CÁC KIỂU DỮ LIỆU NÂNG CAO

# Lý thuyết:

Các chương trình ứng dụng trong thực tế đòi hỏi lưu trữ các kiểu dữ liệu khác nhau. Tuy nhiên, các kiểu dữ liệu của C mà chúng ta đã được học có thể không đủ trong các trường hợp đó. Vì vậy, C cho phép tạo ra các kiểu dữ liệu do người dùng định nghĩa. Một trong những kiểu như vậy là **cấu trúc** (**structure**). Một cấu trúc là một tập các biến được nhóm lại với nhau có cùng tên.

**struct** cat

{

char name [25];

char author [20];

int page;

float price;

};

Từ khóa **struct** báo cho trình biên dịch biết rằng một structure được định nghĩa. Nhãn **cat** không phải là tên biến, vì không phải ta đang khai báo biến. Nó là một **tên kiểu**. Các phần tử của cấu trúc được định nghĩa trong dấu móc, và kết thúc toàn bộ câu lệnh bằng một dấu chấm phẩy.

Khai báo biến kiểu cấu trúc

Cũng như với **int**, **float** và các kiểu dữ liệu khác, ta có thể có một số bất kỳ các biến có kiểu cấu trúc đã cho. Trong một chương trình, có thể khai báo hai biến books1 và books2 có kiểu cấu trúc **cat** . Điều này có thể thực hiện được theo nhiều cách.

struct cat

{

char bk\_name[25];

char author[20];

int edn;

float price;

} books1, books2;

**hoặc**

struct cat books1, books2;

**hoặc**

struct cat books1;

struct cat books2;

Khởi tạo biến cấu trúc

Giống như các biến và mảng, các biến kiểu cấu trúc có thể được khởi tạo tại thời điểm khai báo. Hình thức tương tự như cách khởi tạo mảng. Xét cấu trúc sau dùng để lưu số thứ tự và tên nhân viên:

struct employee

{

int no;

char name[20];

};

Các biến **emp1** và **emp2** có kiểu **employee** có thể được khai báo và khởi tạo như sau:

struct employee emp1 = {346, “Abraham”};

struct employee emp2 = {347, “John”};

**Thực hiện câu lệnh gán với các biến cấu trúc**

Có thể gán giá trị của một biến cấu trúc cho một biến khác cùng kiểu bằng cách sử dụng câu lệnh gán đơn giản. Chẳng hạn, nếu books1 và books2 là các biến cấu trúc có cùng kiểu, thì câu lệnh sau là hợp lệ.

books2 = books1;

Cấu trúc lồng trong cấu trúc

Một cấu trúc có thể lồng trong một cấu trúc khác.

**struct** cat

{

char name [25];

char author [20];

int page;

float price;

};

struct issue

{

char borrower [20];

char dt\_of\_issue[8];

**struct cat** books;

};

Mảng các cấu trúc

Một trong những cách sử dụng thông thường của cấu trúc là mảng cấu trúc. Để khai báo một mảng các cấu trúc, một cấu trúc sẽ được định nghĩa trước, và sau đó một biến mảng có kiểu đó sẽ được khai báo. Ví dụ như, để khai báo một mảng các cấu trúc có kiểu cat, câu lệnh sẽ là:

**struct cat** books[50];

Giống như tất cả các biến, mảng các cấu trúc bắt đầu tại chỉ số 0.

books[4].author

sẽ tương ứng là thành phần author của phần tử thứ tư trong mảng **books**.

# Bài tập thực hành

**BÀI TẬP 1**

**1. Thời lượng:**

**2. Mô tả bài toán:**

**3. Các bước thực hiện:**

struct Sinhvien

{

char hoten[30];

int maso;

float van;

float toan;

float anh;

};

int main()

{

Sinhvien a;

printf("Ma so:");

scanf("%d",&a.maso);

fflush stdin;

printf("Ho ten:");

gets(a.hoten);

printf("Diem Van:");

scanf("%f",&a.van);

printf("Diem Toan:");

scanf("%f",&a.toan);

printf("Diem Anh:");

scanf("%f",&a.anh);

printf("\nHoten:%s",a.hoten);

printf("\nMaso:%d",a.maso);

printf("\nVan:%f",a.van);

printf("\nToan:%f",a.toan);

printf("\nAnh:%f",a.anh);

getche();

