x nhất

**15.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy số nguyên $a_1$, $a_2$ ... $a_n$ và in dãy số ra màn hình trên một dòng.
b. Kiểm tra xem dãy có phải là dãy đối xứng không.
c. Xác định 1 số nguyên chia hết cho 3 lớn nhất trong dãy.

**16.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy số thực $a_1$, $a_2$ ... $a_n$ và in dãy số ra màn hình trên một dòng.
b. Tính trung bình cộng các số nhỏ hơn 10 trong dãy
c. Xác định giá trị nhỏ nhất của hiệu các cặp 2 số liên tiếp trong dãy $(a_1, a_2)$, $(a_2, a_3)$ ... $(a_{n-1}, a_n)$.

**17.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy số thực $a_1$, $a_2$ ... $a_n$ và in dãy số ra màn hình trên một dòng
b. Đếm xem trong dãy số có bao nhiêu phần tử nằm trong đoạn [-10, 25].
c. Kiểm tra xem dãy số có phải là một dãy tăng thực sự không.

**18.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy số nguyên $a_1$, $a_2$ ... $a_n$ và in dãy số ra màn hình trên một dòng.
b. Tính tổng các phần tử trong dãy có giá trị thuộc đoạn [6,19].
c. Xác định giá trị của 1 phần tử chẵn lớn nhất trong dãy trên.

**19.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy số nguyên $a_1$, $a_2$ ... $a_n$ và in dãy số ra màn hình trên một dòng.
b. Tính tích các phần tử chia cho 5 dư 2 và nhỏ hơn 20.
c. Sắp xếp dãy số trên theo thứ tự tăng dần. In dãy số sau khi sắp xếp ra màn hình trên một dòng.

**20.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy số nguyên $a_1$, $a_2$ ... $a_n$ và in dãy số ra màn hình trên một dòng.
b. Đếm xem trong dãy số có bao nhiêu cặp hai phần tử liên tiếp $(a_i, a_{i+1})$ mà tích của chúng chia hết cho tổng của chúng.
c. Xác định giá trị chẵn lớn nhất trong dãy, đếm xem trong dãy số có bao nhiêu phần tử bằng giá trị chẵn lớn nhất trên.

**21.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy các điểm trong mặt phẳng xOy. In dãy các điểm vừa nhập ra màn hình trên một dòng theo mẫu: $(x_1, y_1)$, $(x_2, y_2)$, ... $(x_n, y_n)$.
b. Đếm số điểm nằm bên trong góc phần tư thứ ba.
c. Tính diện tích đường tròn tâm O(0,0) có bán kính nhỏ nhất chứa tất cả các điểm thuộc góc phần tư thứ nhất. Đếm xem có bao nhiêu điểm không thuộc đường tròn trên

**22.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy các điểm trong mặt phẳng xOy. In dãy các điểm vừa nhập ra màn hình trên 1 dòng theo mẫu: (x1, y1), (x2, y2), ... (xn, yn).
b. Tính diện tích hình chữ nhật nhỏ nhất có các cạnh song song với trục tọa độ và chứa tất cả các điểm.
c. Đếm số điểm nằm trong đường tròn tâm A1(x1,y1) có bán kính 15.

**23.** Viết chương trình bằng ngôn ngữ lập trình C để giải quyết bài toán sau:
a. Nhập một dãy các điểm trong mặt phẳng xOy. In dãy các điểm vừa nhập ra màn hình trên một dòng theo mẫu: $(x_1, y_1)$, $(x_2, y_2)$, ... $(x_n, y_n)$.
b. Tính giá trị:

$$S = \sqrt{(x_1 - y_1)^2 + ... + (x_n - y_n)^2}$$

c. Xác định 1 điểm gần gốc tọa độ O(0,0) nhất.

# Bài 3: Mảng 2 chiều

**1. Khái niệm về mảng và ma trận**

- Mảng là một tập hợp các đối tượng có cùng kiểu và cùng một tên. Các phần tử trong mảng được phân biệt nhờ chỉ số. Chỉ số mảng bắt đầu từ 0.
- Tên mảng là địa chỉ của phần tử đầu tiên trong mảng.
- Ma trận là một loại mảng hai chiều, tức là có sử dụng hai chỉ số.

**Ví dụ:** So sánh mảng 1 chiều và mảng 2 chiều

int A[10], B[4][3];
A là mảng 1 chiều có kiểu int, quản lý 10 biến nguyên lần lượt là:
A[0], A[1], …, A[9].
B là mảng 2 chiều (còn gọi là ma trận) có kiểu int, quản lý 4 x 3 biến nguyên lần lượt là:
B[0][0], B[0][1], B[0][2],
B[1][0], B[1][1], B[1][2],
B[2][0], B[2][1], B[2][2],
B[3][0], B[3][1], B[3][2].

**2. Khai báo và khởi gán**

<Kiểu mảng> <Tên mảng>[số hàng][số cột];

**3. Một số dạng bài tập về ma trận**

1. Nhập dữ liệu cho một ma trận, dữ liệu được nhập vào có thể từ bàn phím, từ file dữ liệu
2. In dữ liệu trong ma trận ra màn hình.
3. Tính toán các bài toán về ma trận gồm có:
    a. Các phép toán đối với ma trận như tính tổng, tích hai ma trận
    b. Tính tổng, tích hoặc đếm các phần tử trong ma trận thỏa một số điều kiện nào đó như:
        i. Về vị trí của phần tử: nằm trên hoặc nằm phía trên đường chéo chính, ở hàng thứ hai, ở cột thứ ba, ...
        ii. Về tính chất, thuộc tính của phần tử: như phần tử chẵn lẻ, âm dương, thuộc một đoạn [a, b] với a, b là các giá trị biết trước, ...

