TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN
MÔN: **NHẬP MÔN LẬP TRÌNH**

HƯỚNG DẪN THỰC HÀNH

TUẦN 08

DỮ LIỆU CẤU TRÚC

TP.HCM, ngày 09 tháng 09 năm 2017

MỤC LỤC

1 Khái niệm

Trong phần này ta sẽ tìm hiểu các vấn đề liên quan tới dữ liệu có cấu trúc. Trong thực tế ngoài các kiểu dữ liệu cơ bản như số nguyên, số thực và kí tự thì còn rất nhiều kiểu dữ liệu phức tạp khác là sự pha trộn của các kiểu dữ liệu cơ bản. Ví dụ như để biểu diễn một học sinh ta cần các thông tin như họ tên, tuổi, lớp, điểm trung bình… Rõ ràng ta thấy cần có một kiểu đại diện tập hợp các thông tin trên.

2 Khai báo kiểu dữ liệu có cấu trúc

Ta đã có các kiểu cơ bản như int, float, double, char... Giờ giả sử ta cần tạo ra một kiểu dữ liệu mới ví dụ như HOCSINH. Bên dưới là cú pháp tạo

```
struct hocsinh{
  char HoTen[31];
  int Tuoi;
  int Lop;
  float DTB;
};
typedef struct hocsinh HOCSINH;
```

Câu lệnh ‘typedef’ cuối cùng hàm ý sẽ chuyển kiểu ‘struct hocsinh’ thành HOCSINH. Như vậy sau câu lệnh ‘typedef’ này thì HOCSINH tương đương với ‘struct hocsinh’. Sau khi khai báo tạo xong kiểu dữ liệu mới, ta giờ có thể sử dụng kiểu HOCSINH này trong tất cả các hàm lập trình như những kiểu cơ sở khác bằng cách theo cú pháp cũ:

HOCSINH hs;

Câu khai báo trên ta muốn có một biến tên là ‘hs’ kiểu HOCSINH. Giống như các kiểu khác, sau câu khai báo trên thì hs vẫn chưa chứa bất kì giá trị nào nên vẫn chưa sử dụng được. Đối với các kiểu cơ sở ta dùng ‘scanf’ và ‘printf’ để nhập liệu và xuất liệu, nhưng đối với kiểu mà ta tự tạo thì rõ ràng không thể dùng ‘scanf’ và ‘printf’. Ta cần tạo hai hàm nhập và xuất cho riêng kiểu HOCSINH của ta.

Dòng	
1	#include <stdio.h>
2	
3	struct hocsinh{
4	char HoTen[31];
5	int Tuoi;
6	int Lop;
7	float DTB;

```
8   };
9   typedef struct hocsinh HOCSINH;
10
11  void NhapHocSinh(HOCSINH&);
12  void XuatHocSinh(HOCSINH);
13
14  void NhapHocSinh(HOCSINH &hs){
15    printf("Nhap ho ten hoc sinh: ");
16    gets(hs.HoTen);
17    printf("Nhap tuoi hoc sinh: ");
18    scanf("%d", &hs.Tuoi);
19    printf("Nhap lop hoc sinh: ");
20    scanf("%d", &hs.Lop);
21    printf("Nhap diem trung binh hoc sinh: ");
22    scanf("%f", &hs.DTB);
23  }
24
25  void XuatHocSinh(HOCSINH hs){
26    printf("Ho ten hoc sinh: %s\n", hs.HoTen);
27    printf("Tuoi hoc sinh: %d\n", hs.Tuoi);
28    printf("Lop hoc sinh: %d\n", hs.Lop);
29    printf("Diem trung binh hoc sinh: %f\n", hs.DTB);
30  }
31
32  void main(){
33    HOCSINH hs;
34    NhapHocSinh(hs);
35    XuatHocSinh(hs);
36  }
```

3 Kiểu dữ liệu cấu trúc lồng

Trong phần trên ta đã làm quen với việc tạo ra một kiểu dữ liệu mới đó là HOCSINH, sau đó viết hàm nhập và xuất cho kiểu mới này. Phần này ta sẽ được giới thiệu qua khả năng lồng của kiểu có cấu trúc. Xét ví dụ về kiểu dữ liệu 'Điểm' trong mặt phẳng, ta biết một điểm trong mặt phẳng có hoành độ và tung độ. Vậy nếu đơn giản chỉ xét trục số nguyên ta có thể xây dựng kiểu 'DIEM' như sau:

```
struct diem{
  int X;
  int Y;
```

```
};
typedef struct diem DIEM;
```

Tiếp tục ta xem xét về đường tròn, ta biết một đường tròn trong thực tế cần có một tâm và bán kính để biểu diễn. Như vậy ta thấy rằng để biểu diễn được đường tròn cần tạo trước kiểu 'Điểm' để đặc trưng cho tâm đường tròn. Ta có thể xây dựng kiểu 'DUONGTRON' như sau:

```
struct diem{
  int X;
  int Y;
};
typedef struct diem DIEM;
struct duongtron{
  DIEM I;
  float R;
};
typedef struct duongtron DUONGTRON;
```

Sau khi tạo xong hai kiểu 'DIEM' và 'DUONGTRON' ta cũng cần xây dựng 4 hàm gồm 2 hàm nhập và xuất cho DIEM và 2 hàm nhập và xuất cho DUONGTRON. Lưu ý: hàm nhập và xuất của DUONGTRON sẽ tái sử dụng lại 2 hàm nhập xuất của DIEM.

Dòng	
1	#include <stdio.h>
2	
3	void NhapDiem(DIEM&);
4	void XuatDiem(DIEM);
5	void NhapDuongTron(DUONGTRON&);
6	void XuatDuongTron(DUONGTRON);
7	
8	void NhapDiem(DIEM &d){
9	printf("Nhap hoanh do: ");
10	scanf("%d", &d.X);
11	printf("Nhap tung do: ");
12	scanf("%d", &d.Y);
13	}
14	
15	void NhapDuongTron(DUONGTRON &dt){
16	printf("Nhap tam duong tron: ");
17	NhapDiem(dt.I);
18	printf("Nhap ban kinh: ");
19	scanf("%f", &dt.R);
20	}

```
21  void XuatDiem(DIEM d){
22    printf("Toa do: (%d, %d)", d.X, d.Y);
23  }
24
25  void XuatDuongTron(DUONGTRON dt){
26    printf("Tam duong tron: ");
27    XuatDiem(dt.I);
28    printf("Ban kinh: %f", dt.R);
29  }
30
31  void main(){
32    DUONGTRON dt;
33    NhapDuongTron(dt);
34    XuatDuongTron(dt);
35  }
```

4 Một số kĩ thuật khác

Như vậy ta đã được giới thiệu cách thức tạo kiểu dữ liệu mới và cũng khai báo và định nghĩa hai thao tác cơ bản là nhập và xuất. Tuy nhiên ngoài nhập xuất, ta còn có thể tạo thêm nhiều hàm (chức năng) cho kiểu dữ liệu mới của ta. Ví dụ với kiểu đường tròn ta có thể tạo ra thêm hai hàm là tính diện tích và tính chu vi đường tròn. Bên dưới là bổ sung hai hàm con này vào chương trình, giúp kiểu dữ liệu đường tròn của ta phong phú hơn.

```
Dòng
1   #include <stdio.h>
2
3   //Tuong tu nhu vi du truoc
4
5   float TinhChuViDuongTron(DUONGTRON);
6   float TinhDienTichDuongTron(DUONGTRON);
7
8   float TinhChuViDuongTron(DUONGTRON dt){
9     return 2*3.14*dt.R;
10  }
11
12  float TinhDienTichDuongTron(DUONGTRON dt){
13    return 3.14*dt.R*dt.R;
14  }
15
16  void main(){
```

17	DUONGTRON dt;
18	NhapDuongTron(dt);
19	XuatDuongTron(dt);
20	float cv = TinhChuViDuongTron(dt);
21	float dt = TinhDienTichDuongTron(dt);
22	}

5 Bài tập

Sinh viên thực hiện các bài tập sau đây:

5.1 Viết chương trình tạo kiểu dữ liệu phân số (PHANSO) và xây dựng các chức năng:

5.1.1 Viết hàm cộng hai phân số

5.1.2 Viết hàm trừ hai phân số

5.1.3 Viết hàm nhân hai phân số

5.1.4 Viết hàm chia hai phân số

5.1.5 Viết hàm rút gọn phân số

5.2 Viết chương trình tạo kiểu dữ liệu Tam giác và thực hiện các chức năng sau:

5.2.1 Viết hàm tính diện tích tam giác

5.2.2 Viết hàm tính chu vi tam giác

5.3 Viết chương trình tạo kiểu dữ liệu ngày (NGAY bao gồm: ngày tháng và năm) và thực hiện các chức năng sau (Lấy ngày nhập vào làm gốc).

5.3.1 Viết hàm tính khoảng cách hai ngày với đơn vị là theo ngày, theo tháng hoặc theo năm

5.3.2 Viết hàm kiểm tra năm nhuận

5.3.3 Viết hàm tính ngày hôm qua

5.3.4 Viết hàm tính ngày hôm trước

5.3.5 Viết hàm tính ngày trước k ngày

5.3.6 Viết hàm tính ngày sau k ngày