TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO

HỌC PHẦN: CẤU TRÚC DỮ LIỆU & GIẢI THUẬT

Bài tập danh sách liên kết

|  |  |
| --- | --- |
| Sinh viên thực hiện | MSSV |
| Huỳnh Phạm Hoàng Sơn | 3121411184 |

Giảng viên môn học: ĐỖ NHƯ TÀI

TP.HCM ngày 15 tháng 3 năm 2025

**Bài tập danh sách liên kết**

**Bài 1:**

## **Mục tiêu**

Cài đặt cấu trúc dữ liệu danh sách liên kết (Linked List) để lưu trữ danh sách số nguyên.

Thực hiện các thao tác trên danh sách liên kết:

Thêm phần tử.

Xóa phần tử.

Gộp hai danh sách.

Xuất danh sách.

Viết chương trình thực nghiệm để kiểm tra các thao tác trên danh sách liên kết.

## **Phân tích bài toán**

### **Cấu trúc danh sách liên kết**

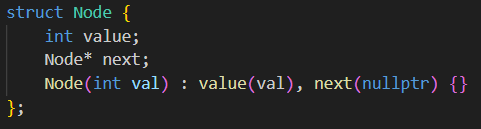
Danh sách liên kết được xây dựng bằng cách sử dụng **các nút (node)**, mỗi nút gồm:

**Giá trị (value)**: Chứa số nguyên.

**Con trỏ (next)**: Trỏ đến phần tử tiếp theo trong danh sách.

1. Cài đặt chi tiết
   1. Định nghĩa cấu trúc danh sách liên kết

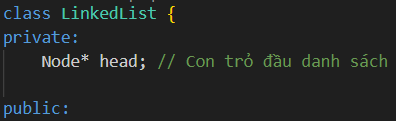
- Cấu trúc của 1 Node gồm:



Mỗi Node chứa một số nguyên (value) và một con trỏ next trỏ đến phần tử tiếp theo.

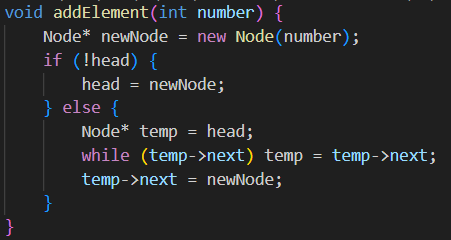
Khi khởi tạo, next = nullptr, vì chưa có phần tử nào tiếp theo.

- Lớp quản lý dánh sách liên kết



* 1. Cài đặt các thao tác trên danh sách

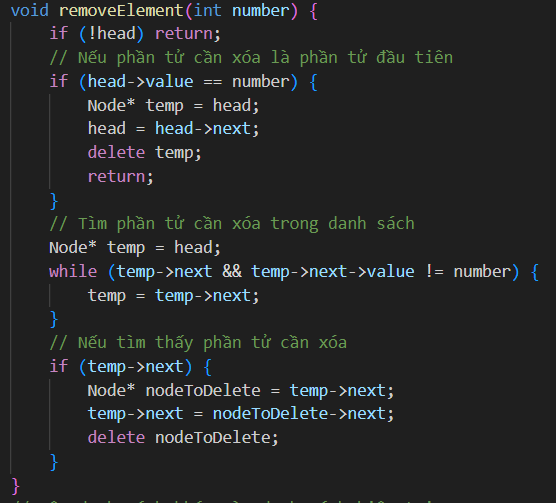
- thêm phần tử vào cuối danh sách



Nếu danh sách rỗng, head sẽ trỏ đến phần tử mới.

Nếu danh sách có phần tử, ta **duyệt đến cuối danh sách** rồi thêm phần tử mới vào.