**4. Phương pháp làm các bài toán về ma trận**

- Về cơ bản khi lập trình để làm các bài toán về ma trận ta phải duyệt (xét) toàn bộ các phần tử trong ma trận, với mỗi phần tử được xét duyệt ta kiểm tra phần tử đó có thỏa điều kiện của bài toán đặt ra không, nếu thỏa thì nó sẽ được tính, nếu không thì phần tử đó không được tính. Xét xong phần tử đó thì chuyển đến xét phần tử tiếp theo cho đến khi xét hết toàn bộ các phần tử trong ma trận.
- Để duyệt toàn bộ một ma trận thì ta phải sử dụng 2 vòng lặp (như for) cùng nhau.
- Để kiểm tra một phần tử có thỏa điều kiện của bài toán thì các câu lệnh điều kiện phải được sử dụng (như if)
- Trong một số trường hợp sẽ phải kết thúc lặp khi gặp một điều kiện nào đó thì ta sẽ sử dụng câu lệnh break

Lưu ý: Khi làm các bài toán về ma trận thì cần làm rõ các đặc điểm sau: ma trận có kiểu dữ liệu nguyên hay thực để áp dụng các phép toán phù hợp. Ma trận có là ma trận vuông không, ...

**Một số ví dụ minh họa**

**Ví dụ 1:** Cho ma trận nguyên A có m hàng n cột, hãy viết đoạn chương trình nhập giá trị các phần tử của ma trận A .

| | |
|---|---|
| `int m, n, i, j;`<br>`int a[10][10], tg;` | khai báo các biến m, n, i, j là số nguyên a là ma trận kích thước tối đa là 10 x 10 |
| `printf("Nhap m = "); scanf("%d",&m);`<br>`printf("Nhap n = "); scanf("%d",&n);` | Nhập các kích thước của ma trận là m, n với m là số hàng, n là số cột.<br>Lưu ý m và n đều phải <= 10 |
| `printf("\nNhap ma tran:\n");`<br>`for(i = 1; i <= m; i++)`<br>`for(j = 1; j <= n; j++)`<br>`{`<br>`  printf("a[%d][%d] = ", i, j);`<br>`  scanf("%d",&tg); a[i][j] = tg;`<br>`}` | Đưa ra dòng thông báo "nhập ma trận"sử dụng hai vòng lặp for để duyệt toàn bộ các phần tử trong ma trận:<br>vòng for có chỉ số i chỉ hàng<br>vòng for có chỉ số j chỉ cột |

Lưu ý: Giả sử cho A là ma trận thực có m hàng n cột, thì để nhập giá trị toàn bộ phần tử cho ma trận A này đoạn code cũng giống như trên nhưng ta cần sửa đổi 2 vị trí sau:
1) vị trí khai báo ma trận và biến tg, float a[10][10], tg;
2) câu lệnh nhập cụ thể là: scanf("%f",&tg);

**Ví dụ 2:** Giả sử đã có giá trị của các phần tử trong ma trận nguyên A có m hàng n cột, hãy viết đoạn chương trình nhập in ma trận A ra màn hình

| | |
|---|---|
| `int m, n, i, j;`<br>`int a[10][10];` | khai báo các biến m, n, i, j là số nguyên<br>a là ma trận kích thước tối đa là 10 x 10 |
| `printf("\nMa tran:\n");`<br>`for(i = 1; i <= m; i++)`<br>`{`<br>`  for(j = 1; j <= n; j++)`<br>`  printf("%5d ",a[i][j]);`<br>`  printf("\n");`<br>`}` | Đưa ra dòng thông báo "Ma trận"<br>sử dụng vòng for có chỉ số i chỉ hàng<br>trong hàng i sử dụng: vòng for có chỉ số j chỉ cột: để in giá trị các phần tử thuộc hàng i đưa lệnh xuống dòng để in hàng tiếp theo |

Lưu ý: Giả sử cho A là ma trận thực có m hàng n cột, thì để in ma trận A ra màn hình ta sử dụng lại đoạn này đoạn code trên nhưng ta cần sửa đổi 1 vị trí sau:
1) in giá trị phần tử a[i][j]) thành printf("%7.1f ",a[i][j]);

**Bài tập**

**1.** Cho ma trận nguyên A có m hàng n cột, hãy viết chương trình nhập giá trị cho các phần tử của ma trận A. Hãy In ma trận a ra màn hình và cho biết có bao nhiêu phần tử lẻ trong ma trận, hãy in ra và tính tổng các phần tử lẻ này.

```
#include <stdio.h>
int main(void)
{
    int m, n, i, j;
    int a[10][10];
    int S, tg, d;
    printf("Nhap m = "); scanf("%d",&m);
    printf("Nhap n = "); scanf("%d",&n);
    printf("\nNhap ma tran:\n");
    for(i = 1; i <= m; i++)
    for(j = 1; j <= n; j++)
```

```
    {
        printf("a[%d][%d] = ", i, j);
        scanf("%d",&tg); a[i][j] = tg;
    }
    printf("\nMa tran:\n");
    for(i = 1; i <= m; i++)
    {
        for(j = 1; j <= n; j++) printf("%5d ",a[i][j]);
        printf("\n");
    }
    printf("\nCac phan tu le trong ma tran:\n");
    d = 0; S = 0;
    for(i = 1; i <= m; i++)
    for(j = 1; j <= n; j++)
    if(a[i][j] % 2 == 1)
    {
        d++;
        S += a[i][j];
        printf("%d ",a[i][j]);
    }
    printf("\nSo phan tu le: %d", d);
    printf("\nTong gia tri phan tu le: %d", S);
}
```

