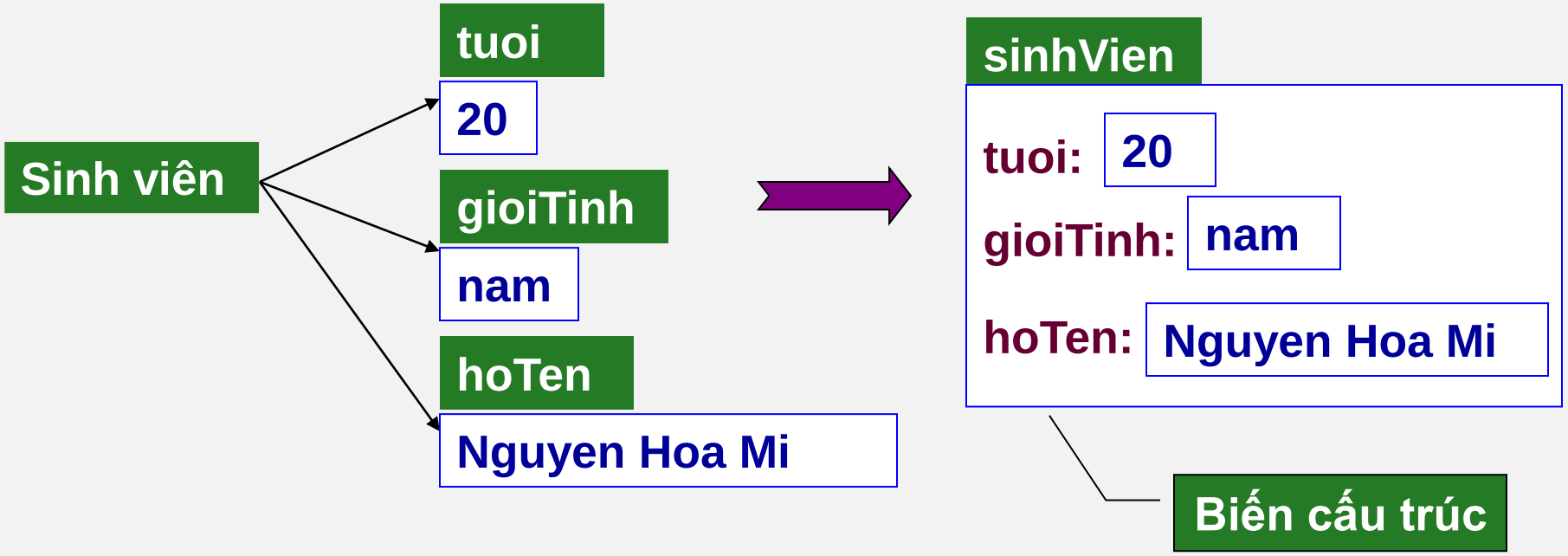
Chương 7

KỸ THUẬT SỬ DỤNG CẤU TRÚC

Nội dung

1. Khái niệm cấu trúc
2. Khai báo cấu trúc
3. Sử dụng biến cấu trúc
4. Hàm và cấu trúc
5. Mảng cấu trúc
6. Con trỏ cấu trúc

7.1. Khái niệm cấu trúc



- **Cấu trúc:** là tập hợp các thành phần dữ liệu của cùng một đối tượng.

7.2. Khai báo cấu trúc

❖ Khai báo kiểu cấu trúc

Mẫu 1

```
struct <tên_kiểu_cấu_trúc>
{
        <kdl_1> bien_tp_1;
        <kdl_2> bien_tp_2;
                … … …
        <kdl_n> bien_tp_n;
};
```

Khai báo cấu trúc (tt)

- **Khai báo kiểu cấu trúc**

Ví dụ

```
struct SinhVien
{
        int tuoi;
        char gioiTinh[4];
        char hoTen[30];
};
```

Khai báo cấu trúc (tt)

- **Khai báo kiểu cấu trúc**

Mẫu 2

```
typedef struct
{
      <kdl_1> bien_tp_1;
      <kdl_2> bien_tp_2;
            … … …
      <kdl_n> bien_tp_n;
} <Tên_Kiểu_Cấu_Trúc>;
```

Khai báo cấu trúc (tt)

