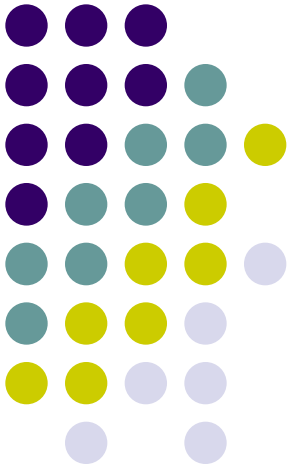
LẬP TRÌNH CĂN BẢN

Phần 2 - Chương 6

KIỂU MẢNG



Nội dung chương này

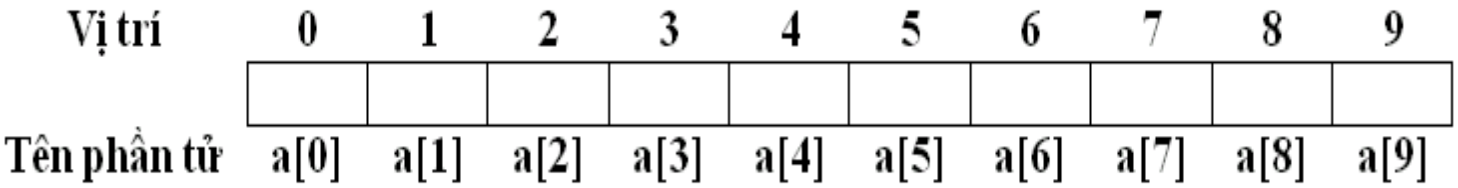
- Giới thiệu kiểu mảng trong C
- Mảng 1 chiều
- Mảng nhiều chiều

Giới thiệu kiểu mảng trong C (1)

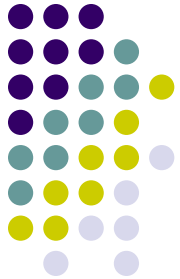
- **Ví dụ:**

```
int  a[10];
```

=> Hình ảnh của a trong bộ nhớ như sau:



Giới thiệu kiểu mảng trong C (2)



- "Mảng là một tập hợp các phần tử cố định có cùng một kiểu, gọi là kiểu phần tử".

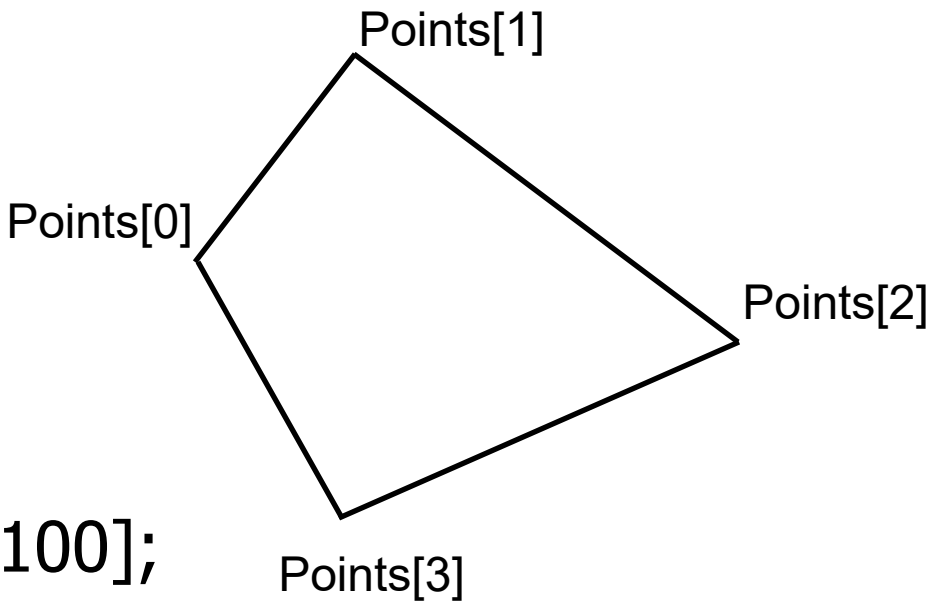
- Kiểu phần tử có thể là có kiểu bất kỳ:
  - ký tự
  - số
  - 1 struct
  - 1 mảng khác (=> mảng của mảng hay mảng nhiều chiều)
  - …;