**2.** Viết chương trình nhập số nguyên dương n và các phần tử của ma trận vuông A gồm các số thực thực, A có n hàng n cột. Tính tổng giá trị của các phần tử nằm phía trên đường chéo chính và không thuộc đường chéo chính (lưu ý: phần tử (1, 2) nằm phía trên đường chéo chính còn phần tử (1, 1) nằm trên đường chéo chính). Tính tích giá trị của các phần tử lớn hơn 5 và nhỏ hơn 20.

```
#include <stdio.h>
int main(void)
{
    int n, i, j;
    float a[10][10], tg, T, S;
    printf("Nhap n = "); scanf("%d",&n);
    printf("\nNhap ma tran vuong A:\n");
    for(i = 1; i <= n; i++)
          for(j = 1; j <= n; j++)
          {
               printf("Nhap a[%d][%d] = ", i,j);
               scanf("%f",&tg); a[i][j] = tg;
          }
    printf("\n\nMa tran A\n\n");
    for(i = 1; i <= n; i++) {
         for(j = 1; j <= n; j++)
               printf("%7.2f ",a[i][j]);
         printf("\n");
    }
    S = 0.0;
    for(i = 1; i <= n; i++)
    for(j = i + 1; j <= n; j++)
          S += a[i][j];
    printf("\nTong gia tri cac pt phia tren dcc: %0.2f",S);
    T = 1;
    for(i = 1; i <= n; i++)
```

```
            for(j = 1; j <= n; j++)
                if(a[i][j] > 5.0 && a[i][j] < 20.0)
                    T *= a[i][j];
    printf("\nTich gia tri cac pt thuoc (5, 20): %0.2f", T);
}
```

**Bài tập làm thêm**

1. Cho ma trận A các số nguyên gồm có m hàng, n cột. Hãy viết chương trình nhập giá trị các phần tử cho ma trận và in ma trận A ra màn hình. Hãy tính tổng các số dương trong ma trận và Tính tích các giá trị lẻ trong ma trận, in các giá trị tính được ra màn hình.

2. Cho ma trận A các số thực gồm có m hàng, n cột. Hãy viết chương trình nhập giá trị các phần tử cho ma trận và in ma trận A ra màn hình. Hãy tính các giá trị tổng trên mỗi dòng trong ma trận A và Tính tích các giá trị dương trên từng cột trong ma trận. In các giá trị tính được ra màn hình.

3. Cho ma trận A các số nguyên gồm có m hàng, n cột. Hãy viết chương trình nhập giá trị các phần tử cho ma trận và in ma trận A ra màn hình. Hãy tính tổng giá trị các phần tử chẵn trong ma trận A và Tính tích giá trị các phần tử lẻ trong ma trận. In các giá trị tính được ra màn hình.

4. Cho ma trận A các số nguyên gồm có m hàng, n cột. Hãy viết chương trình nhập giá trị các phần tử cho ma trận và in ma trận A ra màn hình. Hãy cho biết có bao nhiêu phần tử chẵn trong ma trận A có giá trị nằm trong đoạn [5, 20] và có bao nhiêu phần tử lẻ trong ma trận có giá trị nhỏ hơn 100. In các giá trị tính được ra màn hình.

5. Cho ma trận A các số nguyên gồm có m hàng, n cột. Hãy viết chương trình nhập giá trị các phần tử cho ma trận và in ma trận A ra màn hình. Hãy tính trung bình cộng các phần tử chẵn trong ma trận A và Tính trung bình nhân các các phần tử lẻ trong ma trận. In các giá trị tính được ra màn hình.

6. Cho ma trận A các số thực gồm có m hàng, n cột. Hãy viết chương trình nhập giá trị các phần tử cho ma trận và in ma trận A ra màn hình. Hãy tính giá trị lớn nhất và giá trị nhỏ nhất trong ma trận, có bao nhiêu phần tử trong ma trận có giá trị lớn nhất và có bao nhiêu phần tử trong ma trận có giá trị nhỏ nhất. In các giá trị tính được ra màn hình.

7. Cho ma trận A các số thực gồm có m hàng, n cột. Hãy viết chương trình nhập giá trị các phần tử cho ma trận và in ma trận A ra màn hình. Hãy tính giá trị lớn nhất và giá trị nhỏ nhất trong ma trận, có bao nhiêu phần tử trong ma trận có giá trị lớn nhất và có bao nhiêu phần tử trong ma trận có giá trị nhỏ nhất. In các giá trị tính được ra màn hình.

8. Cho ma trận vuông A các số nguyên gồm có n hàng, n cột. Hãy viết chương trình nhập giá trị các phần tử cho ma trận và in ma trận A ra màn hình. Hãy tính tổng các phần tử nằm phía trên đường chéo chính và hãy cho biết phần tử có giá trị lớn nhất trong ma trận. In các giá trị tính được ra màn hình.

## Bài 4: Hàm

### 1. Tóm tắt lý thuyết

**Khái niệm:** Hàm là một đoạn chương trình độc lập thực hiện trọn vẹn một công việc nhất định sau đó trả về giá trị cho chương trình gọi nó. Một chương trình có thể chia nhỏ thành nhiều hàm thực hiện những việc độc lập.

