CHƯƠNG 7 KIỂU DỮ LIỆU CÓ CẤU TRÚC

Cung cấp cơ chế cho phép khai báo các kiểu dữ liệu mới để giải quyết theo yêu cầu của bài toán dựa vào những kiểu dữ liệu cơ bản được cài đặt sẵn trong ngôn ngữ lập trình.

I. TÓM TẮT LÝ THUYẾT

I.1. Khái niệm

Cấu trúc (struct) thực chất là một kiểu dữ liệu do người dùng định nghĩa bằng cách gom nhóm các kiểu dữ liệu cơ bản có sẵn trong C thành một kiểu dữ liệu phức hợp nhiều thành phần.

I .2. Định nghĩa kiểu dữ liệu

Cú pháp

```
struct < tên cấu trúc >
{
        Các kiểu dữ liệu thành phần ;
};
```

Ngoài ra ta có thể dùng từ khoá **typedef** để định nghĩa một tên mới cho kiểu dữ liệu đã có.

Cú pháp

```
typedef struct < tên cấu trúc > < tên mới >;
```

Ví dụ1: Kiểu dữ liệu DATE gồm các thành phần:

- *Thứ (thu): chuỗi có tối đa 4 ký tự.*
- *Ngày (ngay): số nguyên 1 byte.*
- *Tháng (thang): số nguyên 1 byte.*
- *Năm (nam): số nguyên 2 bytes.*

Ta định nghĩa DATE như sau:

```
struct DATE
{
  char thu[5];
  unsigned char ngay;
  unsigned char  thang;
  int nam;
};
typedef struct DATE d;
```

Kiểu dữ liệu có cấu trúc có thể lồng vào nhau.

Ví dụ 2: Định nghĩa kiểu dữ liệu của học sinh HOCSINH gồm:

- *Mã số học sinh (MSHS): chuỗi có tối đa 5 ký tự.*
- *Họ tên (hoten): chuỗi có tối đa 30 ký tự.*
- *Ngày tháng năm sinh (ngaysinh): kiểu DATE.*
- *Địa chỉ (diachi): chuỗi có tối đa 50 ký tự.*
- *Giới tính (phai): chuỗi có tối đa 3 ký tự.*
- *Điểm trung bình (diemtb): số thực.*

Ta định nghĩa kiểu HOCSINH như sau:

```
struct DATE
{
  char thu[5];
  unsigned char ngay;
  unsigned char  thang;
  int nam;
};

typedef struct HOCSINH
{
  char MSHS[6];
  char hoten[31];
  struct DATE ngaysinh;
  char diachi[51];
  unsigned char phai[4];
  float diemtb;
};
```

✍ *Khi định nghĩa kiểu dữ liệu struct lồng nhau, ta cần lưu ý: Kiểu dữ liệu được sử dụng phải khai báo phía trên.*

I.3. Khai báo

Khi ta định nghĩa kiểu dữ liệu tức là ta có một kiểu dữ liệu mới, muốn sử dụng ta phải khai báo biến. Cú pháp khai báo kiểu dữ liệu cũng giống như cách khai báo của các kiểu dữ liệu chuẩn.

struct < tên cấu trúc > < tên biến > ;

Ví dụ :

```
struct DATE x ;   // Khai bao bien x co kieu du lieu DATE
```

Tuy nhiên nếu ta định nghĩa struct có dùng từ khoá **typedef** thì ta có thể khai báo trực tiếp mà không cần từ khoá "**struc**t".

Ví dụ :

DATE x ; // Khai bao bien x co kieu DATE

***Biến con trỏ kiểu cấu trúc:** Ngoài cách khai báo như trên ta có thể khai báo theo kiểu con trỏ như sau

struct < tên cấu trúc > *< tên biến > ;

Để sử dụng ta cũng phải cấp phát vùng nhớ giống như kiểu dữ liệu chuẩn.

