



Chương 8

Kiểu dữ liệu cấu trúc

(structure)

Khoa Hệ thống thông tin quản lý

Hà Nội – 2015

Nội dung

1

Khái niệm kiểu cấu trúc (struct)

2

Các thao tác với cấu trúc

3

Mảng cấu trúc

4

Con trỏ cấu trúc

5

Chuyển tham số struct cho hàm

6

Union

1. Khái niệm kiểu cấu trúc

☐ Kiểu cấu trúc (struct)

- Là kiểu dữ liệu bao gồm nhiều thành phần có kiểu khác nhau, mỗi thành phần được gọi là một trường (field)
- Kiểu cấu trúc và mảng:
 - ☐ Các phần tử của mảng là cùng kiểu
 - ☐ Các phần tử của cấu trúc có thể có kiểu khác nhau
- Struct được dùng để định nghĩa các kiểu dữ liệu mới

Khai báo cấu trúc

□ Khai báo trực tiếp

```
struct <tên kiểu cấu trúc>
{
    <kiểu 1> <trường 1>;
    ...
    <kiểu n> <trường n>;
} <tên biến 1>, <tên biến 2>;
```

□ Ví dụ

- Khai báo cấu trúc NgayThang gồm 3 trường: ngày, tháng, năm

```
struct NgayThang
{
    unsigned char Ngay;
    unsigned char Thang;
    unsigned int Nam;
} date1, date2;
```

Khai báo cấu trúc (tt)

□ Khai báo gián tiếp

```
typedef struct
{
    <kiểu 1> <trường 1>;
    ...
    <kiểu n> <trường n>;
} <tên kiểu cấu trúc>;
```

□ Ví dụ

- Khai báo kiểu cấu trúc NgayThang gồm 3 trường: ngày, tháng, năm

```
typedef struct
{
    unsigned char Ngay;
    unsigned char Thang;
    unsigned int Nam;
} NgayThang;
```

Khai báo cấu trúc lồng nhau

□ Ví dụ:

- Khai báo cấu trúc SinhVien gồm: mã sinh viên, họ tên, ngày sinh (thuộc kiểu ngaythang ở trên), giới tính, địa chỉ

Khai báo trực tiếp

```
struct SinhVien
{
    char Masv[10];
    char Hoten[40];
    NgayThang NgaySinh;
    int Gioitinh;
    char Diachi[50];
};
```

Khai báo gián tiếp

```
typedef struct
{
    char Masv[10];
    char Hoten[40];
    NgayThang NgaySinh;
    int Gioitinh;
    char Diachi[50];
} SinhVien;
```

Khai báo biến kiểu cấu trúc

□ Khai báo biến kiểu cấu trúc

- Khai báo tương tự như khai báo biến thuộc kiểu dữ liệu chuẩn
- Với cách khai báo cấu trúc trực tiếp, có thể khai báo biến ngay khi khai báo cấu trúc
- Ví dụ 1: Khai báo biến A và B

```
struct Diem
{
    float x;
    float y;
} A,B;
```

- Ví dụ 2: Khai báo biến SV1,SV2 có kiểu SinhVien
- ```
SinhVien SV1,SV2;
```

## 2. Các thao tác với cấu trúc

---

### □ Khởi tạo cấu trúc

- Biến cấu trúc có thể được khởi tạo giá trị trong lúc khai báo.
- Các trường của cấu trúc được đặt giữa cặp dấu { và }, ngăn cách bằng dấu phẩy (,)
- Ví dụ:

Khởi tạo biến cấu trúc ngaysinh

```
struct NgayThang NgaySinh={01,08,1991}
```



# Truy cập vào phần tử struct

## ❑ Đặc điểm

- Không thể truy xuất trực tiếp
- Thông qua toán tử thành phần cấu trúc . hay còn gọi là **toán tử chấm** (dot operation)

## ❑ Cú pháp

`<tên biến cấu trúc>.<tên trường>`

## ❑ Ví dụ:

- Viết ra tọa độ điểm A trong khai báo trên

```
printf("x = %f, y = %f", A.x, A.y);
```

## ❑ Chú ý:

- Các biến cấu trúc có thể gán cho nhau, vd: `B=A;`
- **KHÔNG** thực hiện được các hàm nhập xuất, các phép quan hệ, số học, logic trên biến cấu trúc