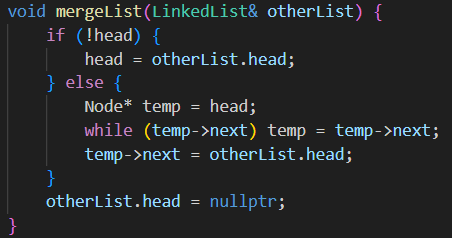
- Xóa một phần tử khỏi danh sách



Trường hợp phần tử cần xóa là phần tử đầu tiên, cập nhật head.

Nếu phần tử cần xóa nằm bên trong, ta **duyệt danh sách**, tìm phần tử đó rồi cập nhật liên kết.

- Gộp danh sách thứ hai vào danh sách thứ nhất

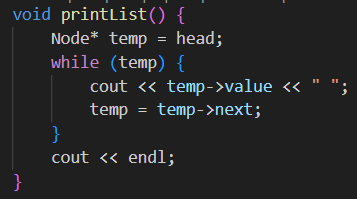


Nếu danh sách hiện tại rỗng, trỏ head đến danh sách thứ hai.

Nếu không, **tìm phần tử cuối** rồi gắn danh sách thứ hai vào.

Đặt otherList.head = nullptr để tránh trỏ trùng.

- Xuất danh sách



Duyệt từng phần tử trong danh sách và in ra màn hình.

* 1. Kiểm thử chương trình

Cho 1 dãy số nguyên gồm 10, sau đó chọn 1 số cần xóa lập tức in ra dãy số mới không còn số đã xóa trong dãy số. Sau đó nhập 5 số nguyên bất kì, sau khi nhập xong sẽ gộp 2 dãy số lại với nhau

**Câu 2:**

**Danh Sách Liên Kết Sinh Viên - Báo Cáo Chi Tiết**

## **1. Giới Thiệu Chung**

Code này sử dụng danh sách liên kết đơn để quản lý danh sách sinh viên, bao gồm các chức năng chính:

Nhập & Xuất danh sách sinh viên.

Xóa sinh viên theo tên hoặc địa chỉ.

Thêm sinh viên mới.

Sắp xếp danh sách bằng Selection Sort sử dụng con trỏ hàm so sánh.

## **2. Cấu trúc dữ liệu & Chức năng**

### **Cấu trúc Sinh Viên**

struct SinhVien {

char ten[50], diaChi[70], lop[10];

int khoa;

};

Lưu trữ họ tên, địa chỉ, lớp, khóa của sinh viên.

### **Cấu trúc Node danh sách liên kết**

struct Nut {

SinhVien data;

Nut\* next;

};

Mỗi node chứa 1 sinh viên và con trỏ tới node kế tiếp.

### **Danh sách sinh viên (DanhSach)**

#### Thêm sinh viên

void them(SinhVien sv) {

Nut\* moi = new Nut{sv, nullptr};

if (!head) head = moi;

else {

Nut\* temp = head;

while (temp->next) temp = temp->next;

temp->next = moi;

}

}

Thêm sinh viên vào cuối danh sách.

#### Xuất danh sách

void xuat() {

for (Nut\* temp = head; temp; temp = temp->next) temp->data.xuat();

}

Duyệt danh sách và in thông tin sinh viên.

#### Xóa sinh viên theo điều kiện

void xoa(bool (\*dieuKien)(SinhVien)) {

while (head && dieuKien(head->data)) {

Nut\* temp = head;

head = head->next;

delete temp;

}

for (Nut\* temp = head; temp && temp->next;) {

if (dieuKien(temp->next->data)) {

Nut\* xoaNut = temp->next;

temp->next = temp->next->next;

delete xoaNut;

} else temp = temp->next;

}

}

Dùng con trỏ hàm để xóa linh hoạt theo tên hoặc địa chỉ.

#### Sắp xếp danh sách

void sapXep(bool (\*soSanh)(SinhVien, SinhVien)) {

for (Nut\* i = head; i && i->next; i = i->next) {

Nut\* min = i;

for (Nut\* j = i->next; j; j = j->next) {

if (soSanh(j->data, min->data)) min = j;

}

swap(i->data, min->data);

}

}

Sắp xếp danh sách bằng Selection Sort.