Ví dụ :

DATE *y; // Khai bao con tro y kieu cau truc DATE

y = (DATE *) malloc (sizeof (DATE)) ;

I.4. Truy xuất

Để truy xuất một thành phần dữ liệu nào đó bên trong cấu trúc ta có 2 trường hợp truy xuất như sau :

- Biến x là một biến cấu trúc thông thường, ta dùng toán tử dấu chấm "**.**"

  Cú pháp :

< Tên cấu trúc >.< Biến thành phần >;

  Ví dụ :

  DATE x ; // khai bao bien x kieu DATE

  x.ngay = 5 ; // gan ngay bang 5

- Biến x là một biến con trỏ, ta dùng toán tử mũi tên "->" (Gồm dấu trừ '–' và dấu lớn hơn '>').

  Cú pháp :

< Tên cấu trúc > -> < Biến thành phần >;

  Ví dụ :

  DATE *x ; // khai bao bien x kieu con tro DATE

  x -> ngay = 5 ; // gan ngay bang 5

✍ Đối với kiểu dữ liệu có struct lồng nhau phải truy cập đến thành phần cuối cùng có kiểu dữ liệu cơ bản.

Ví dụ: Giả sử, có kiểu HOCSINH như trên

HOCSINH hs; // khai bao bien hs kieu HOCSINH

Muốn in học sinh A sinh vào tháng mấy ta phải truy cập như sau:

```
printf("Thang sinh cua hoc sinh A la: %d",(hs.ngaysinh).thang);
```

I.5. Ví dụ minh hoạ

Viết chương trình nhập vào toạ độ hai điểm trong mặt phẳng và tính tổng hai toạ độ này.

```
#include <conio.h>
#include <stdio.h>

typedef struct DIEM   //khai bao mot kieu du lieu DIEM gom toa do x va y
{
        int x;
        int y;
};

void Nhap (DIEM &d)
{
        printf ("\nNhap vao tao do diem\n");
        printf ("Tung do : ");
        scanf ("%d", & d. x);
        printf ("Hoanh do : ");
        scanf ("%d", & d.y);
}

void Xuat (DIEM d)
{
        printf ("\nToa do diem  : (%d , %d)",d.x,d.y);
}

DIEM Tong (DIEM d1,DIEM d2)
{
        DIEM temp;
        temp.x = d1.x + d2.x ;
        temp.y = d1.y + d2.y ;
        return Temp;
}

void main ()
{
        DIEM A , B, AB;              //khai bao 3 diem A, B, AB;
        clrscr ();
        Nhap ( A );
        Xuat ( A );
        Nhap ( B );
        Xuat ( B );
        printf ("\n Tong cua hai diem vua nhap la : ");
        AB = Tong  ( A, B);
        Xuat ( AB );
```

```
        getch ();
}
```

I.6. Mảng cấu trúc

- Cách khai báo tương tự như mảng một chiều hay ma trận (Kiểu dữ liệu bây giờ là kiểu dữ liệu có cấu trúc).
- Cách truy cập phần tử trong mảng cũng như truy cập trên mảng một chiều hay ma trận. Nhưng do từng phần tử có kiểu cấu trúc nên phải chỉ định rõ cần lấy thành phần nào, tức là phải truy cập đến thành phần cuối cùng có kiểu là dữ liệu cơ bản (xem lại bảng các kiểu dữ liệu cơ bản) .

I.7. Nguyên tắc viết chương trình có mảng cấu trúc

Do kiểu dữ liệu có cấu trúc thường chứa rất nhiều thành phần nên khi viết chương trình loại này ta cần lưu ý:

- Xây dựng hàm xử lý cho một kiểu cấu trúc.
- Muốn xử lý cho mảng cấu trúc, ta gọi lại hàm xử lý cho một kiểu cấu trúc đã được xây dựng bằng cách dùng vòng lặp.

***Ví dụ 1**: Cho một lớp học gồm n học sinh (n≤50). Thông tin của một học sinh được mô tả ở ví dụ 2, mục I.2. Hãy viết chương trình nhập và xuất danh sách học sinh sau đó đếm xem có bao nhiêu học sinh được lên lớp (Điều kiện được lên lớp là điểm trung bình ≥ 5.0).*

Cách làm:

- *Trước hết ta phải xây dựng hàm nhập và xuất cho 1 học sinh.*
- *Xây dựng hàm nhập và xuất ngày tháng năm (Kiểu dữ liệu DATE).*
- *Sau đó mới xây dựng hàm nhập và xuất cho danh sách học sinh.*

```
#define MAX 50
struct DATE
{
        char thu[5];
        unsigned char ngay;
        unsigned char  thang;
        int nam;
};
typedef struct HOCSINH
{
        char MSHS[6];
        char hoten[31];
        struct DATE ngaysinh;
```

```
        char diachi[51];
        unsigned char phai[4];
        float diemtb;
};

void NhapNamSinh(DATE &d);
void XuatNamSinh(DATE d);
void Nhap1HS (HOCSINH &hs);
void Xuat1HS (HOCSINH hs);
void NhapDSHS(HOCSINH lh[], int &n);
void XuatDSHS(HOCSINH lh[], int n);
int DemHSLenLop(HOCSINH lh[], int n);

void main()
{
    HOCSINH lh[MAX]; //Khai báo mảng lh gồm có tối đa 50 học sinh
    int n, sohsdau;
    NhapDSHS(lh, n);
    XuatDSHS(lh, n);
    sohsdau = DemHSLenLop(lh, n);
    printf("\nSo luong hoc sinh duoc len lop la: %d", sohsdau);
    getch();
}

void NhapNamSinh(DATE &d)
{
    printf("\nNhap vao ngay: ");
    scanf("%u", &d.ngay);
    printf("\nNhap vao thang: ");
    scanf("%u", &d.thang);
    printf("\nNhap vao nam: ");
    scanf("%d", &d.nam);
}

void XuatNamSinh(DATE d)
{
    printf("%02u / %02u / %4d", d.ngay, d.thang, d.nam);
}

void Nhap1HS(HOCSINH &hs)
{
    float d;

    lushall(); //Xoa vung dem
    printf("\nNhap ma so hoc sinh: ");
    gets(hs.MSHS);
    printf("\nNhap ho ten hoc sinh: ");
    gets(hs.hoten);
    printf("\nNhap ngay thang nam sinh: ");
```

```
    flushall(); //Xoa vung dem
    NhapNamSinh(hs.ngaysinh);
    printf("\nNhap vao dia chi: ");
    flushall(); //Xoa vung dem
    gets(hs.diachi);
    printf("\nPhai: ");
    gets(hs.phai);
    printf("\nNhap vao diem trung binh: ");
    flushall(); //Xoá vùng đệm
    scanf("%f", &d);//Nhập vào biến tạm d sau đó gán vào hs.diemtb
    hs.diemtb=d;
}

void NhapDSHS(HOCSINH lh[], int &n)
{
    printf("\nNhap vao so luong hoc sinh: ");
    scanf("%d", &n);
    for(int i=0; i<n; i++)
    {
        printf("\nNhap vao thong tin cua hoc sinh thu %d:\n", i+1);
        Nhap1HS(lh[i]);     //Goi ham nhap thong tin 1 hoc sinh
    }
}

void Xuat1HS(HOCSINH hs)
{
    printf("\nMa so hoc sinh: %s", hs.MSHS);
    printf("\nHo ten hoc sinh: %s", hs.hoten);
    printf("\nNgay thang nam sinh: ");
    XuatNamSinh(hs.ngaysinh);
    printf("\nDia chi: %s", hs.diachi);
    printf("\nPhai: %s", hs.phai);
    printf("\nDiem trung binh: %2.2f", hs.diemtb);
}

void XuatDSHS(HOCSINH lh[], int n)
{
    for(int i=0; i<n; i++)
    {
        printf("\n\nThong tin hoc sinh thu %d:", i+1);
        Xuat1HS(lh[i]);     //Goi ham xuat thong tin 1 hoc sinh
    }
}

int DemHSLenLop(HOCSINH lh[], int n)
{
    int d=0;
    for(int i=0; i<n; i++)
        if(lh[i].diemtb>=5.0)
```

```
            d++;
    return d;
}
```

Kết quả ví dụ khi chạy chương trình:

Nhap vao thong tin cua hoc sinh thu 1:
Nhap ma so hoc sinh: 02313
Nhap ho ten hoc sinh: Nguyen Van A
Nhap ngay thang nam sinh:
Nhap vao ngay: 12
Nhap vao thang: 03
Nhap vao nam: 1980
Nhap vao dia chi: 60 Phan Dang Luu Q.Phu Nhuan
Phai: Nam
Nhap vao diem trung binh: 6.5

