

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI PHÂN HIỆU TẠI TP. HỒ CHÍ MINH**

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO**

**MÔN: KỸ THUẬT LẬP TRÌNH**

# ĐỀ TÀI: BÀI TẬP LỚN

Giảng viên: ThS. TRẦN PHONG NHÃ

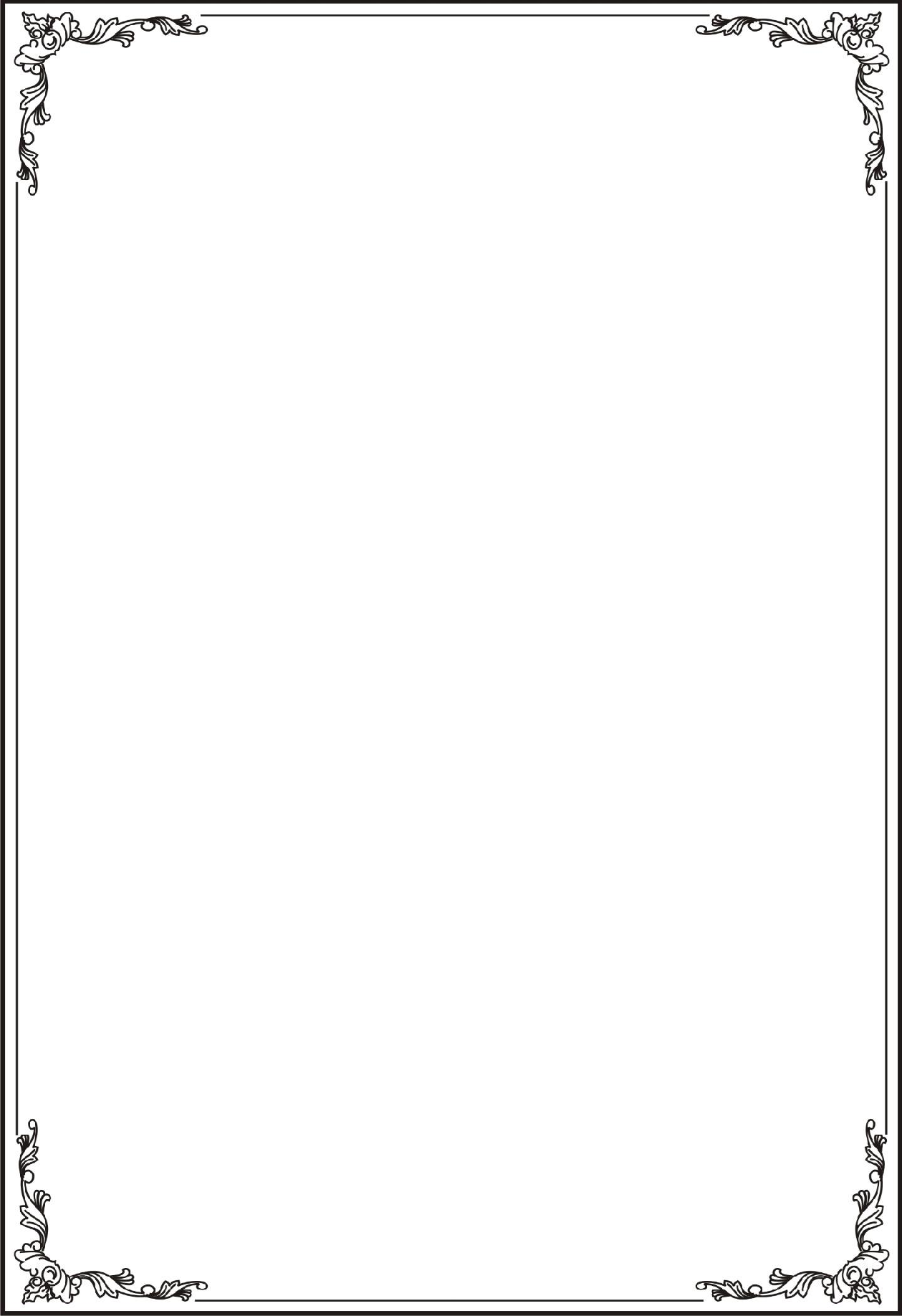
Sinh viên thực hiện: TRẦN HUỲNH HÒA PHÚC

Lớp: CQ.65.CNTT

Khóa: 65

Tp. Hồ Chí Minh, tháng 11 năm 2024

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI PHÂN HIỆU TẠI TP. HỒ CHÍ MINH**