Giới thiệu kiểu mảng trong C (3)

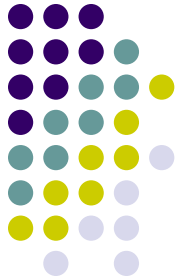
- Ví dụ: Lưu trữ 1 đa giác trong đồ họa:

```
typedef  struct {
        int     x;
        int     y;
        } Point;

typedef  struct{
        Point  Points[100];
        int  nPoints;
   }  Polygon;
```

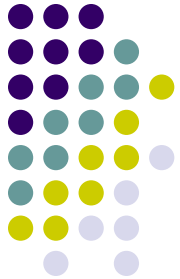


Giới thiệu kiểu mảng trong C (4)



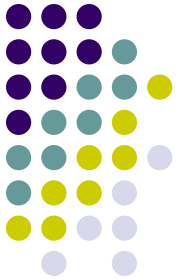
- Ta có thể chia mảng làm 2 loại:
  - Mảng 1 chiều
  - Mảng nhiều chiều

Mảng 1 chiều (1)



- Xét dưới góc độ toán học, mảng 1 chiều giống như một vector.
- Mỗi phần tử của mảng 1 chiều có giá trị *không phải là một mảng khác*.
- ***Khai báo mảng với số phần tử xác định***
  - **Ví dụ:** **float a[100];**
  - **Cú pháp:** **<Kiểu> <Tên mảng ><[số phần tử]>;**
- ***Khai báo mảng với số phần tử không xác định***
  - **Ví dụ:** **float a[];**
  - **Cú pháp:** **<Kiểu> <Tên mảng> <[]> ;**

Mảng 1 chiều (2)



- ***Vừa khai báo vừa gán giá trị***

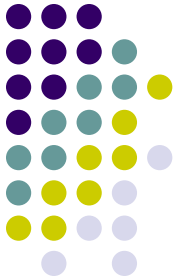
  <Kiểu> <Tên mảng> []= {Các giá trị cách nhau bởi dấu phẩy} ;

  => Số phần tử có thể được xác định bằng sizeof()

  Số phần tử=sizeof(tên mảng)/sizeof(kiểu)

- ***Khai báo mảng là tham số hình thức của hàm***
  - ***không cần chỉ định số phần tử của mảng là bao nhiêu***

Mảng 1 chiều (3)



- **Ví dụ:** Gán giá trị ngay lúc khai báo

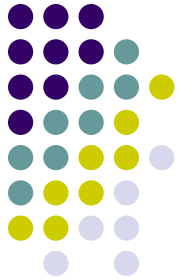
```
int primes[] = {2,3,5,7,11,13};
```

Sẽ tương đương với:

```
int primes[6];
primes[0] = 2;
primes[1] = 3;
primes[2] = 5;
primes[3] = 7;
primes[4] = 11;
primes[5] = 13;
```

=>sizeof(primes)/sizeof(int)=6

Truy xuất từng phần tử của mảng (1)



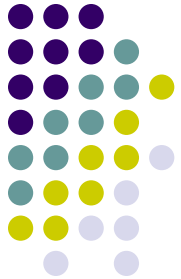
- **Cú pháp:**

Tên biến mảng[Chỉ số]

- **Ví dụ 1:**

```
int a[10];
a[0]=5;  a[1]=5;  a[2]=33;  a[3]=33;  a[4]=15;
printf("%d  %d  %d  %d  %d", a[0], a[1], a[2], a[3], a[4]);
```

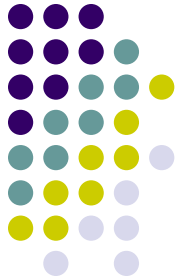
Truy xuất từng phần tử của mảng (2)



- **Ví dụ 2:** Vừa khai báo vừa gán trị cho 1 mảng 1 chiều các số nguyên. In mảng số nguyên này lên màn hình.

```
#include <stdio.h>
#include <conio.h>
int main(){
      int n,i,j,tam;
      int dayso[]={66,65,69,68,67,70};
      clrscr();
      n=sizeof(dayso)/sizeof(int); /*Lấy số phần tử*/
      printf("\n Noi dung cua mang ");
      for(i=0;i<n;i++)
          printf("%d ",dayso[i]);
      return 0;
}
```