Nhap vao thong tin cua hoc sinh thu 2:
Nhap ma so hoc sinh: 03852
Nhap ho ten hoc sinh: Ly Thi B
Nhap ngay thang nam sinh:
Nhap vao ngay: 05
Nhap vao thang: 12
Nhap vao nam: 1981
Nhap vao dia chi: 24 Ly Tu Trong Q.1
Phai: Nu
Nhap vao diem trung binh: 3.5

Thong tin hoc sinh thu 1:
Ma so hoc sinh: 02313
Ho ten hoc sinh: Nguyen Van A
Ngay thang nam sinh: 12 / 03 / 1980
Dia chi: 60 Phan Dang Luu Q.Phu Nhuan
Phai: Nam
Diem trung binh: 6.50

Thong tin hoc sinh thu 2:
Ma so hoc sinh: 03852
Ho ten hoc sinh: Ly Thi B
Ngay thang nam sinh: 05 / 12 / 1981
Dia chi: 24 Ly Tu Trong Q.1
Phai: Nu
Diem trung binh: 3.50
So luong hoc sinh duoc len lop la: 1

Ví dụ 2: *Cho một mảng các phân số (**PHANSO**) gồm n phần tử (n≤50). Hãy viết chương trình nhập và xuất danh sách các phân số sau đó tìm phân số có giá trị lớn nhất, tổng và tích các phân số và nghịch đảo giá trị các phân số trong mảng.*

Cách làm:

- *Trước hết ta phải xây dựng hàm nhập và xuất cho 1 phân số.*
- *Xây dựng hàm tính tổng, hiệu, tích, thương, rút gọn, so sánh và nghịch đảo cho 2 phân số.*
- *Sau đó mới xây dựng hàm nhập, xuất, tính tổng, tích cho mảng các phân số.*

```
#define MAX 100
typedef struct PHANSO
        {
           int tu, mau;
        };

void NhapPS(PHANSO &ps);
void XuatPS(PHANSO ps);
void NhapMangPS(PHANSO dsps[], int &n);
void XuatMangPS(PHANSO dsps[], int n);
PHANSO TimMax(PHANSO dsps[], int n);
int KiemTra(PHANSO ps);
//Tra ve 1: Neu hop le
int USCLN(int a, int b);
PHANSO RutGon(PHANSO ps);
PHANSO NghichDao(PHANSO ps);
PHANSO Nhan(PHANSO ps1, PHANSO ps2);
PHANSO Chia(PHANSO ps1, PHANSO ps2);
PHANSO Tru(PHANSO ps1, PHANSO ps2);
PHANSO Cong(PHANSO ps1, PHANSO ps2);
int SoSanh(PHANSO ps1, PHANSO ps2);
//Tra ve 0: ps1=ps2
//Tra ve 1: ps1>ps2
//Tra ve -1: ps1<ps2
PHANSO TongCacPS(PHANSO dsps[], int n);
PHANSO TichCacPS(PHANSO dsps[], int n);
void NghichDaoCacPS(PHANSO dsps[], int n);

void main()
{
   int n;
   PHANSO a[MAX], max, s, p;
   clrscr();

   NhapMangPS(a, n);
   printf("\nMang cac phan so vua nhap: ");
   XuatMangPS(a, n);
   max=TimMax(a, n);
   printf("\nPhan so co gia tri lon nhat: ");
   XuatPS(max);

   s=TongCacPS(a, n);
```

```
   printf("\nTong gia tri cac phan so co trong mang: ");
   XuatPS(s);

   p=TichCacPS(a, n);
   printf("\nTich gia tri cac phan so co trong mang: ");
   XuatPS(p);

   NghichDaoCacPS(a, n);
   printf("\nMang phan so sau khi nghich dao cac phan tu: ");
   XuatMangPS(a, n);

   getch();
}

void NhapPS(PHANSO &ps)
{
   do{
      printf("\nNhap tu so: ");
      scanf("%d", &ps.tu);
      printf("\nNhap mau so: ");
      scanf("%d", &ps.mau);
      if(!KiemTra(ps))
         printf("\nMau so khong duoc bang 0, nhap lai phan so\n");
      else
         break;
   } while(1);
   ps=RutGon(ps);
}

void XuatPS(PHANSO ps)
{
   printf("%d", ps.tu);
   if(ps.tu&&ps.mau!=1)
      printf("/%d", ps.mau);
}

void NhapMangPS(PHANSO dsps[], int &n)
{
   printf("\nNhap so luong phan so: ");
   scanf("%d", &n);
   for(int i=0; i<n; i++)
   {
      printf("\nNhap vao phan so thu %d: ", i+1);
      NhapPS(dsps[i]);
   }
}

void XuatMangPS(PHANSO dsps[], int n)
{
```

```
   for(int i=0; i<n; i++)
   {
      XuatPS(dsps[i]);
      printf("\t");
   }
}

int KiemTra(PHANSO ps)
{
   if(ps.mau==0)
      return 0;
   return 1;
}

int USCLN(int a, int b)
{
   a=abs(a);
   b=abs(b);
   while(a!=b)
   {
      if(a>b)
         a=a-b;
      else
         b=b-a;
   }
   return a;
}

PHANSO RutGon(PHANSO ps)
{
   int us;
   if(ps.tu==0)
      return ps;

   us=USCLN(ps.tu, ps.mau);
   ps.tu=ps.tu/us;
   ps.mau=ps.mau/us;
   return ps;
}

PHANSO NghichDao(PHANSO ps)
{
   PHANSO kq;
   kq.tu=ps.mau;
   kq.mau=ps.tu;
   return kq;
}

PHANSO Nhan(PHANSO ps1, PHANSO ps2)
```

```
{
   PHANSO kq;
   kq.tu=ps1.tu*ps2.tu;
   kq.mau=ps1.mau*ps2.mau;
   kq=RutGon(kq);
   return kq;
}

PHANSO Chia(PHANSO ps1, PHANSO ps2)
{
   PHANSO kq;
   kq=Nhan(ps1, NghichDao(ps2));
   return kq;
}

PHANSO Tru(PHANSO ps1, PHANSO ps2)
{
   PHANSO kq;
   kq.tu=ps1.tu*ps2.mau-ps1.mau*ps2.tu;
   kq.mau=ps1.mau*ps2.mau;
   kq=RutGon(kq);
   return kq;
}

PHANSO Cong(PHANSO ps1, PHANSO ps2)
{
   PHANSO kq;
   kq.tu=ps1.tu*ps2.mau+ps1.mau*ps2.tu;
   kq.mau=ps1.mau*ps2.mau;
   kq=RutGon(kq);
   return kq;
}

int SoSanh(PHANSO ps1, PHANSO ps2)
{
   ps1=RutGon(ps1);
   ps2=RutGon(ps2);
   if(ps1.tu==ps2.tu&&ps1.mau==ps2.mau)
      return 0;
   if(ps1.tu*ps2.mau>ps2.tu*ps1.mau)
      return 1;
   return -1;
}

PHANSO TimMax(PHANSO dsps[], int n)
{
   PHANSO max;
   max=dsps[0];
   for(int i=1; i<n; i++)
```

```
        if(SoSanh(dsps[i], max)==1)
            max=dsps[i];
    return max;
}

PHANSO TongCacPS(PHANSO dsps[], int n)
{
    PHANSO s=dsps[0];
    for(int i=1; i<n; i++)
    {
        s=Cong(s, dsps[i]);
    }
    return s;
}

PHANSO TichCacPS(PHANSO dsps[], int n)
{
    PHANSO p=dsps[0];
    for(int i=1; i<n; i++)
    {
        p=Nhan(p, dsps[i]);
    }
    return p;
}

void NghichDaoCacPS(PHANSO dsps[], int n)
{
    for(int i=0; i<n; i++)
    {
        dsps[i]=NghichDao(dsps[i]);
    }
}
```