# Gán dữ liệu kiểu cấu trúc

## □ Có 2 cách

`<biến cấu trúc đích> = <biến cấu trúc nguồn>;`

`<biến cấu trúc đích>.<tên thành phần> = <giá trị>;`

## □ Ví dụ

```
struct Diem
{
 int x, y;
} diem1 = {2912, 1706}, diem2;
...
diem2 = diem1;
diem2.x = diem1.x;
diem2.y = diem1.y * 2;
```

# Cấu trúc phức tạp

## □ Cấu trúc đệ quy (tự trỏ)

```
struct PERSON
{
 char hoten[30];
 struct PERSON *father, *mother;
};
```

```
struct NODE
{
 int value;
 struct NODE *pNext;
};
```

- Sử dụng để tạo danh sách liên kết (đơn – LIFO, FIFO, kép, vòng)

# Ví dụ - Nhập/in thông tin sinh viên

---

- ❑ Viết chương trình nhập từ bàn phím thông tin về 1 sinh viên, in các thông tin đó ra màn hình

```
typedef struct
{
 unsigned char Ngay;
 unsigned char Thang;
 unsigned int Nam;
} NgayThang;

typedef struct
{
 char MSSV[10];
 char HoTen[40];
 NgayThang NgaySinh;
 int Gioitinh;
 char DiaChi[40];
} SinhVien;
```

# Ví dụ - Nhập/in thông tin sinh viên

```
void InSV(SinhVien s)
{ printf("MSSV: | Ho va ten | Ngay Sinh | Dia chi\n");
 printf("%s | %s | %d-%d-%d | %s\n", s.MSSV, s.HoTen,
s.NgaySinh.Ngay, s.NgaySinh.Thang, s.NgaySinh.Nam, s.DiaChi);
}
int main()
{ SinhVien SV1, SV2;
 printf("Nhap MSSV: "); gets(SV1.MSSV);
 printf("Nhap Ho va ten: "); gets(SV1.HoTen);
 printf("Sinh ngay: "); scanf("%d", &SV1.NgaySinh.Ngay);
 printf("Thang: "); scanf("%d", &SV1.NgaySinh.Thang);
 printf("Nam: "); scanf("%d", &SV1.NgaySinh.Nam);
 printf("Gioi tinh (0: Nu), (1: Nam):");
 scanf("%d", &SV1.Gioitinh);
 fflush(stdin); // Xoa bo nho dem
 printf("Dia chi: "); gets(SV1.DiaChi);
 InSV(SV1);
 SV2=SV1; /* Gan thong tin cua SV1 cho SV2*/
 InSV(SV2);
}
```

# Ví dụ - Tính tổng 2 số phức

---

❑ Nhập vào 2 số phức, tính tổng và in kết quả

```
typedef struct
{
 float Thuc;
 float Ao;
} SoPhuc;
void InSoPhuc(SoPhuc p)
{
 printf("%.2f + i%.2f\n", p.Thuc, p.Ao);
}
int main()
{
 SoPhuc p1, p2, p;
 printf("Nhap so phuc thu nhat:\n");
 printf("Phan thuc: "); scanf("%f", &p1.Thuc);
 printf("Phan ao: "); scanf("%f", &p1.Ao);
```

# Ví dụ - Tính tổng 2 số phức

---

```
printf("Nhap so phuc thu hai:\n");
printf("Phan thuc: ");scanf("%f",&p2.Thuc);
printf("Phan ao: ");scanf("%f",&p2.Ao);
printf("So phuc thu nhât: ");
InSoPhuc(p1);
printf("So phuc thu hai: ");
InSoPhuc(p2);
p.Thuc = p1.Thuc+p2.Thuc;
p.Ao = p1.Ao + p2.Ao;
printf("Tong 2 so phuc: ");
InSoPhuc(p);
getch();
}
```

# 3. Mảng cấu trúc

---

## □ Mảng cấu trúc

- Khai báo tương tự như mảng với kiểu dữ liệu cơ sở (char, int, float, ...)

## □ Ví dụ:

- Khai báo mảng để lưu danh sách sinh viên

```
struct SinhVien DanhSach[100];
```

- Mảng các struct cũng được đánh chỉ số từ 0
- Truy cập đến Họ tên của sinh viên thứ *i*

```
DanhSach[i].Hoten
```

## □ Bài tập

Nhập vào danh sách *n* sinh viên, in danh sách vừa nhập ra màn hình



## 4. Con trỏ cấu trúc

---

- ❑ C cho phép sử dụng con trỏ trỏ tới cấu trúc cũng như các con trỏ trỏ tới các kiểu dữ liệu khác

- ❑ Khai báo

`struct <Tên cấu trúc> * <Tên biến con trỏ>;`

Khi khai báo, con trỏ chưa trỏ tới địa chỉ cụ thể nào.

- ❑ Ví dụ:

Khai báo một con trỏ cấu trúc kiểu NgayThang ở trên

```
struct NgayThang *p;
struct NgayThang date;
p=&date;
```

Khi đó, **p** sẽ chứa địa chỉ của **date**

# Con trỏ cấu trúc (tt)

□ Truy cập đến các trường của cấu trúc đang được quản lý bởi con trỏ

■ Sử dụng toán tử mũi tên ( $\rightarrow$ )

■ Sử dụng toán tử lấy giá trị ( $*$ )

$p \rightarrow \text{Ngay}$  là tương đương  $(*p) . \text{Ngay}$

□ Ví dụ: Viết ra ngày-tháng-năm

```
struct NgayThang *p;
```

//Sử dụng toán tử mũi tên

```
printf ("%d-%d-%d", p->Ngay, p->Thang, p->Nam) ;
```

//Sử dụng toán tử  $*$

```
printf ("%d-%d-%d", (*p) .Ngay, (*p) .Thang, (*p) .Nam) ;
```

## 5. Chuyển tham số struct cho hàm

---

- ❑ Tham số của hàm có thể là
  - Từng trường của cấu trúc
  - Biến cấu trúc (tham số thực sự là giá trị cấu trúc)
  - Con trỏ cấu trúc (tham số thực sự là địa chỉ của biến cấu trúc)
  - Con trỏ cấu trúc hoặc mảng cấu trúc (tham số thực sự là tên mảng cấu trúc)
- ❑ Hàm có thể trả về
  - Giá trị cấu trúc
  - Con trỏ cấu trúc

# Ví dụ - Số phức

- ❑ Khai báo kiểu số phức, viết các hàm
  - `SoPhuc cong(SoPhuc u, SoPhuc v);` trả về tổng của các giá trị phức u,v.
  - `void InSP(SoPhuc u);` dùng để in số phức u

```
typedef struct
{
 float Thuc;
 float Ao;
} SoPhuc;

//Ham tinh tong 2 so phuc u,v
SoPhuc cong(SoPhuc u, SoPhuc v)
{
 SoPhuc tong;
 tong.Thuc=u.Thuc+v.Thuc;
 tong.Ao=u.Ao+v.Ao;
 return tong;
}
```

# Ví dụ - Số phức (tt)

```
void InSoPhuc (SoPhuc u)
{
 printf("%.2f + i%.2f\n", u.Thuc, u.Ao) ;
}

int main()
{
 SoPhuc p1, p2, p;
 printf("Nhap so phuc thu nhat:\n");
 printf("Phan thuc: ") ; scanf ("%f", &p1.Thuc) ;
 printf("Phan ao: ") ; scanf ("%f", &p1.Ao) ;
 printf("Nhap so phuc thu hai:\n");
 printf("Phan thuc: ") ; scanf ("%f", &p2.Thuc) ;
 printf("Phan ao: ") ; scanf ("%f", &p2.Ao) ;
 printf("Tong 2 so phuc: ") ;
 InSoPhuc (cong (p1, p2)) ;
 getch() ;
}
```

## 6. Union

---

### □ Khái niệm

- Được khai báo và sử dụng như cấu trúc
- Các thành phần của union có chung địa chỉ đầu (nằm chồng lên nhau trong bộ nhớ)

### □ Khai báo

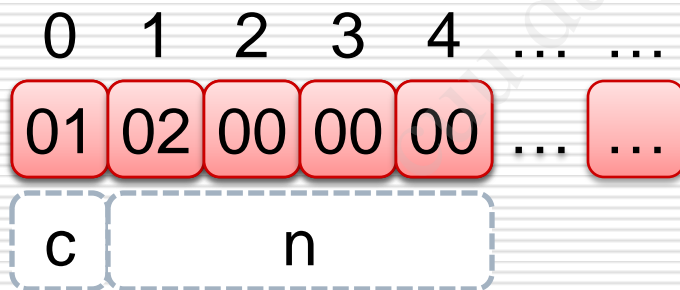
```
union <tên kiểu union>
{
 <kiểu dữ liệu> <tên thành phần 1>;
 ...
 <kiểu dữ liệu> <tên thành phần 2>;
};
```

# So sánh struct và union

## □ Ví dụ

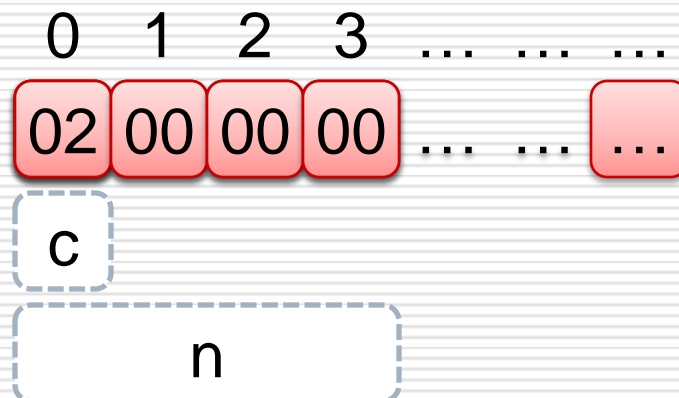
```
struct MYSTRUCT
{
 char c;
 int n;
} s;
```

```
s.c = 1; s.n = 2;
```



```
union MYUNION
{
 char c;
 int n;
} u;
```

```
u.c = 1; u.n = 2;
```



# Bài tập thực hành

---

## 1. Phân số

- Khai báo kiểu dữ liệu phân số (PHANSO)
- Nhập/Xuất phân số
- Rút gọn phân số
- Tính tổng, hiệu, tích, thương hai phân số
- Kiểm tra phân số tối giản
- Quy đồng hai phân số
- So sánh hai phân số



# Bài tập thực hành

---

## 2. Mảng phân số

- Nhập/Xuất n phân số
- Rút gọn mọi phân số
- Đếm số lượng phân số âm/dương trong mảng
- Tìm phân số dương đầu tiên trong mảng
- Tìm phân số nhỏ nhất/lớn nhất trong mảng
- Sắp xếp mảng tăng dần/giảm dần

# Bài tập thực hành

---

## 3. Điểm trong mặt phẳng Oxy

- Khai báo kiểu dữ liệu điểm (DIEM)
- Nhập/Xuất tọa độ 2 điểm A và B
- Tính khoảng cách giữa hai điểm A và B
- Tìm điểm đối xứng qua gốc tọa độ/trục Ox/Oy

## 4. Tam giác

- Khai báo kiểu dữ liệu tam giác (TAMGIAC)
- Nhập/Xuất tam giác
- Tính chu vi, diện tích tam giác

# Bài tập thực hành

---

## 5. Mảng điểm

- Nhập/Xuất n điểm
- Đếm số lượng điểm có hoành độ dương
- Tìm điểm có hoành độ lớn nhất/nhỏ nhất
- Tìm điểm gần gốc tọa độ nhất

## 6. Quản lý học sinh

- Khai báo kiểu dữ liệu học sinh (HOCSINH)
- Nhập n học sinh với các thuộc tính: Họ tên, năm sinh, điểm toán, lý, hoá, tổng điểm. (Tổng điểm = toán + lý + hoá, được tính sau khi nhập các điểm toán, lý, hoá)

# Bài tập thực hành

---

## 6. Quản lý học sinh

- Sắp xếp danh sách theo thứ tự giảm của tổng điểm, in kết quả ra màn hình
- Tìm kiếm theo họ và tên của học sinh, đưa kết quả ra màn hình.
- In ra màn hình các thí sinh có tổng điểm lớn hơn 15