**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO**

**MÔN: KỸ THUẬT LẬP TRÌNH**

# ĐỀ TÀI: BÀI TẬP LỚN

Giảng viên: ThS. TRẦN PHONG NHÃ

Sinh viên thực hiện: TRẦN HUỲNH HÒA PHÚC

Lớp: CQ.65.CNTT

Khóa: 65

Tp. Hồ Chí Minh, tháng 11 năm 2024

# LỜI CẢM ƠN

Lời nói đầu tiên, em xin gửi tới Quý Thầy Cô Bộ môn Công nghệ Thông tin Trường Đại học Giao thông vận tải phân hiệu tại thành phố Hồ Chí Minh lời chúc sức khỏe và lòng biết ơn sâu sắc.

Em xin chân thành cảm ơn quý thầy cô đã giúp đỡ tạo điều kiện để em hoàn thành báo cáo bài tập lớn . Đặc biệt em xin cảm ơn thầy Trần Phong Nhã đã nhiệt tình giúp đỡ, hướng dẫn cho em kiến thức, định hướng và kỹ năng để có thể hoàn thành bài báo cáo này này.

Tuy đã cố gắng trong quá trình nghiên cứu tìm hiểu tuy nhiên do kiến thức còn hạn chế nên vẫn còn tồn tại nhiều thiếu sót. Vì vậy em rất mong nhận được sự đóng góp ý kiến của Quý thầy cô bộ môn để đề tài của em có thể hoàn thiện hơn.

Lời sau cùng, em xin gửi lời chúc tới Quý Thầy Cô Bộ môn Công nghệ thông tin và hơn hết là thầy Trần Phong Nhã có thật nhiều sức khỏe, có nhiều thành công trong công việc. Em xin chân thành cảm ơn!

# NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

|  |  |
| --- | --- |
|  | **Giảng viên hướng dẫn**  **ThS.Trần Phong Nhã** |

## **BÀI TẬP LỚN**

## **A. LÝ THUYẾT**

## **1. Hàm**

**Khái niệm:**  
- Hàm là một khối mã có tên, thực hiện một chức năng cụ thể.  
- Giúp chia nhỏ chương trình thành các phần dễ quản lý, dễ bảo trì.  
- Có thể gọi đi gọi lại nhiều lần.  
- Có thể nhận đầu vào (tham số) và trả về giá trị (return).  
**Cấu trúc hàm:**  
<kiểu\_trả\_về> <tên\_hàm>(<danh\_sách\_tham\_số>) {  
 // thân hàm  
 return <giá\_trị>;  
}  
**Phân loại:**  
- Hàm có trả về (int, float, char,...)  
- Hàm không trả về (void)  
- Hàm không có tham số hoặc có nhiều tham số  
**Ưu điểm:**  
- Giảm lặp mã.  
- Tăng tính tái sử dụng.  
- Tăng tính rõ ràng và cấu trúc chương trình.

**Ví dụ:**

#include <stdio.h>

int tong(int a, int b) {

return a + b;

}

int giaithua(int n) {

if (n == 0) return 1;

return n \* giaithua(n - 1);

}

int laNguyenTo(int n) {

if (n < 2) return 0;

for (int i = 2; i <= n / 2; i++)

if (n % i == 0) return 0;

return 1;

}

int main() {

printf("Tong: %d\n", tong(8, 9));

int n;

printf("Nhap so nguyen n de tinh giai thua: ");

scanf("%d", &n);

if (n < 0)

printf("Khong tinh duoc giai thua so am!\n");

else

printf("Giai thua cua %d la: %d\n", n, giaithua(n));

printf("Nhap so nguyen n de kiem tra so nguyen to: ");

scanf("%d", &n);

if (laNguyenTo(n))

printf("%d la so nguyen to.\n", n);

else

printf("%d khong phai so nguyen to.\n", n);

return 0;

}

**2. Con trỏ**

Con trỏ là một biến dùng để lưu địa chỉ của biến khác trong bộ nhớ.

Dùng toán tử \* để truy cập giá trị mà con trỏ đang trỏ đến (giải tham chiếu).

Dùng toán tử & để lấy địa chỉ của một biến.