Truy xuất từng phần tử của mảng (3)



- **Ví dụ 3**: Đổi một số nguyên dương thập phân thành số nhị phân.

```
#include <conio.h>
#include <stdio.h>
int main(){
      unsigned int N;
      printf("Nhap vao so nguyen N= ");scanf("%d",&N);
      unsigned int Du;
      unsigned int NhiPhan[20],K=0;
      do{
            Du=N % 2;
            NhiPhan[K]=Du; // Luu số dư vào mảng ở vị trí K
            K++;  // Tăng K lên để lần kế lưu vào vị trí kế
            N = N/2;
      }while(N>0);

      printf("Dang nhi phan la: ");
      for(int i=K-1;i>=0;i--)
            printf("%d",NhiPhan[i]);
      getch();
      return 0;
}
```

Truy xuất từng phần tử của mảng (4)

- **Ví dụ 4**: Nhập vào một dãy n số và sắp xếp các số theo thứ tự tăng.

```
#include<conio.h>
#include<stdio.h>
void Nhap(int a[],int N){
  for(int i=0; i< N; i++){
        printf("Phan tu thu %d: ",i);
        scanf("%d",&a[i]);
  }
}
void InMang(int a[], int N){
  for (int i=0; i<N;i++)
        printf("%d ",a[i]);
  printf("\n");
}
void SapXep(int a[], int N){
  int t;
  for(int i=0;i<N-1;i++)
        for(int j=i+1;j<N;j++)
              if(a[i]>a[j]){
                    t=a[i];
                    a[i]=a[j];
                    a[j]=t;
              }
}
```

```
int main()
{
  int b[20], N;
  printf("So phan tu thuc te cua mang N= ");
  scanf("%d",&N);
  Nhap(b,N);
  printf("Mang vua nhap: ");
  InMang(b,N);
  SapXep(b,N); // Gọi hàm sắp xếp
  printf("Mang sau khi sap xep: ");
  InMang(b,N);
  getch();
  return 0;
}
```

Kết quả chạy chương trình có thể là:

```
TC.EXE
So phan tu thuc te cua mang N= 4
Phan tu thu 0: 2
Phan tu thu 1: 6
Phan tu thu 2: 0
Phan tu thu 3: -1
Mang vua nhap: 2 6 0 -1
Mang sau khi sap xep: -1 0 2 6
```

Truy xuất từng phần tử của mảng (5)

- **Ví dụ 5**: Chương trình sau sẽ hiển thị kết quả gì?

```
#include <stdio.h>

int a[12], b;

main()
{
    int i;
    b = 5;
    printf("b=%d\n", b);

    for (i=0; i<=12; i++) {
        a[i] = i;
    }
    printf("b=%d\n", b);
}
```

Output:

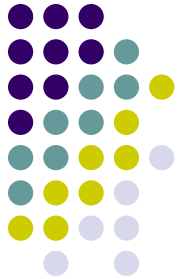
```
b=5
b=12
```

Sai gì ở đây?

Sửa lỗi này thế nào?

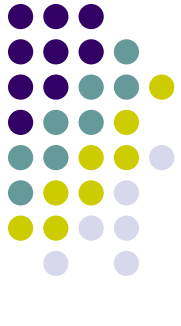
Các phần tử của mảng a[0], …, a[11]. Việc truy cập a[12] sẽ vượt ra bên ngoài mảng, ô nhớ của biến b.

Mảng nhiều chiều



- Mảng nhiều chiều là mảng có từ 2 chiều trở lên.
- Điều đó có nghĩa là mỗi phần tử của mảng là một mảng khác.
- Người ta thường sử dụng mảng nhiều chiều để lưu các ma trận, các tọa độ 2 chiều, 3 chiều…

Khai báo mảng 2 chiều tường minh



- **Cú pháp:**

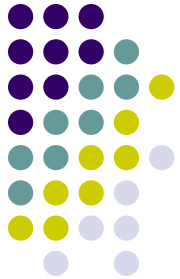
  <Kiểu> <Tên mảng><[Số phần tử chiều 1]><[Số phần tử chiều 2]> ;

- **Ví dụ:**

  float m[8][9]; // mảng 2 chiều có 8*9 phần tử là số thực

Dòng\Cột	0	1	2	3	4	5	6	7	8
0	m[0][0]	m[0][1]	m[0][2]	m[0][3]	m[0][4]	m[0][5]	m[0][6]	m[0][7]	m[0][8]
1	m[1][0]	m[1][1]	m[1][2]	m[1][3]	m[1][4]	m[1][5]	m[1][6]	m[1][7]	m[1][8]
2	m[2][0]	m[2][1]	m[2][2]	m[2][3]	m[2][4]	m[2][5]	m[2][6]	m[2][7]	m[2][8]
3	m[3][0]	m[3][1]	m[3][2]	m[3][3]	m[3][4]	m[3][5]	m[3][6]	m[3][7]	m[3][8]
4	m[4][0]	m[4][1]	m[4][2]	m[4][3]	m[4][4]	m[4][5]	m[4][6]	m[4][7]	m[4][8]
5	m[5][0]	m[5][1]	m[5][2]	m[5][3]	m[5][4]	m[5][5]	m[5][6]	m[5][7]	m[5][8]
6	m[6][0]	m[6][1]	m[6][2]	m[6][3]	m[6][4]	m[6][5]	m[6][6]	m[6][7]	m[6][8]
7	m[7][0]	m[7][1]	m[7][2]	m[7][3]	m[7][4]	m[7][5]	m[7][6]	m[7][7]	m[7][8]

Khai báo mảng 2 chiều không tường minh



- Để khai báo mảng 2 chiều không tường minh, ta vẫn phải chỉ ra số phần tử của chiều thứ hai (chiều cuối cùng).

- **Cú pháp:**

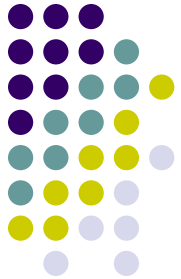
  <Kiểu> <Tên mảng> <[]><[Số phần tử chiều 2]>;

- Ví dụ:

  float m[][9];

- Cách khai báo này cũng được áp dụng trong trường hợp:
  - vừa khai báo vừa gán trị
  - mảng 2 chiều là tham số hình thức của 1 hàm.

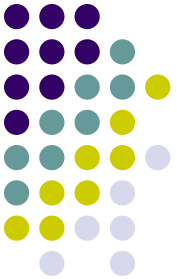
Truy xuất từng phần tử của mảng 2 chiều



- **Dùng:**

Tên mảng[Chỉ số 1][Chỉ số 2]

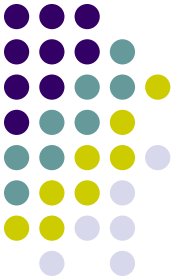
Ví dụ (1)



- Viết chương trình cho phép nhập 2 ma trận a, b có m dòng n cột, thực hiện phép toán cộng hai ma trận a,b và in ma trận kết quả lên màn hình.

```
#include<conio.h>
#include<stdio.h>
void Nhap(int a[][10],int M,int N){
      for(int i=0;i<M;i++)
            for(int j=0; j<N; j++){
                  printf("Phan tu o dong %d cot %d: ",i,j);
                  scanf("%d",&a[i][j]);
            }
}
void InMaTran(int a[][10], int M, int N){
      for(int i=0;i<M;i++){
            for(int j=0; j< N; j++)
                  printf("%d ",a[i][j]);
            printf("\n");
      }
}
// Cong 2 ma tran A & B ket qua la ma tran C
void CongMaTran(int a[][10],int b[][10],int M,int N,int c[][10]){
      for(int i=0;i<M;i++)
            for(int j=0; j<N; j++)
                  c[i][j]=a[i][j]+b[i][j];
}
```

Ví dụ (2)



```
int main(){
      int a[10][10], b[10][10], M, N;
      printf("So dong M= "); scanf("%d",&M);
      printf("So cot M= "); scanf("%d",&N);
      printf("Nhap ma tran A\n");
      Nhap(a,M,N);
      printf("Nhap ma tran B\n");
      Nhap(b,M,N);
      printf("Ma tran A: \n");
      InMaTran(a,M,N);
      printf("Ma tran B: \n");
      InMaTran(b,M,N);

      int c[10][10];// Ma tran tong
      CongMaTran(a,b,M,N,c);
      printf("Ma tran tong C:\n");
      InMaTran(c,M,N);
      getch();
      return 0;
}
```


Hết chương