Kết quả ví dụ khi chạy chương trình:

```
Nhap so luong phan so: 5

Nhap vao phan so thu 1:
Nhap tu so: 1
Nhap mau so: 3

Nhap vao phan so thu 2:
Nhap tu so: 7
Nhap mau so: 4

Nhap vao phan so thu 3:
Nhap tu so: 9
Nhap mau so: 7

Nhap vao phan so thu 4:
```

```
Nhap tu so: 5
Nhap mau so: 6

Nhap vao phan so thu 5:
Nhap tu so: 4
Nhap mau so: 7

Mang cac phan so vua nhap: 1/3  7/4     9/7     5/6     4/7
Phan so co gia tri lon nhat: 7/4
Tong gia tri cac phan so co trong mang: 401/84
Tich gia tri cac phan so co trong mang: 5/14
Mang phan so sau khi nghich dao cac phan tu: 3  4/7     7/9     6/5     7/4
```

II. BÀI TẬP

II.1. Bài tập cơ bản

1. Viết chương trình sử dụng con trỏ cấu trúc để hiển thị giờ, phút, giây ra màn hình, và tính khoảng cách giữa 2 mốc thời gian.
2. Viết chương trình sử dụng con trỏ cấu trúc thể hiện ngày, tháng, năm ra màn hình, và tính khoảng cách giữa 2 ngày.
3. Viết chương trình khai báo kiểu dữ liệu thể hiện một số phức. Sử dụng kiểu này để viết hàm tính tổng, hiệu, tích của hai số phức.
4. Viết chương trình khai báo kiểu dữ liệu để biểu diễn một phân số. Hãy viết hàm thực hiện những công việc sau:
   - Tính tổng, hiệu, tích, thương hai phân số.
   - Rút gọn phân số.
   - Qui đồng hai phân số.
   - So sánh hai phân số.
5. Viết chương trình khai báo kiểu dữ liệu để biểu diễn một hỗn số. Hãy viết hàm thực hiện những công việc sau :
   - Đổi hỗn số sang phân số
   - Tính tổng, tích hai hỗn số
6. Viết chương trình khai báo kiểu dữ liệu để biểu diễn một điểm trong hệ tọa độ 0xy . Hãy viết hàm thực hiện các công việc sau:
   - Tìm những điểm đối xứng của nó qua tung độ, hoành độ, toạ độ tâm.
   - Hãy tính tổng, hiệu, tích của hai điểm trong mặt phẳng toạ độ 0xy.
   - Tính khoảng cách giữa hai điểm.

7. Cho một hình trụ có các thông tin sau: BanKinh (bán kính hình trụ kiểu số thực), ChieuCao (chiều cao hình trụ kiểu số thực). Hãy thực hiện các công việc sau.
   - Nhập dữ liệu cho hình trụ trên.
   - Tính diện tích xung quanh, diện tích toàn phần, thể tích hình trụ.

II.2. Bài Tập Luyện Tập

8. Viết chương trình tạo một mảng các số phức. Hãy viết hàm tính tổng, tích các số phức có trong mảng.
9. Viết chương trình tạo một mảng các phân số. Hãy viết hàm thực hiện các công việc sau :
   - Tính tổng tất cả các phân số (kết quả dưới dạng phân số tối giản)
   - Tìm phân số lớn nhất, phân số nhỏ nhất.
   - Sắp xếp mảng tăng dần.
10. Viết chương trình khai báo kiểu dữ liệu STACK (cơ chế LIFO). Viết hàm làm những công việc sau :
    - Kiểm tra STACK rỗng
    - Kiểm tra STACK đầy
    - Thêm phần tử vào STACK
    - Lấy phần tử ra khỏi STACK
11. Tổ chức dữ liệu để quản lí sinh viên bằng cấu trúc mẫu tin trong một mảng N phần tử, mỗi phần tử có cấu trúc như sau:
    - *Mã sinh viên.*
    - *Tên.*
    - *Năm sinh.*
    - *Điểm toán, lý, hoá, điểm trung bình.*

    Viết chương trình thực hiện những công việc sau:
    - Nhập danh sách các sinh viên cho một lớp học.
    - Xuất danh sách sinh viên ra màn hình.
    - Tìm sinh viên có điểm trung bình cao nhất.
    - Sắp xếp danh sách lớp theo thứ tự tăng dần của điểm trung bình.
    - Sắp xếp danh sách lớp theo thứ tự giảm dần của điểm toán.