**Cú pháp xây dựng hàm:**

```
<Kiểu dữ liệu> <Tên hàm> ([Danh sách tham số])
{
     <Khối lệnh chức năng>
     [return [giá trị / biểu thức];]
}
```

Trong cú pháp trên, những thành phần nào đặt trong cặp dấu ngoặc vuông "[ ]" là nhưng thành phần có thể khuyết, còn những thành phần đặt trong cặp dấu nhọn "< >" là những thành phần bắt buộc phải có khi khai báo và xây dựng hàm.

- **Kiểu dữ liệu** trả về của hàm (kết quả của hàm/ đầu ra), gồm 2 loại:
  - **void**: Không trả về giá trị nào cả (dạng này chính là thủ tục).
  - **float** / **int** / **long** / **char *** / **kiểu cấu trúc** / … : Trả về giá trị kết quả có kiểu dữ liệu tương ứng với bài toán (chỉ trả về được 1 giá trị theo kiểu dữ liệu).
- **Tên hàm**: Đặt tên theo qui ước sao cho phản ánh đúng chức năng thực hiện của hàm.
- **Danh sách các tham số** (nếu có): đây chính là các tham số hình thức, chúng sẽ nhận giá trị thực bằng cách truyền tham số mỗi khi hàm này được gọi đến. Nếu có nhiều tham số thì phải có dấu phẩy (,) để phân cách. Trong trường hợp hàm không có tham số thì vẫn phải để cặp dấu ngoặc đơn.

**Ví dụ 1:** Nhập số tự nhiên n tính tổng tất cả các chữ số của tất cả các số từ 0 đến n. Chúng ta sẽ viết 1 hàm tính tổng các chữ số của một số nguyên không âm cho trước từ đó áp dụng để tính tổng tất cả các chữ số trong [0, n]

```
#include<stdio.h>
int TongChuSo(int k)
{
	int s=0;
	for(;k!=0;k/=10) s+=k%10;
	return s;
}
int main()
{
	int n,i,t=0;
	printf("Nhap vao n = "); scanf("%d",&n);
	for(i=0;i<=n;i++) t+=TongChuSo(i);
	printf("Tong tat ca cac chu so tu 0 den n la %d",t);
}
```

Trong đoạn chương trình trên thì hàm TongChuSo là một đoạn chương trình độc lập thực hiện công việc là tính tổng các chữ số của số nguyên không âm k bất kỳ (k ở đây được gọi là tham số hình thức) và trả về (return) tổng các chữ số tính được chính là s;

**Ví dụ 2:** Nhập 4 số nguyên không âm a, b, c, d tìm ước chung lớn nhất của chúng
Chúng ta sẽ viết 1 hàm để tìm ước chung lớn nhất của hai số nguyên không âm bất kỳ bằng thuật chia Euclid từ đó áp dụng để tìm được ước chung lớp nhất của bốn số

```
#include<stdio.h>
int ucln(int, int);  //Khai báo nguyên mẫu hàm
int main()
{
	int a,b,c,d;
	printf("Nhap vao a,b,c,d : "); scanf("%d%d%d%d",&a,&b,&c,&d);
	printf("UCLN cua a,b,c,d la %d",ucln(ucln(a,b),ucln(c,d)));
}
int ucln(int a,int b)  //a,b là tham sô hình thức
{
	while(b!=0)
	{
		int r=a%b;
		a=b;
```

```
            b=r;
      }
      return a;
}
```

**Chú ý:** tham số a,b trong hàm ucln là tham số hình thức, còn a,b, c,d trong hàm main là tham số thực đây là các biến hoàn toàn độc lập với nhau mặc dù tên có thể trung nhau, khi ở hàm main gọi 3 hàm ucln thì mỗi lần gọi sẽ sinh ra hai biến a,b khác nhau và khi ra khỏi hàm ucln bằng câu lệnh return thì biến a,b hình thức này sẽ mất đi luôn.

**2. Bài tập mẫu**

**Bài 1.** Viết hàm kiểm tra một số có là số nguyên tố không. Áp dụng nhập vào một dãy và in ra các số nguyên tố trong dãy đó đồng thời đếm và in ra số các số nguyên tố trong dãy

*Hướng dẫn*: Để kiểm tra 1 số nguyên tố ta chỉ cần kiểm tra xem nó chia hết cho số nào từ 2 đến căn của nó không?

```
#include<stdio.h>
int ktnt(int n)  //Ham tra ve 0 neu n khong nguyen to va 1 neu
nguyen to
{
      if(n<2) return 0;
      for(int i=2;i*i<=n;i++)
      if(n%i==0) return 0;
      return 1;
}
int main()
{
      int i,n,a[100],d=0;
      printf("Nhap so phan tu cua day : "); scanf("%d",&n);
      for(i=1;i<=n;i++)
      {
            printf("a[%d] = ",i);
            scanf("%d",&a[i]);
      }
      printf("\nCac so nguyen to trong day :\n");
      for(i=1;i<=n;i++)
      if(ktnt(a[i]))
      {
            printf("%d ",a[i]);
            d++;
      }
      printf("\nSo cac so nguyen to trong day %d",d);
}
```

**Bài 2.** Viết hàm tính số fibonacci thứ n theo công thứ

$$F(n)=\begin{cases}0 & Khi\ \ n=0\\ 1 & Khi\ \ n=1\\ F(n-1)+F(n-2) & Khi\ n>1\end{cases}$$

