CÀI CÁC TÁC VỤ DANH SÁCH LIÊN KẾT ĐƠN

**1. Main Idea:**

Chương trình cung cấp các chức năng cơ bản để thao tác với danh sách liên kết, gồm thêm, xóa, tìm kiếm, truy xuất, sắp xếp và giải phóng bộ nhớ.

**2. Output:**

Chương trình sẽ tương tác với người dùng thông qua giao diện dòng lệnh. Đầu ra bao gồm:

* **Danh sách các số nguyên:** Hiển thị các phần tử của danh sách liên kết sau khi nhập, sắp xếp, và xóa.
* **Kết quả tìm kiếm:** Thông báo liệu một phần tử có tồn tại trong danh sách hay không.
* **Giá trị tại một vị trí:** Hiển thị giá trị của phần tử tại một vị trí cụ thể trong danh sách.
* **Thông báo lỗi:** Hiển thị thông báo nếu vị trí truy xuất không hợp lệ.
* **Danh sách sau khi xóa:** Hiển thị danh sách sau khi xóa một phần tử.
* **Danh sách sau khi giải phóng:** Hiển thị danh sách rỗng sau khi giải phóng bộ nhớ.

**3. Giải thuật:**

* **Danh sách liên kết đơn:** Sử dụng cấu trúc Node để đại diện cho mỗi phần tử, với một trường info để lưu trữ dữ liệu và một trường next để trỏ đến phần tử tiếp theo.
* **Thêm phần tử (InsertFirst):** Tạo một node mới, trỏ next của node mới đến pHead hiện tại, và cập nhật pHead để trỏ đến node mới.
* **Tìm kiếm phần tử (Search):** Duyệt danh sách liên kết, so sánh info của mỗi node với giá trị cần tìm. Trả về con trỏ đến node nếu tìm thấy, ngược lại trả về NULL.
* **Xóa phần tử (Remove):** Duyệt danh sách liên kết, tìm node cần xóa. Cập nhật next của node trước đó để bỏ qua node cần xóa, và giải phóng bộ nhớ của node đó.
* **Truy xuất phần tử (Truy\_xuat):** Duyệt danh sách liên kết, đếm số lượng node đã duyệt. Trả về info của node tại vị trí được chỉ định, hoặc -1 nếu vị trí không hợp lệ.
* **Duyệt danh sách (Traverse):** Duyệt danh sách liên kết và in info của mỗi node.
* **Giải phóng bộ nhớ (ClearList):** Duyệt danh sách liên kết, giải phóng bộ nhớ của từng node.
* **Sắp xếp (SelectionSort):** Sử dụng thuật toán sắp xếp chọn (selection sort) để sắp xếp các phần tử trong danh sách liên kết theo thứ tự tăng dần.

**4. Chức năng của từng hàm:**

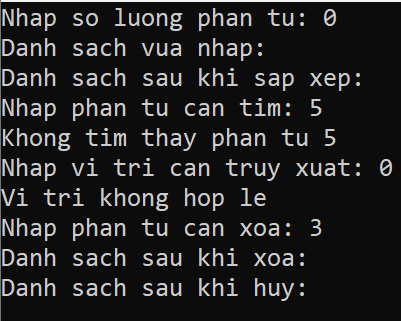
* **Node\* CreateNode(int x):** Tạo một node mới với giá trị x và trả về con trỏ đến node đó.
* **void Init(Node\*& pHead):** Khởi tạo danh sách liên kết rỗng bằng cách đặt pHead thành NULL.
* **bool isEmpty(Node\* pHead):** Kiểm tra xem danh sách liên kết có rỗng hay không. Trả về true nếu rỗng, false nếu không.
* **void InsertFirst(Node\*& pHead, int x):** Thêm một phần tử x vào đầu danh sách liên kết.
* **Node\* Search(Node\* pHead, int x):** Tìm kiếm phần tử x trong danh sách liên kết. Trả về con trỏ đến node nếu tìm thấy, NULL nếu không.
* **void Remove(Node\*& pHead, int x):** Xóa phần tử x khỏi danh sách liên kết.
* **int Truy\_xuat(Node\* pHead, int pos):** Lấy giá trị của phần tử tại vị trí pos trong danh sách liên kết.
* **void Traverse(Node\* pHead):** Duyệt và in các phần tử trong danh sách liên kết.
* **void ClearList(Node\*& pHead):** Giải phóng bộ nhớ của tất cả các node trong danh sách liên kết.
* **void SelectionSort(Node\*& pHead):** Sắp xếp các phần tử trong danh sách liên kết theo thứ tự tăng dần sử dụng thuật toán sắp xếp chọn.
* **int main():** Hàm chính, nơi thực hiện nhập dữ liệu, gọi các hàm để thao tác với danh sách liên kết, và hiển thị kết quả.