- Tìm kiếm và in ra các sinh viên có điểm trung bình lớn hơn 5 và không có môn nào dưới 3.
- Tìm sinh viên có tuổi lớn nhất.
- Nhập vào tên của một sinh viên. Tìm và in ra các thông tin liên quan đến sinh viên đó (nếu có).

12. Tổ chức dữ liệu quản lí danh mục các bộ phim VIDEO, các thông tin liên quan đến bộ phim này như sau:
    - *Tên phim (tựa phim).*
    - *Thể loại (3 loại : hình sự, tình cảm, hài).*
    - *Tên đạo diễn.*
    - *Tên điễn viên nam chính.*
    - *Tên diễn viên nữ chính.*
    - *Năm sản xuất.*
    - *Hãng sản xuất*

    Viết chương trình thực hiện những công việc sau :
    - Nhập vào bộ phim mới cùng với các thông tin liên quan đến bộ phim này.
    - Nhập một thể loại: In ra danh sách các bộ phim thuộc thể loại này.
    - Nhập một tên nam diễn viên. In ra các bộ phim có diễn viên này đóng.
    - Nhập tên đạo diễn. In ra danh sách các bộ phim do đạo diễn này dàn dựng.
13. Một thư viện cần quản lý thông tin về các đầu sách. Mỗi đầu sách bao gồm các thông tin sau : MaSSach (mã số sách), TenSach (tên sách), TacGia (tác giả), SL (số lượng các cuốn sách của đầu sách). Viết chương trình thực hiện các chức năng sau:
    - Nhập vào một danh sách các đầu sách (tối đa là 100 đầu sách)
    - Nhập vào tên của quyển sách. In ra thông tin đầy đủ về các sách có tên đó, nếu không có thì tên của quyển sách đó thì báo là :Không Tìm Thấy.
    - Tính tổng số sách có trong thư viện.
14. Viết chương trình tạo một mảng danh sách các máy tính của một cửa hàng, thông tin của một máy tính bao gồm :
    - *Loại máy*
    - *Nơi sản xuất*

- *Thời gian bảo hành*

- Viết hàm nhập một dãy các loại máy tính có thông tin như trên.
- Hãy viết hàm thống kê xem có bao nhiêu máy có thời gian bảo hành là 1 năm.
- In ra danh sách các máy tính có xuất xứ từ Mỹ.

15. Để lắp ráp một máy vi tính hoàn chỉnh cần phải có tối thiểu 10 linh kiện loại A và có thể lắp bổ sung thêm vào khoảng tối đa 8 linh kiện loại B. Tại một cửa hàng vi tính cần quản lý bán hàng các loại linh kiện tại cửa hàng. Thông tin về một loại linh kiện gồm có: Tên linh kiện, quy cách , loại, đơn giá loại 1 (chất lượng tốt – số nguyên), đơn giá loại 2 (chất lượng thường – số nguyên). Viết chương trình thực hiện những công việc sau :
    - Nhập vào thông tin về các linh kiện có ở cửa hàng.
    - Xuất danh sách các linh kiện đã nhập theo thứ tự tăng dần của loại linh kiện và tên linh kiện.
    - Cho biết đã có đủ 10 linh kiện loại A cần thiết lắp ráp máy hay chưa?
16. Một cửa hàng cần quản lý các mặt hàng, thông tin một mặt hàng bao gồm:
    - *Mã hàng.*
    - *Tên mặt hàng.*
    - *Số lượng.*
    - *Đơn giá.*
    - *Số lượng tồn.*
    - *Thời gian bảo hành (tính theo đơn vị tháng).*
    - Hãy nhập vào một danh sách các mặt hàng.
    - Tìm mặt hàng có số lượng tồn nhiều nhất.
    - Tìm mặt hàng có số lượng tồn ít nhất.
    - Tìm mặt hàng có giá tiền cao nhất.
    - In ra những mặt hàng có thời gian bảo hành lớn hơn 12 tháng.
    - Sắp xếp các mặt hàng theo thứ tự tăng dần của số lượng tồn.
17. Viết chương trình quản lý hồ sơ nhân viên trong một công ty, chương trình thực hiện những công việc sau :
    - *Họ và tên.*
    - *Phái.*