Áp dụng nhập một dãy số a1, a2,… an các số nguyên không âm thuộc [1,100] tính giá trị biểu thức

$$S = \frac{F(a_1)}{1} + \frac{F(a_2)}{2} + \frac{F(a_3)}{3} + ... + \frac{F(a_n)}{n}$$

```
#include<stdio.h>
int Fibo(int n)  {  //Ham tra ve so Fibonacci thu n
      int F[n+2]={0,1};
      for(int i=2;i<=n;i++) F[i]=F[i-1]+F[i-2];
      return F[n];
}
int main()
{
      int i,n,a[100],d=0;
      printf("Nhap so phan tu cua day : "); scanf("%d",&n);
      for(i=1;i<=n;i++)
      {
            printf("a[%d] = ",i);
            scanf("%d",&a[i]);
      }
      float s=0;
      for(i=1;i<=n;i++) s+=Fibo(a[i])/float(i);
      printf("\nGia tri bieu thuc la %f",s);
}
```

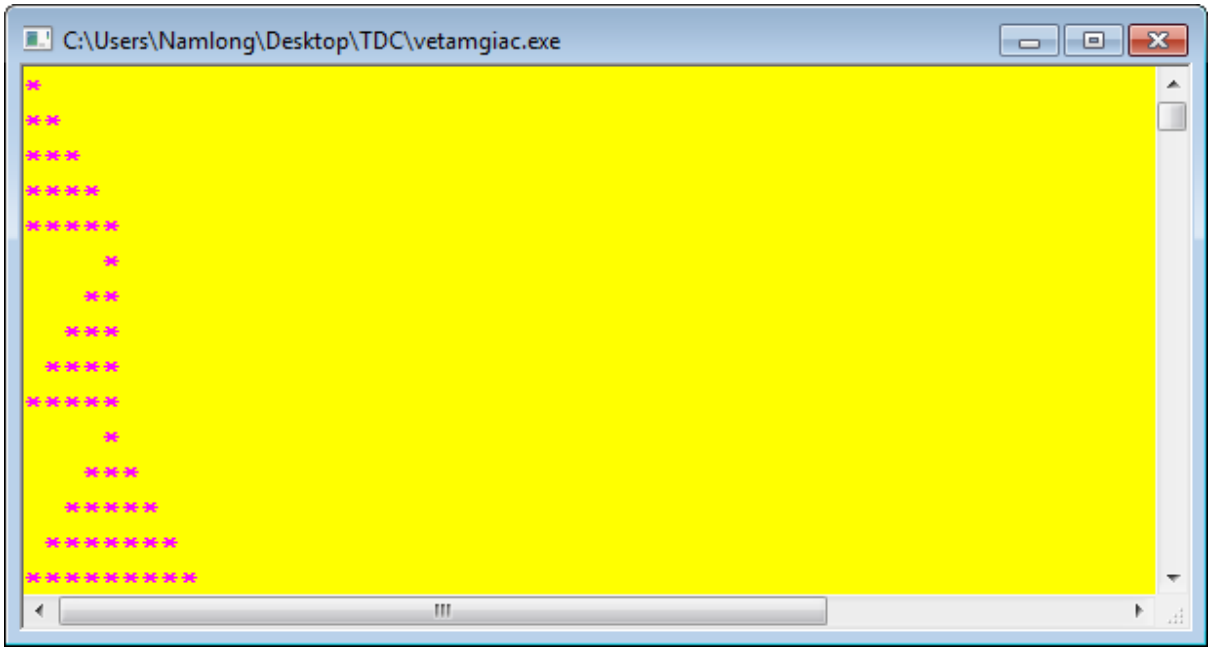
**Bài 3.** Viết hàm nhập một dãy thực và hàm tính trung bình cộng một dãy số thực. Áp dụng nhập vào hai dãy thực a1, a2, …, an gồm n phần tử và b1,b2,…bm gồm m phần tử so sánh trung bình cộng xem dãy nào lớn hơn hoặc bằng nhau.

```
#include<stdio.h>
void Nhap(float a[],int n,char ten)
//Nhap vao day a khi biet truoc so phan tu n
//Bien ten de in ra khi nhap day cho de nhan biet
{
      int i;
      for(i=1;i<=n;i++)
      {
            printf("%c[%d] = ",ten,i);
            scanf("%f",&a[i]);
      }
}
float TBC(float a[],int n) //Tinh trung binh cong day a co n phan tu
[1,2...n]
{
      float s=0;
      for(int i=1;i<=n;i++) s+=a[i];
      return s/n;
}

int main()
{
      int n,m;
```

```
	float a[100],b[100];
	printf("Nhap so phan tu day A : "); scanf("%d",&n);
	Nhap(a,n,'A');
	printf("Nhap so phan tu day B : "); scanf("%d",&m);
	Nhap(b,m,'B');
	float tA=TBC(a,n), tB=TBC(b,n);
	if(tA==tB) printf("Trung binh cong hai day bang nhau");
	else printf(tA>tB?"TBC day A lon hon B":"TBC day A nho hon B");
}
```

