|  |
| --- |
| **Mục tiêu:**   1. ***Tổ chức dữ liệu kiểu cấu trúc và danh sách đặc gồm các phần tử kiểu cấu trúc.*** 2. ***Tổ chức dữ liệu kiểu con trỏ cấu trúc và danh sách liên kết đơn*** 3. ***Cài đặt một số ứng dụng trên kiểu cấu trúc.*** |

**Bài toán: Tổ chức và Quản lý danh mục môn học tại trường học.**

* **Vấn đề 1: Quản lý môn học**

Thông tin về các môn học được phòng đào tạo của một trường đại học tổ chức và lưu trữ trong một danh sách liên kết đơn với nút đầu được trỏ bởi ***headC*** gọi là danh sách môn học (DSMH). Mỗi nút của danh DSMH gồm 3 trường nội dung là ***ma\_mh, ten\_mh, so\_tc*** và một con trỏ ***next*** trỏ đến nút kế tiếp của danh sách được định nghĩa như sau:

**typedef struct \_monhoc**

**{**

**char** \*ma\_mh;//Mã môn học (khóa chính)

**char** \*ten\_mh;//Tên môn học

**int** so\_tc;// Số tín chỉ

**}\_monhoc\_node;**

**typedef \_monhoc\_node \*type\_monhoc; //Con trỏ kiểu \_monhoc\_node**

**1.1.** Viết chương trình tạo danh sách môn học với các dữ liệu được nhập vào từ bàn phím (quá trình nhập kết thúc khi ***ma\_mh*** rỗng) hoặc đọc từ file (với số phần tử lưu ở dòng đầu tiên của file).

**1.2.** Đọc dữ liệu từ danh sách và hiển thị thông tin các môn học lên màn hình.

**1.3.** Xây dựng hàm **type\_monhoc dinhvi(type\_monhoc headC, char \*Bma\_mh)** trả về địa chỉ của nút thuộc danh sách ***head*** có mã môn học là ***Bma\_mh*** hoặc trả về ***NULL*** nếu mọi nút thuộc danh sách head đều có ***ma\_mh*** khác ***Bma\_mh***.

* **Vấn đề 2: Quản lý giảng viên**

Để quản lý các môn học được phân công giảng dạy của mỗi giảng viên, người ta lại sử dụng một danh sách liên kết đơn khác gọi là danh sách giảng viên (DSGV) với nút đầu được trỏ bởi ***headT***. Mỗi nút của DSGV là một bản ghi gồm 4 trường; trường ***ma\_gv*** để lưu mã giảng viên (đây là trường khóa), trường ***hoten\_gv*** để lưu họ tên giảng viên, trường ***down*** lưu địa chỉ của nút tiếp theo và trường ***dsmon*** để lưu địa chỉ nút đầu của một danh sách khác chứa mã môn học do giảng viên đó đảm nhận gọi là danh sách môn của giảng viên (DSMGV). Mỗi nút của DSMGV là một cấu trúc gồm 2 trường: ***ma\_Mon*** để lưu mã môn học đã được đăng ký trong DSMH (danh sách trỏ đầu bởi ***headC***) và trường ***tiep*** lưu địa chỉ của nút tiếp theo trong danh sách này. Hai cấu trúc dữ liệu có mô tả ở trên được định nghĩa như sau:

**typedef struct \_maM**

**{ char** \*ma\_Mon;//Mã môn học

**\_maM** \*tiep;

**}\_maM\_node;**

**typedef \_maM\_node \*type\_maM;**

**typedef struct \_giangvien**

**{ char** \*ma\_gv; //Mã giảng viên

**char** \*hoten\_gv; //Họ tên giảng viên

**type\_maM** dsmon; //Quản lý DSMGV

**\_giangvien** \*down;

**}\_giangvien\_node;**

**typedef \_giangvien\_node \*type\_giangvien; //Con trỏ kiểu \_giangvien\_node**

**headT**



**N.V.An**

**dsmon**

**CS001**

**IS004**  •

**GV02**

**T.N.Bình**

**CS002** •

…

**2.1.** Viết chương trình lấy dữ liệu bằng cách nhập từ bàn phím (quá trình nhập kết thúc khi ***ma\_gv*** rỗng) hoặc đọc từ file (với số phần tử lưu ở dòng đầu tiên của file) để tạo danh sách giảng viên được sắp theo thứ tự tăng dần của ***ma\_gv*** (quá trình chèn phần tử vào danh sách phải đảm bảo DSGV sắp tăng) và ***ma\_mon*** của DSMGV phải được kiểm tra để đảm bảo đã tồn tại môn học đó trong SDMH..