}

**BÀI TẬP 2**

**1. Thời lượng:**

**2. Mô tả bài toán:**

**3. Các bước thực hiện:**

struct Sinhvien

{

char hoten[30];

int maso;

float van;

float toan;

float anh;

};

int main()

{

Sinhvien a[10];//Sinhvien a[0];Sinhvien a[1]

int n;

printf("Hay nhap so sinh vien:");

scanf("%d",&n);

for(int i=0;i<n;i++)

{

printf("Ma so:");

scanf("%d",&a[i].maso);

fflush stdin;

printf("Ho ten:");

gets(a[i].hoten);

printf("Diem Van:");

scanf("%f",&a[i].van);

printf("Diem Toan:");

scanf("%f",&a[i].toan);

printf("Diem Anh:");

scanf("%f",&a[i].anh);

}

printf("\n\nThong tin Sinh vien la:\n");

for(int i=0;i<n;i++)

{

printf("\nSinh vien thu %d",i);

printf("\nHoten:%s",a[i].hoten);

printf("\nMaso:%d",a[i].maso);

printf("\nVan:%0.1f",a[i].van);

printf("\nToan:%0.1f",a[i].toan);

printf("\nAnh:%0.1f",a[i].anh);

float dtb;

dtb=(a[i].van + a[i].toan + a[i].anh)/3;

printf("Diem trung binh: %0.1f",dtb);

}

getche();

}

**BÀI TẬP 3**

**1. Thời lượng:**

**2. Mô tả bài toán:**

**3. Các bước thực hiện:**

struct mark

{

float van;

float toan;

float anh;

};

struct Sinhvien

{

char hoten[30];

int maso;

mark diemso;

};

int main()

{

Sinhvien a[10];//Sinhvien a[0];Sinhvien a[1]

int n;

printf("Hay nhap so sinh vien:");

scanf("%d",&n);

for(int i=0;i<n;i++)

{

printf("Ma so:");

scanf("%d",&a[i].maso);

fflush stdin;

printf("Ho ten:");

gets(a[i].hoten);

printf("Diem Van:");

scanf("%f",&a[i].mark.van);

printf("Diem Toan:");

scanf("%f",&a[i].mark.toan);

printf("Diem Anh:");

scanf("%f",&a[i].mark.anh);

}

printf("\n\nThong tin Sinh vien la:\n");

for(int i=0;i<n;i++)

{

printf("\nSinh vien thu %d",i);

printf("\nHoten:%s",a[i].hoten);

printf("\nMaso:%d",a[i].maso);

printf("\nVan:%0.1f",a[i].mark.van);

printf("\nToan:%0.1f",a[i].mark.toan);

printf("\nAnh:%0.1f",a[i].mark.anh);

float dtb;

dtb=(a[i].mark.van + a[i].mark.toan + a[i].mark.anh)/3;

printf("Diem trung binh: %0.1f",dtb);

}

getche();

}

**BÀI TẬP 4**

**1. Thời lượng:**

**2. Mô tả bài toán:**

Nhập và xuất thông số của n hình tròn .gồm : tâm (x,y) , bán kính

(tương tự như bài struct lồng struct ở ví dụ STRUCT)

Gợi ý:

struct TOADO

{

X : kiểu Float

Y : kiểu Float

}

struct HINHTRON

{

* tâm : kiểu TOADO
* bán kính : kiểu float

}

* **Mẫu nhập của chương trình như sau :**

Hinh tron 1:

Nhap tam : …

X: …

Y:…

Nhap ban bkinh:…

Hinh tron 2:

Nhap tam : ….

X: ….

Y: .…

Nhap ban bkinh:…

* **Mẫu xuất của chương trình như sau :**

Hình tròn 1: bán kính ….. Tâm ( X,Y)

Hình tròn 2: bán kính ….. Tâm ( X,Y)

**3. Các bước thực hiện:**

**BÀI TẬP 5**

**1. Thời lượng:**

**2. Mô tả bài toán**: Nhập vào độ dài 3 cạnh của n tam giác . Xét xem với 3 cạnh đó có thể tạo thành tam giác hay không . Nếu là tam giác thì tính chu vi tam giác đó.