- **Khai báo kiểu cấu trúc**

Ví dụ

```
typedef struct
{
      int tuoi;
      char gioiTinh[4];
      char hoTen[30];
} SinhVien;
```

Khai báo cấu trúc (tt)

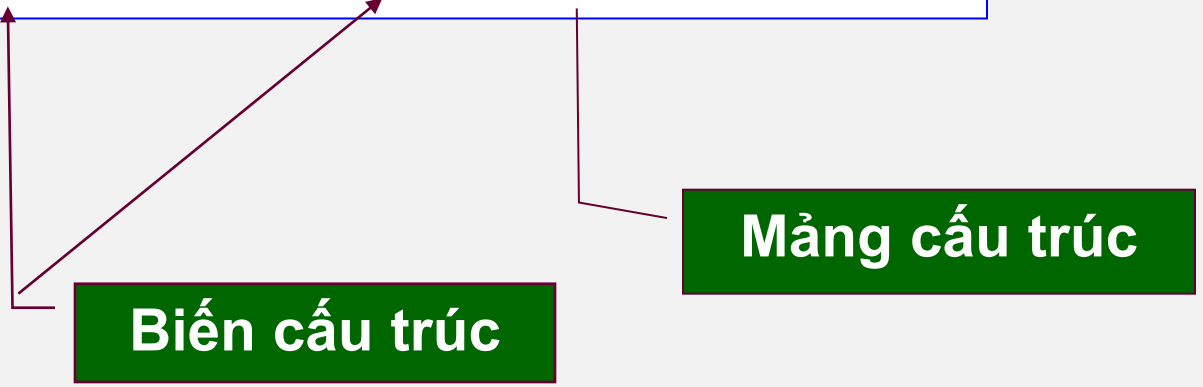
- **Khai báo biến cấu trúc**

Mẫu

```
<tên_kiểu_cấu_trúc> <dãy_tên_biến> ;
```

Ví dụ

```
SinhVien sinhVien, sv1, s[20];
```



7.3. Sử dụng biến cấu trúc

- Truy xuất vào các thành phần của cấu trúc
- Khởi tạo dữ liệu cho biến cấu trúc
- Hiển thị dữ liệu ra màn hình
- Ví dụ ứng dụng

7.3.1. Truy xuất vào các thành phần của cấu trúc

- Mỗi biến cấu trúc gồm các biến thành phần.
- Khi thao tác với biến cấu trúc cần truy xuất vào các biến thành phần.
- Để truy xuất vào các biến thành phần ta viết theo mẫu.

Mẫu

```
<tên_biến>.<tên_thành_phần>
```

Ví dụ

```
sinhVien.hoTen
```

7.3.2. Khởi tạo dữ liệu cho biến cấu trúc

- **Khởi tạo trong câu lệnh khai báo biến.**

Ví dụ

```
struct SinhVien
{
        int tuoi;
        char gioiTinh[4];
        char hoTen[30];
};
```

sinhVien

tuoi: 20

gioiTinh: nam

hoTen: Nguyen Hoa Mi

```
SinhVien sinhVien = {20, "nam", "Nguyen Hoa Mi"};
```

Khởi tạo dữ liệu cho biến cấu trúc (tt)

- Gán dữ liệu cho từng thành phần của cấu trúc.

```
SinhVien sinhVien;
sinhVien.tuoi = 20;
strcpy(sinhVien.gioiTinh, "nam");
strcpy(sinhVien.hoTen, "Nguyen Hoa Mi");
```

Khởi tạo dữ liệu cho biến cấu trúc (tt)

- **Gán từ biến cấu trúc cùng kiểu**


```
SinhVien sinhVien_01;
sinhVien_01 = sinhVien;
```

Khởi tạo dữ liệu cho biến cấu trúc (tt)

- **Nhập dữ liệu từ bàn phím**

```
SinhVien sinhVien;
```

```
cout<<"Nhap ho ten sv:";
fflush(stdin);
gets(sinhVien.hoTen);
cout<<"Nhap gioi tinh: ";
fflush(stdin);
gets(sinhVien.gioiTinh);
cout<<"Nhap tuoi: ";
cin>>sinhVien.tuoi;
```