**Con trỏ giúp:**

* Truyền tham chiếu cho hàm (để hàm thay đổi giá trị gốc).
* Quản lý cấp phát bộ nhớ động.
* Làm việc với cấu trúc dữ liệu như mảng, danh sách,…

**Ví dụ:**

#include <stdio.h>

int main() {

//vd 1

int a = 10;

int \*p = &a;

printf("Gia tri cua a: %d, qua con tro: %d\n", a, \*p);

//vd 2

\*p = 20;

printf("Gia tri moi cua a sau khi thay doi qua con tro: %d\n", a);

//vd 3

int arr[] = {1, 2, 3, 4, 5};

int \*ptr = arr;

printf("Duyet mang qua con tro: ");

for (int i = 0; i < 5; i++) {

printf("%d ", \*(ptr + i));

}

printf("\n");

return 0;

}

**3. Con trỏ mảng**

Là con trỏ dùng để trỏ đến mảng (hoặc phần tử đầu tiên của mảng).

Trong C, tên mảng chính là con trỏ trỏ đến phần tử đầu.

Có thể truy cập phần tử qua con trỏ: \*(a + i) tương đương a[i].

**Thường dùng để:**

* Duyệt mảng linh hoạt.
* Tối ưu bộ nhớ và tốc độ xử lý.

**Ví dụ:**

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

int tong( int \*a, int n){

int s=0;

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

s+= \*(a+i);

}

return s;

}

void nhap(int \*a, int n){

printf("Nhap mang:\n");

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

printf("a[%d]= ",i+1);

scanf("%d",a+i);

}

}

void xuat(int \*a, int n){

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

printf("%d\t",\*(a+i));

}

}

int snt(int n){

for(int i=2; i<=sqrt(n); i++){

if(n%i == 0){

return 0;

}

}

return 1;

}

int sohoanhao(int n){

int s=0;

for(int i=1; i<n; i++){

if(n%i==0){

s+=i;

}

}

if(s==n) return 1;

else return 0;

}

void sapXep( int \*a, int n){

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

for(int j=0; j<n-i-1; j++){

if(\*(a+j)>\*(a+j+1)){

int tam= \*(a+j);

\*(a+j)= \*(a+j+1);

\*(a+j+1)= tam;

}

}

}

}

int demSnt(int \*a, int n){

int dem=0;

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

if(snt(\*(a+i))==1){

dem++;

}

}

return dem;

}

void insnt(int \*a, int n){

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

if(snt(\*(a+i))==1){

printf("%d\t",\*(a+i));

}

}

}

int demshh(int \*a, int n){

int dem=0;

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

if(sohoanhao(\*(a+i))==1){

dem++;

}

}

return dem;

}

void inshh(int \*a, int n){

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

if(sohoanhao(\*(a+i))==1){

printf("%d\t",\*(a+i));

}

}

}

int main(){

int n;

printf("Nhap n= ");

scanf("%d",&n);

int \*a =(int\*)malloc(n\*sizeof(int));

nhap(a,n);

printf("Tong day la: %d\n",tong(a,n));

printf("Mang da sap xep la: \n");

sapXep(a,n);

xuat(a,n);

if(demSnt(a,n)==0){

printf("\nMang khong ton tai so nguyen to.\n");

} else{

printf("So nguyen to co trong mang la: %d\n",demSnt(a,n));

insnt(a,n);

}

if(demshh(a,n)==0){

printf("\nMang khong ton tai so hoan hao.\n");

} else{

printf("So hoan hao co trong mang la: %d\n",demshh(a,n));

inshh(a,n);

}

free(a);

return 0;

}

**4. Mảng con trỏ**

Là một mảng mà mỗi phần tử là một con trỏ.

Thường gặp nhất là mảng con trỏ chuỗi: char \*ten[] = {"A", "B", "C"};

**Ưu điểm:**

Lưu danh sách các chuỗi có độ dài không cố định.

Linh hoạt hơn so với mảng 2 chiều ký tự.

**Ứng dụng**: menu lệnh, danh sách tên,…

**Ví dụ:**

#include <stdio.h>

int main() {

const char \*danhSachSinhVien[] = {

"Nguyen Van A",

"Tran Thi B",

"Le Van C",

"Pham Thi D",

"Hoang Van E"

};

int soLuong = sizeof(danhSachSinhVien) / sizeof(danhSachSinhVien[0]);

printf("Danh sach sinh vien:\n");

for (int i = 0; i < soLuong; i++) {

printf("%d. %s\n", i + 1, danhSachSinhVien[i]);

}

return 0;

}

**5. Con trỏ hàm**

Là con trỏ lưu địa chỉ của một hàm có kiểu cụ thể.