**Testcase 1: Danh sách rỗng**

**Input:** n 0

**Output:**

* Danh sach vua nhap: (rỗng)
* Danh sach sau khi sap xep: (rỗng)
* Nhap phan tu can tim: 5 Khong tim thay phan tu 5
* Nhap vi tri can truy xuat: 0 Vi tri khong hop le
* Nhap phan tu can xoa: 3 Danh sach sau khi xoa: (rỗng)
* Danh sach sau khi huy: (rỗng)

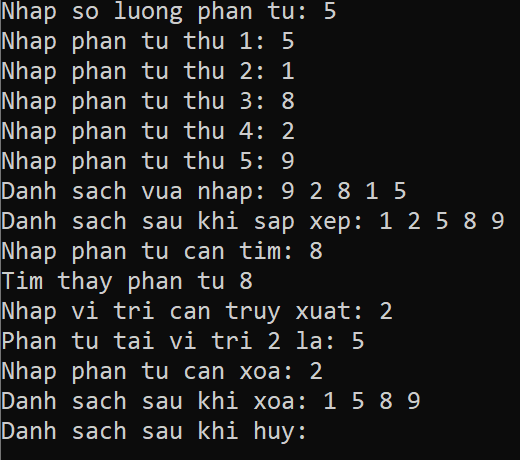


**Testcase 2: Danh sách với nhiều phần tử**

**Input:** n 5, 5 1 8 2 9

**Output:**

* Danh sach vua nhap: 9 2 8 1 5
* Danh sach sau khi sap xep: 1 2 5 8 9
* Nhap phan tu can tim: 8 Tim thay phan tu 8
* Nhap vi tri can truy xuat: 2 Phan tu tai vi tri 2 la: 5
* Nhap phan tu can xoa: 2 Danh sach sau khi xoa: 1 5 8 9
* Danh sach sau khi huy: (rỗng)

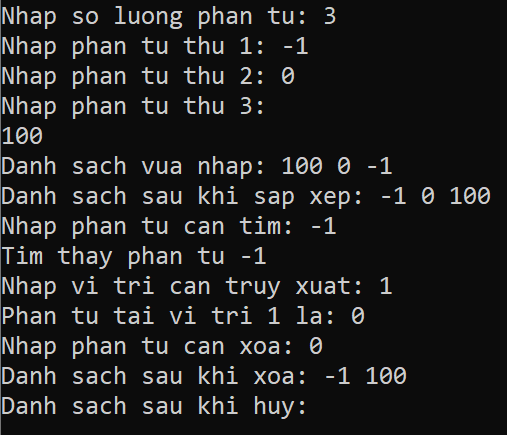


**Testcase 3: Thêm, tìm kiếm, xóa, truy xuất với các giá trị đặc biệt**

**Input:** n 3, 1 0 100

**Output:**

* Danh sach vua nhap: 100 0 1
* Danh sach sau khi sap xep: 1 0 100
* Nhap phan tu can tim: Tim thay phan tu 1
* Nhap vi tri can truy xuat: 1 Phan tu tai vi tri 1 la: 0
* Nhap phan tu can xoa: 0 Danh sach sau khi xoa: 1 100
* Danh sach sau khi huy: (rỗng)



**Testcase 4: Vị trí truy xuất không hợp lệ**

**Input:** n 3, 1 2 3

**Output:**

* + Danh sach vua nhap: 3 2 1
  + Nhap vi tri can truy xuat: 5 Vi tri khong hop le