**2.2.** Viết hàm **void danh\_sach\_mon\_cua\_gv(type\_giangvien headT, char \*Bmagv)** để liệt kê các môn học do giảng viên có mã giảng viên là *Bmagv* đảm nhận. Thông tin hiển thị lên màn hình theo mẫu sau:

+ + + +

| STT | TEN MON HOC | SO TC |

+ + + +

| 1 | Ngon ngu lap trinh C | 5 |

+ + + +

| 2 | Cau truc du lieu & Giai thuat | 4 |

+ + + +

| 3 | ....... | ... |

+ + + +

**2.3.** Viết hàm **void danh\_sach\_giangvien(type\_giangvien headT, char \*Bma\_mon)** để liệt kê các giảng viên đảm nhận môn học có mã môn học là ***Bma\_mon***. Thông tin hiển thị lên màn hình theo mẫu sau:

+ + + +

| STT | TEN GIANG VIEN | MA GV |

+ + + +

| 1 | Nguyen Thanh Nam | GV07 |

+ + + +

| 2 | Phan Huu Hoang Nhan | GV12 |

+ + + +

| 3 | ....... | ... |

**Vấn đề 3:** Xét đa thức tổng quát: P(x) = anxn + an-1.xn-1  +… + a2.x2 +a1x + a0 =

(an,n) + (an-1,n-1) + ...+ (a2,2) + (a1,1) + (a0,0)

Dùng danh sách liên kết đơn để biểu diễn đa thức. Mỗi node của DSLK sẽ chứa các thông tin sau:

Địa chỉ của node tiếp theo trong danh sách

Số mũ

Hệ số

HeSo

SoMu

LINK

Giả sử ta có hai đa thức:

P1(x) = 3x7 + 5x6  + x5 + 2x3 - 7x + 9

P2(x) = 2x7 + 3x5  - 5x4 + 2x3 + x - 8

Cộng hai đa thức P1 và P2 ta được đa thức tổng:

T(x) = 5x7 + 5x6 + 4x5  - 5x4 + 4x3 - 6x + 1

Như vậy, với đa thức P1(x) ở trên thì danh sách biểu diễn nó có dạng:

3

7

5

6

1

5

2

3

-77

1

9

0

A

Ở đây A là con trỏ lưu địa chị nút đầu tiên của danh sách.

1. Định nghĩa cấu trúc **Node** với mỗi phần tử của Node gồm có 3 thành phần: hệ số, số mũ và địa chỉ của Node tiếp theo.

struct Node{

float heSo;

int soMu;

Node \*link;

};

struct List

{

Node \*first, \*last;

};

1. Viết hàm khởi tạo một danh sách rỗng

void Init(List &L)

3. Viết hàm Node \*GetNode(float heSo, int soMu)để tạo một node có hệ số là heSo và số mũ là soMu. Hàm trả về con trỏ trỏ đến node vừa tạo.

4. Viết hàm void InsertLast (List &L, float heSo, int soMu) thêm một node với dữ liệu là heSo và soMu vào cuối danh sách

Nếu đa thức (3,7) + (5,6) + (1,5) + (2,3) + (-7,1) + (9,0) + (4,5) và ta thêm vào số hạng (4,5)

5. Viết hàm nhập và một đa thức void NhapDaThuc(List &L)

6. Viết hàm xuất ra đa thức dạng P(x) = anx^n + an-1.x^n-1 +… + a2.x^2 +a1x + a0

7. Viết hàm cộng hai đa thức void CongDaThuc(List d1, List d2, List &d3)

Với d3 = d2 + d1

Vấn đề 4: Tập hợp

Một tập hợp được lưu trữ trong một danh sách liên kết. Mỗi phần tở của danh sách liên kết chứa một phần tử của tập hợp

Cấu trúc một phần tử và của tập hợp như sau:

1. Viết hàm TAPHOP \* duyNhat (TAPHOP A) trả về tập hợp B bao gồm các số (không trùng nhau) của tập hợp A và tập hợp B có thứ tự tăng dần

struct PHANTU{

int giaTri;

Node \*link;

};

struct TAPHOP

{

Node \*first, \*last;

};

VD:

Tập hợp A: 1 5 3 7 5 9 7

Tập hợp B: 1 3 5 7 9

1. Viết hàm TAPHOP \* tapHieu (TAPHOP A, TAPHOP B)trả về tập hợp C là hiệu của tập hợp A và B. ( C có thứ tự tăng dần)

VD:

* Tập hợp A: 1 5 3 7 9 4 2
* Tập hợp B: 9 6 2 3 8
* Tập hợp C: 1 4 5 7

1. Viết hàm TAPHOP \* tapGiao (TAPHOP A, TAPHOP B)trả vể tập hợp C là giao của 2 tập hợp A, B

* Tập hợp A: 1 5 3 7 9 4 2
* Tập hợp B: 9 6 2 3 8
* Tập hợp C: 2 3 9

1. Viết hàm TAPHOP \* tapHop (TAPHOP A, TAPHOP B)trả vể tập hợp C là hợp của 2 tập hợp A, B

* Tập hợp A: 1 5 3 7 9 4 2
* Tập hợp B: 9 6 2 3 8
* Tập hợp C: 1 2 3 4 5 6 7 8 9

**Yêu cầu : Thành lập nhóm 2-3 sinh viên, Hiện thực bài toán bằng chương trình (một sản phẩm phần mềm hoàn thiện). Viết báo cáo và nộp bản cứng (in giấy) trình bày : Giải thuật từng modun và Full Code.**