Màn hình nhập xuất như sau :

* **Mẫu nhập của chương trình như sau :**

Hay nhap vao so luong tam giac : 2

Tam giac 1:

Nhap Canh 1:1

Nhap Canh 2:1

NhapCanh 3:4

Tam giac 2:

Nhap Canh 1:1

Nhap Canh 2:1

Nhap Canh 3:1

* **Mẫu xuất của chương trình như sau :**

+Hinh 1:

Ko phai tam giac

+Hinh 2:

la tam giac , Tinh chu vi

**3. Các bước thực hiện:**

**BÀI TẬP 6**

**1. Thời lượng:**

**2. Mô tả bài toán:**

Nhập vào thông tin của các nhân viên (số lượng cho nhập tùy ý) (mỗi nhân viên gồm mã NV, họ tên , lương) .

a/Hãy in ra nhân viên có lương thấp nhất

b/sắp xếp nhân viên theo lương tăng dần

**3. Các bước thực hiện:**

**BÀI TẬP 7**

**1. Thời lượng:**

**2. Mô tả bài toán:** Chúng ta hãy viết một chương trình C để cài đặt một hệ thống quản lý thư viện cơ bản. Hệ thống lưu trữ danh mục sách và ghi nhận các giao dịch mượn và trả sách. Sử dụng hệ thống này, ta có thể thêm thông tin chi tiết của một sách, ghi nhận các giao dịch mượn/trả sách và sắp xếp các ghi nhận này.

Mẫu kết quả của chương trình như sau:

Select from Menu

1. Add book names
2. Record Issue/Return
3. Sort Transactions
4. Exit

Enter choice:

Nếu nhập vào 1, mẫu kết xuất của chương trình sẽ là:

Book code: 1

Book name: Detective

Author: Hailey

Number of copies: 3

Continue? (y/n): y

Nếu nhập vào 2, mẫu kết xuất của chương trình sẽ là:

Book code: 1

Issue or Return? (I/R): I

Date: 2 22 03

Continue? (y/n): y

Nếu nhập vào 3, mẫu kết xuất của chương trình sẽ là:

Book code 1 had 3 transactions

Book code 2 had 1 transactions

Book code 3 had 2 transactions

Book code 4 had 0 transactions

Book code 5 had 4 transactions

**3. Các bước thực hiện:**

1. Định nghĩa một cấu trúc để lưu trữ chi tiết sách. Câu lệnh sẽ là:

struct book\_st{

int book\_cd;

char book\_nm[30];

char author[30];

int copies;

};

1. Định nghĩa một cấu trúc để lưu trữ các giao dịch mượn/trả sách. Lưu ý rằng ngày mượn/trả cũng sẽ là một cấu trúc và cũng phải được định nghĩa. Câu lệnh sẽ là:

struct date\_st { int month;

int day;

int year;};

struct tran\_st { int book\_code;

char tran\_type;

struct date\_st tran\_dt; };

1. Khai báo các biến có hai kiểu cấu trúc trên. Để thực hành, ta giả sử rằng cần lưu trữ chi tiết của 5 quyển sách và 10 giao dịch. Câu lệnh sẽ là:

struct book\_st books[5];

struct tran\_st trans[10];

1. Viết một vòng lặp để hiển thị menu các thao tác mà chương trình có thể thực hiện.. Câu lệnh sẽ là:

while(choice != 4)

{ clrscr();

printf("\nSelect from Menu\n1. Add book names\n2. Record Issue/Return\n3. Sort Transactions\n4. Exit\n\nEnter choice: ");

scanf("%d", &choice);

.

.

.

}

1. Nếu thao tác được chọn là thêm sách, thì nhập các thông tin chi tiết của sách trong một vòng lặp. Câu lệnh sẽ là:

for(i = 0; i < 5 && addflag == 'y'; i++)

{ books[i].book\_cd = i + 1;

printf("\n\nBook code: %d\n\nBook name: ", i + 1);

scanf("%s", books[i].book\_nm);

printf("\nAuthor: ");

scanf("%s", books[i].author);

printf("\nNumber of copies: ");

scanf("%d", &books[i].copies);

printf("\n\nContinue? (y/n): ");

scanf(" %c", &addflag);

}

1. Nếu thao tác được chọn là thêm giao dịch, đặt một vòng lặp để nhập các thông tin chi tiết của giao dịch. Câu lệnh sẽ là:

for(i = 0; i < 10 && addflag == 'y'; i++)