**Bài 4.** Viết một hàm in ra n ký tự c liên tục. Áp dụng hàm này để in ra các tam giác * trái, phải và ở giữa* ví dụ n =5




```
#include<stdio.h>
void In_day_ky_tu(int n,char c)
{
	for(int i=1;i<=n;i++) printf("%c",c);
}
void TamGiacTrai(int n)
{
	for(int i=1;i<=n;i++)
	{
		In_day_ky_tu(i,'*');
		printf("\n");
	}
}
void TamGiacPhai(int n)
{
	for(int i=1;i<=n;i++)
	{
		In_day_ky_tu(n-i,' ');
		In_day_ky_tu(i,'*');
		printf("\n");
	}
}
```

```
void TamGiacGiua(int n)
{
	for(int i=1;i<=n;i++)
	{
		In_day_ky_tu(n-i,' ');
		In_day_ky_tu(i*2-1,'*');
		printf("\n");
	}
}
int main()
{
	int n=10;
	TamGiacTrai(n);
	TamGiacPhai(n);
	TamGiacGiua(n);
}
```

**3. Bài tập**

**Bài 1:** Viết hàm tính giai thừa của số nguyên dương n và hàm tính lũy thừa $x^n$ của số thực x và số nguyên không âm n qua đó áp dụng tính giá trị gần đúng hàm $e^x$

$$e^x \approx \frac{1}{0!}+\frac{x}{1!}+\frac{x^2}{2!}+...+\frac{x^n}{n!}$$

**Bài 2:** Viết hàm kiểm tra một năm n có phải là năm nhuận dương hay không tức là có chia hết cho 400 hoặc ngược lại chia hết cho 4 và không chia hết cho 100 không, áp dụng hàm này để kiểm tra một ngày d/m/y có là ngày hợp lệ không biết rằng các tháng 1,3,5,7,8,10,12 có 31 ngày tháng 2 năm nhuận có 29 ngày còn không nhuận có 28 ngày các tháng còn lại đều 30 ngày

**Bài 3:** Viết hàm nhập vào một dãy số nguyên và hàm tìm giá trị lớn nhất của dãy số nguyên. Áp dụng nhập vào 3 dãy số nguyên tính tổng giá trị lớn nhất của 3 dãy đó

**Bài 4.** Viết hàm kiểm tra 3 số thực a, b, c có là 3 cạnh tam giác không, viết hàm tính diện tích tam giác từ 3 cạnh theo công thức Heron. Áp dụng nhập vào 1 dãy số thực a1,a2,… an. Hỏi từ dãy số thực này thì có bao nhiêu các chọn ra 3 số tạo thành 3 cạnh tam giác, tính trung bình cộng các diện tích tam giác tạo được nếu không tồn tại tam giác nào xuất ra "khong chon duoc tam giac"

**Bài 5.** Viết một hàm giải và biện luận phương trình bậc 2 với các hệ số thực a, b, c xuất ra các kết luận về nghiệm trong hàm (hàm kiểu void). Áp dụng nhập vào 3 dãy a1,a2..,an; b1,b2, …bn và c1,c2…cn đều có n phần tử với mỗi ai, bi, cj gọi hàm ở trên để giải và biện luận phương trình bậc 2 theo các hệ số đó.

**4. Bài tập làm thêm**

**Bài 1.** Viết hàm cho trước số nguyên dương n tính tổng các ước dương nhỏ hơn n của n nếu đúng bằng n thì n là số hoàn hảo còn bé hơn n thì số nghèo nàn ngược lại là số phong phú. Áp dung nhập vào dãy số a1, a2,…,an, và kết luận về tính chất này của từng số ai

**Bài 2.** Viết các hàm:

- Tìm giá trị âm lớn nhất trong mảng số nguyên, nếu không có trả về 0
- Tìm số chẵn đầu tiên trong mảng số nguyên, nếu không có giá trị chẵn thì trả về giá trị 1.

- Kiểm tra một mảng số nguyên có tăng dần không?
- Kiểm tra xem dãy này có phải là dãy cấp số cộng tăng dần không?
- Kiểm tra xem dãy có đối xứng không?
- Đếm số cặp hai số liên tiếp trong dãy mà có tổng chẵn
- Đếm số cặp 2 số ở 2 vị trí bất kỳ khác nhau có tổng đúng bằng 10
- Liệt kê các số lẻ trong dãy
- Liệt kê các cặp 2 số liên tiếp mà số sau lớn hơn số trước
- Tính giá trị của biểu thức $S = a_1a_n + a_2a_{n-1} + ... + a_{n-1}a_2 + a_na_1$

Hàm main nhập vào một dãy số nguyên và thực hiện tất cả các thao tác ở trên

# Bài 5: Cấu trúc

**1. Mục tiêu:** Hiểu và vận dụng kiểu cấu trúc để giải quyết bài toán có kiểu dữ liệu gồm nhiều thành phần.

**2. Nội dung**

***1. Định nghĩa***

```
typedef struct <Tên kiểu cấu trúc>
{
     <kiểu 1>  <thuộc tính 1>;
     <kiểu 2>  <thuộc tính 2>;
     ....
};
```

**Ví dụ1:** Định nghĩa kiểu cấu trúc Phân số gồm các thuộc tính: tử số, mẫu số.

```
typedef struct PS
{
     int ts;
     int ms;
};
```

hoặc

```
typedef struct PS
{
     int ts, ms;
};
```

**Ví dụ 2:** Định nghĩa cấu trúc Sinhviên gồm các thành phần: mã sinh viên, họ tên, điểm

```
typedef struct Sinhvien
{
     char  masv[10],hoten[25];
     float diem;
};
```

***2. Biến, mảng cấu trúc***

Sau khi đã định nghĩa kiểu cấu trúc ta có thể khai báo các biến, mảng có kiểu cấu trúc như khai báo các biến, mảng có kiểu int, float…

```
<kiểu cấu trúc>  <danh sach biến cấu trúc>;
<kiểu cấu trúc>  <mảng cấu trúc>[số_phần_tử_max];
```

**Ví dụ:**
- Khai báo 2 biến cấu trúc phân số: PS p1, p2;
- Khai báo hai biến có kiểu sinh viên:SV sv1, sv2;
- Khai báo mảng 100 sinh viên thì ta viết là: SV sv[100];
- Khai báo 10 phân số thì ta viết là: PS p[10];

Sinh viên thứ i là sv[i], phân số thứ 3 là p[3].

**Chú ý:** Chúng ta có thể khai báo kiểu cấu trúc đồng thời khai báo biến theo cú pháp như sau :

```
struct Tênkiểucấutrúc
{
      <kiểu 1>  <thuộc tính 1>;
      <kiểu 2>  <thuộc tính 2>;
      ....
}danh_sách_các_biến;
```

***3. Truy xuất***

```
<Biến cấu trúc>.<Thành phần>
```

**Ví dụ:**
- p.ts được hiểu là tử số của phân số p
- sv2.diem nghĩa là điểm của sinh viên sv2;
- sv[i].ht tức là họ tên của sinh viên thứ i của mảng sinh viên sv.

***4. Một số ví dụ***

**Ví dụ 1:** Định nghĩa cấu trúc Điểm (hoành độ, tung độ). Nhập vào một điểm và cho biết điểm đó có thuộc góc phần tư thứ nhất hay không?

**Chú ý:** khi nhập các thuộc tính thực của biến cấu trúc thì ta phải nhập qua biến trung gian.

```
#include”stdio.h”
#include”conio.h”
typedef struct Diem{ float x,y;};
int main() {
      Diem M;
      float hd,td;
      printf(“nhap toa do cho diem M:”);
      scanf(“%f%f”,&hd,&td);//nhap qua bien trung gian
      M.x=hd;M.y=td;
      if(M.x>0 && M.y>0)
            printf(“diem M thuoc goc phan tu thu nhat”);
      else
            printf(“Diem M khong thuoc goc phan tu thu nhat”);

}
```

**Ví dụ 2:** Định nghĩa cấu trúc Điểm (hoành độ, tung độ). Nhập vào một n điểm và cho biết có bao nhiêu điểm thuộc góc phần tư thứ nhất?

```
#include”stdio.h”
#include”conio.h”
typedef struct Diem{ float x,y;};
int main()
{     Diem M[50]; int i,n, Dem;float hd,td;
      printf(“nhap so diem:”);
      scanf(“%d”,&hd,&td);
      for(i=1;i<=n;i++)
      {     printf(“nhap toa do cho diem thu %d:”,i);
            scanf(“%f%f”,&hd,&td);
            M[i].x=hd;M[i].y=td;
```

```
      }
      Dem=0;
      for(i=1;i<=n;i++)
            if(M[i].x>0 && M[i].y>0) Dem++;
      printf("\nCo %d diem thuoc goc phan tu thu nhat",Dem);
}
```

**Ví dụ 3:** Định nghĩa cấu trúc Tuyển sinh (số báo danh, họ tên, điểm toán, điểm lý, điểm hoá). Hãy nhập vào n thí sinh. Nhập điểm chuẩn và in ra danh sách thí sinh trúng tuyển. Cho biết một thí sinh có điểm toán cao nhất. In ra 3 thí sinh có tổng điểm cao nhất

```
#include<stdio.h>
#include<conio.h>
typedef struct TS{
      char sobd[10],hten[30];
      float toan,ly,hoa,td;
      };
int main() {
      TS ts[100], tg,max;    int i,j,n;    float dc, t,l,h;
      printf("n=");scanf("%d",&n);
      for (i=1;i<=n;i++)
      {
            printf("nhap thong tin cho thi sinh thu %d:",i);
            printf("so bao danh:");    fflush(stdin); gets(ts[i].sobd);
            printf("ho ten:");         fflush(stdin);  gets(ts[i].hten);
            printf("nhap diem toan, ly hoa:");
            scanf("%f%f%f",&t,&l,&h);
            ts[i].toan=t;   ts[i].ly=t;ts[i].hoa=h; ts[i].td=t+l+h;
      }
      //nhap dien chuan va in ra ds thi sinh trung tuyen
      printf("nhap diem chuan:");scanf("%f",&dc);
      puts("danh sach thi sinh trung tuyen:");
      for (i=1;i<=n;i++)
      if(ts[i].td>=dc&&ts[i].toan*ts[i].ly*ts[i].ly!=0)
      printf("%s  %s%.2f %.2f %.2f
      %.2f",ts[i].sobd,ts[i].hten,ts[i].toan,ts[i].ly,ts[i].hoa,ts[i].td);

      //tim thi sinh co diem toan cao nhat
      max=ts[1];
      for (i=2;i<=n;i++)
      if (max.toan<ts[i].toan)
            max=ts[i];
      printf("\nThong tin ve 1 sinh vien co diem toan cao nhat:");
      printf("\nHo ten: %s",max.hoten);
      printf("Toan  : %3.1f\n",max.toan);
      //in ra 3 thi sinh co tong diem cao nhat
      //sap xep theo thu tu giam dan cua tong diem
      for(i=1;i<=n-1;i++)
        for(j=i+1;j<=n;j++)
          if(ts[j].td>ts[i].td)
            {tg=ts[i];ts[i]=ts[j];ts[j]=tg;}
      puts("3 thi sin co diem cao nhat la:");
      for(i=1 ;i<=3;i++)
```

```
        printf("%s  %s   %.2f    %.2f   %.2f %.2f",
            ts[i].sobd,ts[i].hten,ts[i].toan,ts[i].ly,ts[i].hoa,ts[i].td);
}
```