Cú pháp: int (\*ptr)(int, int); để trỏ đến một hàm nhận 2 int và trả về int.

Cho phép gọi hàm qua con trỏ: ptr(a, b);

**Ứng dụng:**

-Xây dựng bảng chọn động (menu).

-Truyền hàm như tham số (callback).

**Ví dụ:**

#include <stdio.h>

#define MAX 100

void inMang(int arr[], int n) {

printf("Mang: ");

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

printf("%d ", arr[i]);

printf("\n");

}

void tinhTong(int arr[], int n) {

int tong = 0;

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

tong += arr[i];

printf("Tong cac phan tu: %d\n", tong);

}

void timMax(int arr[], int n) {

int max = arr[0];

for (int i = 1; i < n; i++)

if (arr[i] > max)

max = arr[i];

printf("Gia tri lon nhat: %d\n", max);

}

void demChan(int arr[], int n) {

int dem = 0;

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

if (arr[i] % 2 == 0)

dem++;

printf("So luong phan tu chan: %d\n", dem);

}

typedef void (\*HamXuLy)(int[], int);

int main() {

int n, arr[MAX];

printf("Nhap so luong phan tu (toi da 100): ");

scanf("%d", &n);

printf("Nhap cac phan tu:\n");

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

printf("arr[%d] = ", i);

scanf("%d", &arr[i]);

}

HamXuLy dsHam[] = {inMang, tinhTong, timMax, demChan};

const char \*menu[] = {

"In mang",

"Tinh tong",

"Tim gia tri lon nhat",

"Dem so chan"

};

int chon;

do {

printf("\n----- MENU -----\n");

for (int i = 0; i < 4; i++)

printf("%d. %s\n", i + 1, menu[i]);

printf("0. Thoat\n");

printf("Lua chon: ");

scanf("%d", &chon);

if (chon >= 1 && chon <= 4) {

dsHam[chon - 1](arr, n);

} else if (chon != 0) {

printf("Lua chon khong hop le!\n");

}

} while (chon != 0);

return 0;

}

**6. Cấp phát động**

Cấp phát động là quá trình xin bộ nhớ trong lúc chương trình đang chạy (runtime).

Sử dụng các hàm trong <stdlib.h>:

* malloc() – cấp phát bộ nhớ chưa khởi tạo.
* calloc() – cấp phát và khởi tạo về 0.
* realloc() – thay đổi kích thước vùng nhớ.
* free() – giải phóng bộ nhớ.

**Ưu điểm:**

* Không cần xác định trước kích thước mảng.
* Quản lý bộ nhớ hiệu quả hơn.

**Ví dụ:**

#include <stdio.h>

#include <stdlib.h>

int main() {

int \*arr;

int n, i, tong = 0;

printf("Nhap so luong phan tu: ");

scanf("%d", &n);

arr = (int \*)malloc(n \* sizeof(int));

if (arr == NULL) {

printf("Khong the cap phat bo nho.\n");

return 1;

}

printf("Nhap %d phan tu:\n", n);

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

printf("arr[%d] = ", i);

scanf("%d", &arr[i]);

}

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

tong += arr[i];

}

printf("Tong cac phan tu la: %d\n", tong);

free(arr);

return 0;

}

**7. Xử lý tệp**

* Cho phép đọc/ghi dữ liệu ra/đọc vào từ file.
* **Gồm các bước:**
  1. Mở file (fopen)
  2. Đọc/ghi dữ liệu (fprintf, fscanf, fgets, fputs, v.v.)
  3. Đóng file (fclose)
* **Có hai chế độ:**
  + Văn bản: "r", "w", "a",…
  + Nhị phân: "rb", "wb",…
* **Ứng dụng:** lưu dữ liệu sinh viên, đọc cấu hình,…
* **Ví dụ:**

#include <stdio.h>

#include <stdlib.h>

typedef struct {

char ten[50];

float diem;

} SinhVien;

int main() {

FILE \*file;

int n;

printf("Nhap so luong sinh vien: ");

scanf("%d", &n);

SinhVien \*ds = (SinhVien \*)malloc(n \* sizeof(SinhVien));

if (ds == NULL) {

printf("Khong cap phat duoc bo nho!\n");

return 1;

}

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

printf("\nNhap thong tin sinh vien thu %d:\n", i + 1);

printf("Ten: ");

scanf(" %[^\n]", ds[i].ten);

printf("Diem: ");

scanf("%f", &ds[i].diem);

}

file = fopen("sinhvien.txt", "w");

if (file == NULL) {

printf("Khong mo duoc file de ghi!\n");

free(ds);

return 1;

}

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

fprintf(file, "%s %.2f\n", ds[i].ten, ds[i].diem);

}

fclose(file);

printf("\nDa ghi du lieu vao file 'sinhvien.txt'.\n");

file = fopen("sinhvien.txt", "r");

if (file == NULL) {

printf("Khong mo duoc file de doc!\n");

free(ds);

return 1;

}

printf("\nNoi dung file 'sinhvien.txt':\n");

char ten[50];

float diem;

while (fscanf(file, " %[^\n] %f", ten, &diem) == 2) {

printf("Ten: %s | Diem: %.2f\n", ten, diem);

}

fclose(file);

free(ds);

return 0;

}

**8. Kiểu cấu trúc**

Kiểu cấu trúc (struct) cho phép nhóm nhiều biến (thuộc tính) với các kiểu khác nhau thành một đơn vị.

* Có thể khai báo biến cấu trúc, mảng cấu trúc, con trỏ cấu trúc.
* Dễ tổ chức và quản lý dữ liệu phức tạp.

**Ví dụ:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct {

char maSV[10];

char hoTen[50];

float diem;

} SinhVien;

int main() {

int n;

printf("Nhap so luong sinh vien: ");

scanf("%d", &n);

SinhVien \*ds = (SinhVien \*)malloc(n \* sizeof(SinhVien));

if (ds == NULL) {

printf("Khong the cap phat bo nho!\n");

return 1;

}

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

printf("\nNhap thong tin sinh vien thu %d:\n", i + 1);

printf("Ma SV: ");

scanf("%s", ds[i].maSV);

printf("Ho ten: ");

scanf(" %[^\n]", ds[i].hoTen);

printf("Diem: ");

scanf("%f", &ds[i].diem);

}

printf("\nDanh sach sinh vien vua nhap:\n");

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

printf("MaSV: %s | Ho ten: %s | Diem: %.2f\n", ds[i].maSV, ds[i].hoTen, ds[i].diem);

}

printf("\nDanh sach sinh vien dau (diem >= 5):\n");

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

if (ds[i].diem >= 5.0) {

printf("MaSV: %s | Ho ten: %s | Diem: %.2f\n", ds[i].maSV, ds[i].hoTen, ds[i].diem);

}

}

free(ds);

return 0;

}

**9. Danh sách liên kết**

Là cấu trúc dữ liệu động gồm các phần tử (node) liên kết nhau bằng con trỏ.

**Mỗi node gồm:**

* Dữ liệu.
* Con trỏ trỏ tới node tiếp theo.

**Ưu điểm:**

* Kích thước linh hoạt.
* Dễ thêm/xoá phần tử ở bất kỳ vị trí nào.

**Nhược điểm:**

* Truy cập chậm hơn mảng.

**Các loại:**

* Danh sách liên kết đơn.
* Danh sách đôi.
* Danh sách vòng.

**Ví dụ:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct {

char maSV[10];

char hoTen[50];

float diem;

} SinhVien;

typedef struct Node {

SinhVien data;

struct Node \*next;

} Node;

Node\* taoNode(SinhVien sv) {

Node \*newNode = (Node \*)malloc(sizeof(Node));

newNode->data = sv;

newNode->next = NULL;

return newNode;

}

void themCuoi(Node \*\*head, SinhVien sv) {

Node \*newNode = taoNode(sv);

if (\*head == NULL) {

\*head = newNode;

} else {

Node \*temp = \*head;

while (temp->next != NULL)

temp = temp->next;

temp->next = newNode;

}

}

void inDanhSach(Node \*head) {

printf("\nDanh sach sinh vien:\n");

while (head != NULL) {

printf("MaSV: %s | Ho ten: %s | Diem: %.2f\n", head->data.maSV, head->data.hoTen, head->data.diem);

head = head->next;

}

}

void giaiPhong(Node \*head) {

Node \*temp;

while (head != NULL) {

temp = head;

head = head->next;

free(temp);

}

}

int main() {

Node \*danhSach = NULL;

int n;

printf("Nhap so luong sinh vien: ");

scanf("%d", &n);

