# CHAPTER 5. STRUCTURES

# Nội dung

1.Giới thiệu kiểu dữ liệu có cấu trúc

2.Khai báo biến thuộc kiểu dữ liệu có cấu trúc

3.Truy xuất đến thành phần của kiểu dữ liệu có cấu trúc

4.Ví dụ và thảo luận

# 1.GIỚI THIỆU KIỂU DỮ LIỆU CÓ CẤU TRÚC

- Ngoài các kiểu dữ liệu chuẩn, các ngôn ngữ lập trình cung cấp cơ chế để người lập trình tự tạo ra/tự định nghĩa thêm các kiểu dữ liệu mới; đó là kiểu dữ liệu có cấu trúc.
- Một biến thuộc kiểu dữ liệu có cấu trúc thì thường có nhiều thành phần cần quản lý; mà mỗi thành phần trong đó thuộc về các kiểu dữ liệu chuẩn hoặc thuộc về một kiểu dữ liệu có cấu trúc đã được định nghĩa trước đó.

Xem xét một số ngữ cảnh sau:

- Để giải quyết bài toán mảng hai chiều mà mỗi phần tử của mảng là một phân số thì cần tổ chức các biến như thế nào ?
- Để giải quyết bài toán quản lý nhân sự thì cần tổ chức các biến như thế nào ?
- Để giải quyết bài toán liên quan đến các điểm trong mặt phẳng tọa độ OXY thì cần tổ chức các biến như thế nào ?
- Để giải quyết bài toán liên quan đến các đa thức thì cần tổ chức các biến như thế nào ?

❖ Việc xây dựng lên các kiểu dữ liệu có cấu trúc là cần thiết

# 2. SỬ DỤNG BIẾN THUỘC KIỂU DỮ LIỆU CÓ CẤU TRÚC

Cú pháp khai báo một kiểu dữ liệu có cấu trúc

```
struct tên_kiểu_cấu_trúc
{   //khai báo các thành phần của cấu trúc
    <kiểu dữ liệu> <tên thuộc tính 1>
    <kiểu dữ liệu> <tên thuộc tính 2>
    <kiểu dữ liệu> <tên thuộc tính 3>
    ...
};
```

# Ví dụ

```
struct ngaysinh
        {
                int ngay;
                int thang;
                int nam;
        };
```

```
struct hoso
        {
        char hoten [30];
        struct ngaysinh ns;
        long luongcoban;
        long thuong;
        long thuclanh;
        };
```

# KHAI BÁO BIẾN THUỘC KIỂU DỮ LIỆU CÓ CẤU TRÚC

<kiểu cấu trúc> <tên biến cấu trúc>

Chẳng hạn:

hoso nv[10000];

# TRUY XUẤT ĐẾN THÀNH PHẦN CỦA CẤU TRÚC

## Biến cấu trúc được khai báo tĩnh

Với cấu trúc đơn thì theo cú pháp sau:

tên biến cấu trúc.tên thành phần

Ví dụ:

```
nv[i].hoten;      // chỉ họ tên của nhân viên thứ i
nv[i].thuong;     // chỉ thưởng của nhân viên thứ i
```

Với cấu trúc phức thì theo cú pháp sau:

tên biến cấu trúc.tên biến cấu trúc.tên thành phần

Ví dụ:

nv[i]. ns.ngay;

nv[i]. ns.thang;

nv[i]. ns.nam;

## Biến cấu trúc được khai báo động

tên biến cấu trúc ->tên thành phần

# 3. MỘT SỐ VÍ DỤ ÁP DỤNG

## Ví dụ 1.

Viết chương trình cộng, trừ, nhân, chia hai phân số a/b và c/d trong đó a, b, c, d là các số nguyên. Yêu cầu phân số kết quả phải ở dạng tối giản.

Tương tự. Giải bài toán tính tổng, hiệu, tích, thương của 2 số phức.

# Gợi ý về thiết kế

```
struct phanso
{
        int tu;
        int mau;
};
void nhapps(phanso &ps);
void xuatps(phanso ps);
phanso congps(phanso ps1, phanso ps2);                  // Cach 1
//void congps(phanso ps1, phanso ps2, phanso &ps);      // Cach 2
//void congps(phanso ps1, phanso ps2);                  // Cach 3
phanso trups(phanso ps1, phanso ps2);
phanso nhanps(phanso ps1, phanso ps2);
phanso chiaps(phanso ps1, phanso ps2);
```

# Ví dụ 2.

Viết chương trình thực hiện các công việc sau:

**a.** Nhập vào hồ sơ của *n* nhân viên với các thông tin: họ và tên, ngày sinh, lương cơ bản, thưởng, thực lãnh; trong đó thực lãnh = lương cơ bản + thưởng.

**b**. In danh sách nhân viên theo thực lĩnh giảm dần.

(Câu hỏi SV tìm hiểu thêm: Sắp xếp danh sách theo thứ tự họ và tên tăng dần)

# Gợi ý về thiết kế

```
struct ngaysinh
{
        int ngay;
        int thang;
        int nam;
};
```

```
struct hoso
{
        char hoten[32];
        ngaysinh ns;
        long luong;
        long thuong;
        long thuclanh;
};
```

```
void nhap(hoso hs[], int &n);
void xuat(hoso hs[], int n);
void swap(hoso &hs1, int  &hs2);
void sapxep(hoso hs[], int n);
```

# Ví dụ 3.

Trong mặt phẳng tọa độ *OXY*, cho *n* điểm $P_i$ ($x_i$,$y_i$); tọa độ mỗi điểm $p_i$ là các số nguyên.

a.Hãy viết chương trình đếm xem có bao nhiêu điểm thuộc về mỗi góc phần tư ? Có bao nhiêu điểm nằm trên các trục tọa độ ?

(Ghi chú: I. X>0, y>0;  II.x<0, y>0; III.x<0, y<0; IV. X>0, y<0; nằm trên các trục tọa độ: có hoành độ bằng 0 hoặc có tung độ bằng 0).

b.Tìm cặp điểm gần nhau nhất

# Gợi ý về thiết kế

```
struct point
{
        int x;
        int y;
};
void nhap(point P[], int &n);
void xuly(point P[], int n);
void capdiemgannhat(point P[], int n);
```

# Hết