7.3.3. Hiển thị dữ liệu ra màn hình

```
cout<<"Ho va ten:";
cout<<sinhVien.hoTen<<endl;
cout<<"Gioi tinh: ";
cout<<sinhVien.gioiTinh<<endl;
cout<<"Tuoi: "<<sv.tuoi;
```

7.3.4. Sử dụng biến cấu trúc – Ví dụ

- **Viết chương trình với các yêu cầu:**
  - Khai báo kiểu dữ liệu SinhVien (sinh viên) gồm các thông tin: họ và tên, tuổi, giới tính, điểm tổng kết.
  - Khai báo một biến cấu trúc và khởi tạo với bộ dữ liệu ("Nguyen Hoa Mai", 20, "Nu", 7.5).
  - Hiển thị dữ liệu khởi tạo ra màn hình.

Sử dụng biến cấu trúc – Ví dụ (tt)

- **Khai báo kiểu dữ liệu cấu trúc - SinhVien**

```
typedef struct
{
      char hoTen[30];
      int tuoi;
      char gioiTinh[4];
      double diemTk;
} SinhVien;
```

Sử dụng biến cấu trúc – Ví dụ (tt)

- Khai báo, khởi tạo biến cấu trúc và in dữ liệu ra màn hình.

```
int main() {
      SinhVien sv = {"Nguyen Hoa Mai",
                     20, "Nu", 7.5};
      cout<<"Thong tin ve sinh vien\n";
      cout<<"\tHo va ten sv:"<<sv.hoTen<<endl;
      cout<<"\tGioi tinh: "<<sv.gioiTinh<<endl;
      cout<<"\tTuoi: "<<sv.tuoi<<endl;
      cout<<"\tDiem tk: "<<sv.diemTk;
      return 0;
}
```

7.4. Hàm và cấu trúc

- Hàm trả về cấu trúc
- Tham số của hàm là cấu trúc

7.4.1. Hàm trả về cấu trúc

- **Xét hàm khởi tạo và trả về cấu trúc sinh viên.**

```
SinhVien taoSv(char *ht,int tuoi,
                    char *gt,double d)
{
      SinhVien  sv;
      strcpy(sv.hoTen,ht);
      strcpy(sv.gioiTinh,gt);
      sv.tuoi = tuoi;
      sv.diemTk = d;
      return sv;
}
```

7.4.2. Đối của hàm là cấu trúc

- Truyền tham trị.
- Xét hàm hiển thị thông tin sinh viên.

```
void hienThiSv(SinhVien sv){
      cout<<"Thong tin ve sinh vien\n";
      cout<<"\tHo va ten: ";
      cout<<sv.hoTen<<endl;
      cout<<"\tGioi tinh: ";
      cout<<sv.gioiTinh<<endl;
      cout<<"\tTuoi: "<<sv.tuoi<<endl;
      cout<<"\tDiem tong ket: "<<sv.diemTk;
}
```

Đối của hàm là cấu trúc (tt)

- Truyền tham chiếu.
- Xét hàm nhập thông tin sinh viên.

```
void nhapSv(SinhVien &sv)
 {
      cout<<"Nhap thong tin ve sinh vien\n";
      cout<<"\tHo va ten sv:";
      fflush(stdin);  gets(sv.hoTen);
      cout<<"\tGioi tinh: ";
      fflush(stdin);  gets(sv.gioiTinh);
      cout<<"\tTuoi: "; cin>>sv.tuoi;
      cout<<"\tDiem tong ket: ";
      cin>>sv.diemTk;
}
```

7.5. Mảng cấu trúc

- Khai báo và sử dụng mảng cấu trúc
- Các thao tác xử lý danh sách
- Bài tập ứng dụng

7.5.1. Khai báo mảng cấu trúc

- Được sử dụng để lưu trữ danh sách.
- Mỗi phần tử mảng lưu trữ một cấu trúc.
- Khai báo mảng cố định.

```
SinhVien s[4] = { {20, "nam", "Nguyen Hoa Mi"},
                  {22, "nu", "Tran Hoa Mai"},
                  {21, "nu", "Bui Hoa Hong"},
                  {20, "nam", "Nguyen Tu Tai"}
                };
```