- *Ngày sinh.*
- *Địa chỉ.*
- *Lương cơ bản.*
- *Bảo hiểm xã hội.*
- *Thưởng.*
- *Phạt.*
- *Lương thực lĩnh = lương cơ bản + thưởng – BH xã hội – phạt.*

- Nhập vào hồ sơ của các nhân viên trong công ty.
- Xuất danh sách các nhân viên theo lương thực lĩnh giảm dần bằng 2 cách sau :
  - *Cấp phát vùng nhớ tĩnh.*
  - *Cấp phát vùng nhớ động.*

18. **(*)** Viết chương trình quản lý lớp học của một trường. Các thông tin của một lớp học như sau :
    - *Tên lớp.*
    - *Sĩ số.*
    - *Danh sách các sinh viên trong lớp.*

    - Nhập vào danh sach các lớp với thông tin yêu cầu như trên.
    - In danh sách các lớp có trên 5 sinh viên có điểm trung bình loại giỏi.
    - Tìm lớp có nhiều sinh viên nhất.
    - Tìm lớp có ít sinh viên nhất.
    - Tìm sinh viên có điểm trung bình cao nhất.
    - Tìm lớp có số lượng sinh viên đạt điểm trung bình loại giỏi nhiều nhất.
19. Viết chương trình quản lý vé tàu, thông tin một vé tàu như sau :
    - *Ngày giờ khởi hành, ngày giờ đến.*
    - *Ga đi, ga đến.*
    - *Loại tàu, loại chỗ ngồi (ngồi, nằm, cứng, mềm).*
    - *Số toa, số ghế.*

    - Viết hàm nhập vào danh sách các vé tàu.
    - In danh sách các vé tàu có ga đến là Huế.
    - In danh sách các vé tàu có ga đến là Hà Nội và đi ngày 8/6/2005.
    - Đếm xem có bao nhiêu khách đi tàu loại chỗ ngồi là *nằm cứng*.

20. Viết chương trình tính tiền điện hàng tháng của các hộ gia đình, thông tin các khách hàng như sau :
    - *Kỳ thu, từ ngày……đến ngày.*
    - *Tên khách hàng, mã khách hàng.*
    - *Địa chỉ.*
    - *Điện năng tiêu thụ (Kwh).*
    - Nhập vào danh sách các khách hàng.
    - Xuất danh sách hoá đơn theo thứ tự tăng dần của điện năng tiêu thụ.
    - Tính tiền điện của các khách hàng theo quy định sau.
        - *100 kw đầu tiên là 550 đ / kw*
        - *50 kw tiếp theo là 900 đ / kw*
        - *50 kw tiếp theo là 1210 đ / kw*
        - *Thuế 10 % trên tổng số tiền phải trả*
    - Tính tổng số tiền thu được của các khách hàng.

III. KẾT LUẬN

- Kiểu dữ liệu có cấu trúc cho phép ta định nghĩa những kiểu dữ liệu bất kỳ trên cơ sở là những kiểu dữ liệu cơ bản có sẵn trong ngôn ngữ lập trình.
- Khi xây dựng xong kiểu dữ liệu mới ta phải **định nghĩa những thao tác** cho kiểu dữ liệu đó.
- Những kiểu dữ liệu tự định nghĩa này thông thường có rất nhiều thành phần, mỗi thành phần cũng có thể là một kiểu dữ liệu tự định nghĩa, vấn đề là ta chọn kiểu dữ liệu cơ bản nào để xây dựng nên chúng sao cho **phù hợp về mặt kiểu dữ liệu** và **phù hợp về kích thước lưu trữ** (vừa đủ).
- Các sử dựng những kiểu dữ liệu tự định nghĩa cũng giống như các kiểu dữ liệu cơ bản. Muốn sử dụng phải khai báo biến, khi truy cập các thành phần phải truy cập theo quy ước.
- Nếu thành phần cấu trúc có kiểu dữ liệu là số thực thì khi sử dụng hàm **scanf()** phải thông qua biến trung gian rồi gán lại cho thành phần cấu trúc đó.
- Đối với mảng các kiểu dữ liệu có cấu trúc ta nên **xử lý cho từng thành phần cấu trúc** rồi mới xử lý cho mảng cấu trúc bằng cách dùng vòng lặp.