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

SinhVien sv;

printf("\nNhap thong tin sinh vien thu %d:\n", i + 1);

printf("Ma SV: ");

scanf("%s", sv.maSV);

printf("Ho ten: ");

scanf(" %[^\n]", sv.hoTen);

printf("Diem: ");

scanf("%f", &sv.diem);

themCuoi(&danhSach, sv);

}

inDanhSach(danhSach);

giaiPhong(danhSach);

return 0;

}

**B. ỨNG DỤNG**

**Bài toán quản lý món ăn**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct {

char maMon[10];

char tenMon[50];

float gia;

} MonAn;

typedef struct Node {

MonAn data;

struct Node \*next;

} Node;

Node\* taoNode(MonAn m) {

Node \*newNode = (Node\*)malloc(sizeof(Node));

newNode->data = m;

newNode->next = NULL;

return newNode;

}

void themMonAn(Node \*\*head, MonAn m) {

Node \*newNode = taoNode(m);

if (\*head == NULL) {

\*head = newNode;

} else {

Node \*temp = \*head;

while (temp->next != NULL)

temp = temp->next;

temp->next = newNode;

}

}

void hienThiDanhSach(Node \*head) {

printf("\n--- DANH SÁCH MÓN AN ---\n");

while (head != NULL) {

printf("Ma: %s | Ten: %s | Gia: %.2f VND\n",

head->data.maMon, head->data.tenMon, head->data.gia);

head = head->next;

}

}

void timKiem(Node \*head, char tenCanTim[]) {

int found = 0;

while (head != NULL) {

if (strstr(head->data.tenMon, tenCanTim) != NULL) {

printf("Tim thay: %s - %s - %.2f\n",

head->data.maMon, head->data.tenMon, head->data.gia);

found = 1;

}

head = head->next;

}

if (!found) printf("Khong tim thay mon an nao!\n");

}

void xoaMonAn(Node \*\*head, char maXoa[]) {

Node \*temp = \*head, \*prev = NULL;

while (temp != NULL && strcmp(temp->data.maMon, maXoa) != 0) {

prev = temp;

temp = temp->next;

}

if (temp == NULL) {

printf("Khong tim thay mon an co ma %s\n", maXoa);

return;

}

if (prev == NULL) {

\*head = temp->next;

} else {

prev->next = temp->next;

}

free(temp);

printf("Da xoa mon an co ma %s\n", maXoa);

}

void ghiFile(Node \*head) {

FILE \*f = fopen("thucdon.txt", "w");

if (f == NULL) {

printf("Khong mo duoc file!\n");

return;

}

while (head != NULL) {

fprintf(f, "%s,%s,%.2f\n",

head->data.maMon, head->data.tenMon, head->data.gia);

head = head->next;

}

fclose(f);

printf("Da ghi file thanh cong.\n");

}

void giaiPhong(Node \*head) {

Node \*temp;

while (head != NULL) {

temp = head;

head = head->next;

free(temp);

}

}

int main() {

Node \*danhSach = NULL;

int chon;

do {

printf("\n===== MENU QUAN LY MON AN =====\n");

printf("1. Them mon an moi\n");

printf("2. Hien thi danh sach\n");

printf("3. Tim mon an theo ten\n");

printf("4. Xoa mon an theo ma\n");

printf("5. Ghi danh sach ra file\n");

printf("0. Thoat\n");

printf("Chon chuc nang: ");

scanf("%d", &chon);

getchar(); // Xóa ký t? '\n'

if (chon == 1) {

MonAn m;

printf("Nhap ma mon: "); fgets(m.maMon, 10, stdin); m.maMon[strcspn(m.maMon, "\n")] = '\0';

printf("Nhap ten mon: "); fgets(m.tenMon, 50, stdin); m.tenMon[strcspn(m.tenMon, "\n")] = '\0';

printf("Nhap gia: "); scanf("%f", &m.gia);

themMonAn(&danhSach, m);

} else if (chon == 2) {

hienThiDanhSach(danhSach);

} else if (chon == 3) {

char ten[50];

printf("Nhap ten mon can tim: ");

fgets(ten, 50, stdin); ten[strcspn(ten, "\n")] = '\0';

timKiem(danhSach, ten);

} else if (chon == 4) {

char ma[10];

printf("Nhap ma mon can xoa: ");

fgets(ma, 10, stdin); ma[strcspn(ma, "\n")] = '\0';

xoaMonAn(&danhSach, ma);

} else if (chon == 5) {

ghiFile(danhSach);

}

} while (chon != 0);

giaiPhong(danhSach);

return 0;