- Hoặc sử dụng con trỏ và cấp phát mảng động.

```
SinhVien *s;
s = new SinhVien[4];
```

Khai báo mảng cấu trúc (tt)

- **Mảng s[4] chứa dữ liệu 4 sinh viên.**

7.5.2. Các thao tác xử lý danh sách

- **Khởi tạo dữ liệu**
- **Hiển thị dữ liệu**
- **Tính toán và thống kê**
- **Tìm kiếm**
- **Thêm, sửa, xóa dữ liệu**
- **Sắp xếp dữ liệu**
- ***v.v…***

7.6. Con trỏ cấu trúc

- Là con trỏ trỏ đến biến cấu trúc.
- Khai báo kiểu con trỏ cấu trúc.

Mẫu

```
typedef <tên_kiểu_CT> *<tên_kiểu_con_trỏ>;
```

Ví dụ

```
typedef SinhVien *StudentPointer;
```

Con trỏ cấu trúc (tt)

- **Khai báo biến con trỏ cấu trúc**

Mẫu 1

```
<tên_kiểu_cấu_trúc> *<tên_con_trỏ>;
```

Ví dụ 1

```
SinhVien *sp;
```

Mẫu 2

```
<tên_kiểu_con_trỏ> <tên_con_trỏ>;
```

Ví dụ 2

```
StudentPointer sp;
```

Con trỏ cấu trúc (tt)

- **Cấp phát bộ nhớ cho con trỏ cấu trúc**

Mẫu

```
<tên_con_trỏ> = new <tên_kiểu_CT>;
```

Ví dụ

```
sp = new SinhVien;
```

Con trỏ cấu trúc (tt)

- **Truy xuất các thành phần của cấu trúc thông qua con trỏ.**

Mẫu

```
<tên_con_trỏ> -> <tên_thành_phần>
```

Ví dụ

```
sp -> hoTen
```

Con trỏ cấu trúc (tt) - Ứng dụng

- **Cài đặt chương trình sử dụng con trỏ để:**
  - **Nhập thông tin một sinh viên.**
  - **Hiển thị thông tin sinh viên ra màn hình.**

7.7. Bài tập ứng dụng

1. Viết chương trình thực hiện

- Khai báo kiểu cấu trúc CanBo (cán bộ) gồm các thông tin: Mã cán bộ, họ và tên, năm sinh, nơi sinh, hệ số lương, lương (hệ số lương * 1150000).
- Cài đặt các hàm chức năng:
  - Hàm khoiTaoCb(…) để khởi tạo thông tin cho 1 cán bộ.
  - Hàm khoiTaoDs(…) để khởi tạo danh sách 5 cán bộ.
  - Hàm hienThiCb(…) để hiển thị thông tin 1 cán bộ.
  - Hàm hienThiDs(…) để hiển thị danh sách ra màn hình.
  - Hàm tongLuong(…) để tính và trả về tổng lương của tất cả các cán bộ.
  - Hàm xoaCb(…) để xóa cán bộ thứ 3 trong danh sách.
- Hàm main() gọi các hàm trên để kiểm nghiệm các kết quả.

Bài tập ứng dụng (tt)

2. Viết chương trình thực hiện

- Khai báo kiểu cấu trúc NhanVien gồm các thông tin: Mã nhân viên, họ đệm, tên, ngày sinh (ngày/tháng/năm), giới tính, nơi sinh, điện thoại, lương tháng.
- Hàm nhapNv(…) để nhập dữ liệu cho 1 nhân viên.
- Hàm nhapDs(…) sử dụng hàm nhapNv(…) để nhập dữ liệu cho n nhân viên (1 <= n <= 10).
- Hàm hienThiNv(…) để hiển thị thông tin 1 nhân viên
- Hàm hienThiDs(…) để hiển thị danh sách ra màn hình
- Hàm viewMinAge(…) để hiển thị ra màn hình họ và tên, tuổi, giới tính của các nhân viên có tuổi thấp nhất.
- Hàm chenNv(…) để chèn cán bộ mới vào vị trí thứ 3 trong danh sách.
- Hàm sapXep(…) để sắp xếp danh sách theo tên tăng dần.
- Hàm main() gọi các hàm trên để kiểm nghiệm các kết quả.

Thank you…!

Design by Minh An