**Ví dụ 4:** Định nghĩa cấu trúc Điểm(hoành độ, tung độ, mầu:0-xanh,1-đỏ,2-vàng). Hãy nhập vào n điểm. Tính độ dài đường gấp khúc lần lượt đi qua các điểm thứ 1,2,..n. Đếm số điểm màu vàng và in các điểm đó.Tìm một điểm đỏ xa gốc toạ độ nhất.

```
#include<stdio.h>
#include<conio.h>
typedef struct Diem{
      float x,y; int mau;
      };
int main()
{
      Diem d[100],max;
      int i,d,n,m;
      float hd,td,s;
      printf(“n=”); scanf(“%d”,&n);
      for (i=1;i<=n;i++)
       {
            printf(“nhap thong tin cho diem  thu %d:”,i);
            printf(“nhap toa do x,y,mau:”);
            scanf(“%f%f”,&hd,&td,&m);
            d[i].x=hd;
            d[i].y=td;
            d[i].mau=m;
      }
      //Tinh do dai duong gap khuc
      s=0;
      for (i=1;i<=n-1;i++) s+= sqrt(pow(d[i].x-d[i+1].x,2)+pow(d[i].y-
                                                      d[i+1].y,2));
      printf(“\n do dai duong gap khuc:%.2f”,s);
      //dem so diem vang
      d=0;
      for (i=1;i<=n;i++) if(d[i].mau==2) {
            d++;printf(“(%.2f, %.2f)  “,d[i].x,d[i].y);
     }
      printf(“\n so diem vang :%d“,d);
      //tim diem do xa goc toa nhat
      //dem so diem do, neu co thi gan max
      d=0;
      for (i=1;i<=n;i++) if(d[i].mau==1) {
            d++;
            max=d[i];
       }
      if(d==0) puts("khong co diem do nao");
      else {
      for(i=1 ;i<=n;i++)
        if(d[i].mau==1 &&
          sqrt(d[i].x*d[i].x+d[i].y*d[i].y)>sqrt(max.x*max.x+max.y*max.y))
             max=d[i];
```

```
        printf("diem do xa goc toa do nhat (%.2,%.2f) kc= %.2f", max.x, max.y,
        sqrt(max.x*max.x+max.y*max.y));
}
```

**Bài tập**

1. Định nghĩa cấu trúc Số Phức gồm hai thành phần là phần thực và phần ảo. Hãy nhập vào 2 số phức, in ra màn hình 2 số phức vừa nhập và tổng, hiệu, tích, thương của 2 số phức đó.

2. Tổ chức dữ liệu để quản lý sinh viên bằng cấu trúc trong một mảng n phần tử, mỗi sinh viên có cấu trúc gồm các thành phần: mã sinh viên, họ tên, điểm trung bình):

Viết chương trình thực hiện những công việc sau:

- Nhập danh sách các sinh viên cho một lớp học.
- Xuất danh sách sinh viên ra màn hình.
- Tìm sinh viên có điểm trung bình cao nhất.
- Sắp xếp danh sách lớp theo thứ tự tăng dần của điểm trung bình.
- Tìm kiếm và in ra các sinh viên có điểm trung bình lớn hơn 5.
- Nhập vào mã sinh viên. Tìm và in ra các thông tin liên quan đến sinh viên đó (nếu có).

3. Xây dựng một cấu trúc ứng với phiếu điểm của thí sinh thi học sinh giỏi gồm các thành phần:Họ tên,Quê quán,Trường(giả sử có 3 trường dự thi là A, B, C), Số báo danh, Điểm Toán, Lý, Hoá.Viết chương trình thực hiện:

- Nhập số liệu của 20 phiếu điểm và lưu trữ vào mảng cấu trúc nói trên.
- Tìm kiếm và in ra các thí sinh có tổng số điểm ba môn lớn hơn 15.
- Sắp xếp lại các phần tử của mảng cấu trúc theo thứ tự giảm dần của tổng số điểm, sau đó in danh sách thí sinh.
- Tính tổng điểm Toán theo trường

4. Viết chương trình nhập vào một dãy điểm, mỗi điểm là một cấu trúc gồm hoành độ và tung độ rồi thực hiện:

//chẵn

- Đếm số điểm nằm trong (ngoài, trên biên) đường tròn bán kính R (R nhập từ bàn phím).
- Tính tổng khoảng cách của các điểm đến gốc tọa độ (trục hoành, trục tung)
- Tính độ dài đường gấp khúc lần lượt đi qua các điểm thứ 1,2..n
- Có bao nhiêu đoạn cắt trục tung.
- Tìm điểm gần (xa) gốc tọa độ (trục hoành, trục tung) nhất

5. Định nghĩa cấu trúc Đa thức gồm số nguyên n biểu diễn bậc của đa thức, mảng thực để lưu trữ các hệ số của đa thức. Viết chương trình nhập đa thức P bậc n, đa thức Q bậc m và số thực d.In các đa thức P, Q. Tính P(d)/Q(d).

6. Định nghĩa cấu trúc ma trận gồm hai số nguyên lưu trữ số hàng và số cột của ma trận và một mảng nhiều chiều để lưu trữ các phần tử của ma trận. Viết chương trình thực hiện:

- Nhập ma trận

- In ma trận

- Tìm giá trị lớn nhất của ma trận

- Tính giá trị trung bình các phần tử của ma trận

---------- **HẾT** ----------