{ printf("\n\nBook code: ");

scanf("%d", &trans[i].book\_code);

printf("\nIssue or Return?(I/R): ");

scanf(" %c", &trans[i].tran\_type);

printf("\nDate: ");

scanf("%d %d %d",

&trans[i].tran\_dt.month,&trans[i].tran\_dt.day, &trans[i].tran\_dt.year);

printf("\n\nContinue? (y/n): ");

scanf("%c", &addflag);

}

1. Nếu thao tác được chọn là sắp xếp các giao dịch, thì truyền tham số mảng cấu trúc vào hàm. Hàm sẽ sắp xếp mảng theo mã sách sử dụng phương pháp buble sort. Câu lệnh sẽ là:

for(i = 0; i < 10; i++)

for(j = i + 1; j < 10; j++)

{

if(tran[i].book\_code > tran[j].book\_code)

{

temptran=tran[i];

tran[i]=tran[j];

tran[j]=temptran;

}

}

1. Hiển thị số giao dịch cho mỗi quyển sách trong hàm sắp xếp. Câu lệnh sẽ là:

for(i = 0, j = 0; i < 10; j = 0)

{ tempcode = tran[i].book\_code;

while(tran[i].book\_code == tempcode && i < 10)

{

j++;

i++;

}

printf("\nBook code %d had %d transactions", tempcode, j);

}

**BÀI TẬP 8**

**1. Thời lượng:**

**2. Mô tả bài toán:** Viết một chương trình C để lưu trữ các thông tinh về sinh viên trong một cấu trúc. Dữ liệu phải bao gồm mã sinh viên, tên sinh viên, khóa học đã đăng ký và năm đăng ký. Viết một hàm để hiển thị các thông tin chi tiết của các sinh viên đã nhập học trong một năm học nào đó. Viết một hàm khác để xác định và hiển thị thông tin chi tiết của một sinh viên khi biết mã của sinh viên đó.

**3. Các bước thực hiện:**

* 1. Định nghĩa một cấu trúc để lưu trữ thông tin chi tiết của sinh viên.
  2. Khai báo và khởi tạo biến cấu trúc với thông tin chi tiết của 10 sinh viên.
  3. Viết vòng lặp để hiển thị một menu các thao tác mà chương trình có thể thực hiện.
  4. Nhận vào lựa chọn danh mục và gọi hàm thích hợp với tham số là mảng cấu trúc.
  5. Trong hàm dùng để hiển thị thông tin chi tiết của các sinh viên nhập học trong một năm, viết chương trình để thực hiện nhập vào năm học cần được hiển thị thông tin, sau đó sử dụng vòng lặp để kiểm tra năm nhập học của từng sinh viên, nếu trùng với năm cần hiển thị thông tin yêu cầu thì hiển thị thông tin của sinh viên đó. Ngoài ra, hàm này còn cho phép người dùng có thể tiếp tục thực hiện việc hiển thị thông tin của những năm khác cho đến khi họ không muốn sử dụng chức năng này nữa.
  6. Hàm dùng để hiển thị thông tin chi tiết của sinh viên cho phép nhập vào mã của sinh viên,dùng một vòng lặp để kiểm tra mã của mỗi sinh viên,nếu mã của sinh viên nào trùng với mã đã được nhập thì hiển thị thông tin chi tiết của sinh viên đó. Ngoài ra, hàm này còn cho phép người dùng có thể tiếp tục thực hiện việc hiển thị thông tin của những sinh viên khác cho đến khi họ không muốn sử dụng chức năng này nữa

**BÀI TẬP 9**

**1. Thời lượng:**

**2. Mô tả bài toán:**

Viết một chương trình C để lưu trữ thông tin chi tiết của nhân viên trong một mảng cấu trúc. Thông tin của một nhân viên phải bao gồm mã nhân viên, tên, lương và ngày vào làm. Ngày vào làm phải được lưu trong một cấu trúc khác. Chương trình phải thực hiện các thao tác sau đây dựa trên sự lựa chọn trongmenu các chức năng của chương trình:

* 1. Tăng lương theo các luật sau:

|  |  |
| --- | --- |
| **Salary Range** | **Percentage increase** |
| <= 2000 | 15% |
| > 2000 and <= 5000 | 10% |
| >5000 | No increase |

* 1. Hiển thị thông tin chi tiết của các nhân viên đã làm việc trong công ty từ 10 năm trở lên.

**3. Các bước thực hiện:**

**Link tham khảo**: