**Họ tên:** Nguyễn Thị Thảo Nguyên

**MSSV:** 3121411150

**Bài 1:**

* **Ý tưởng;**
* Cấu trúc dữ liệu:
* Sử dụng danh sách liên kết đơn để lưu trữ các số nguyên.
* Mỗi phần tử trong danh sách là một node gồm hai thành phần:
  + Giá trị (data): Lưu số nguyên.
  + Con trỏ (next): Trỏ đến phần tử tiếp theo.
* Cài đặt các thao tác chính:
* Thêm phần tử vào danh sách:
  + Tạo một node mới chứa giá trị cần thêm.
  + Nếu danh sách rỗng, node mới sẽ là phần tử đầu tiên.
  + Nếu danh sách không rỗng, duyệt đến cuối danh sách và liên kết node mới vào.
* Xóa phần tử có giá trị xác định:
  + Duyệt danh sách, kiểm tra từng phần tử.
  + Nếu phần tử có giá trị bằng kkk, cập nhật liên kết để bỏ qua node đó và giải phóng bộ nhớ.
* Gộp hai danh sách:
  + Duyệt đến cuối danh sách thứ nhất.
  + Trỏ node cuối cùng của danh sách thứ nhất đến node đầu tiên của danh sách thứ hai.
* Xuất danh sách:
  + Duyệt từng node và in ra giá trị.
* **Input:**

Nhập 10 số nguyên vào danh sách 1

Nhập một số k và xóa số k trong danh sách 1

Nhập 5 số nguyên vào danh sách thứ 2

* **Output:**

In danh sách thứ nhất sau khi nhập 10 số.

In danh sách thứ nhất sau khi xóa số nguyên cho trước.

In danh sách thứ nhất sau khi gộp thêm danh sách thứ hai.

* **Mã nguồn:**

#include <iostream>

using namespace std;

struct Node {

    int data;

    Node\* next;

};

Node\* createNode(int value) {

    return new Node{value, NULL};

}

void add(Node\*& head, int value) {

    Node\* newNode = createNode(value);

    if (!head) {

        head = newNode;

    } else {

        Node\* temp = head;

        while (temp->next) {

            temp = temp->next;

        }

        temp->next = newNode;

    }

}

void remove(Node\*& head, int value) {

    Node\* current = head;

    Node\* prev = NULL;

    while (current) {

        if (current->data == value) {

            if (prev) {

                prev->next = current->next;

            } else {

                head = current->next;

            }

            delete current;

            return;

        }

        prev = current;

        current = current->next;

    }

}

void addList(Node\*& head, Node\* other) {

    Node\* temp = other;

    while (temp) {

        add(head, temp->data);

        temp = temp->next;

    }

}

void display(Node\* head) {

    Node\* temp = head;

    while (temp) {

        cout << temp->data << " ";

        temp = temp->next;

    }

    cout << endl;

}

void deleteList(Node\*& head) {

    Node\* current = head;

    while (current) {

        Node\* temp = current;

        current = current->next;

        delete temp;

    }

    head = NULL;

}

int main() {

    Node\* list1 = NULL;

    Node\* list2 = NULL;

    int value;

    cout << "Nhập 10 số nguyên vào danh sách thứ nhất:" << endl;

    for (int i = 0; i < 10; i++) {

        cin >> value;

        add(list1, value);

    }

    cout << "Danh sách thứ nhất: ";

    display(list1);

    cout << "Nhập số nguyên cần xóa: ";

    cin >> value;

    remove(list1, value);

    cout << "Danh sách sau khi xóa: ";

    display(list1);

    cout << "Nhập 5 số nguyên vào danh sách thứ hai:" << endl;

    for (int i = 0; i < 5; i++) {

        cin >> value;

        add(list2, value);

    }

    addList(list1, list2);

    cout << "Danh sách thứ nhất sau khi thêm danh sách thứ hai: ";

    display(list1);

    deleteList(list1);

    deleteList(list2);

    return 0;

}

**Bài 2:**

* **Ý tưởng:**
* Sử dụng danh sách liên kết đơn để quản lý danh sách sinh viên.
* Mỗi sinh viên bao gồm: họ tên, địa chỉ, lớp, khóa và con trỏ next trỏ đến sinh viên tiếp theo.
* Các thao tác chính:
  1. Tạo sinh viên mới và thêm vào danh sách.
  2. Xuất danh sách sinh viên.
  3. Xóa sinh viên theo họ tên hoặc địa chỉ.
  4. Thêm sinh viên mới vào danh sách.
  5. Giải phóng bộ nhớ khi kết thúc chương trình.
* **Input:**

1. Nhập thông tin 10 sinh viên, mỗi sinh viên gồm:
   * Họ tên (chuỗi, tối đa 50 ký tự).
   * Địa chỉ (chuỗi, tối đa 70 ký tự).
   * Lớp (chuỗi, tối đa 10 ký tự).
   * Khóa (số nguyên).
2. Xóa sinh viên có họ tên là "Nguyen Van Teo"
3. Xóa sinh viên có địa chỉ là "Nguyen Van Cu"
4. Thêm sinh viên mới:
   * Họ tên: "Tran Thi Mo"
   * Địa chỉ: "25 Hong Bang"
   * Lớp: "TT0901"
   * Khóa: 2009

* **Output:**

1. Danh sách sinh viên ban đầu sau khi nhập.
2. Danh sách sau khi xóa sinh viên "Nguyen Van Teo" và "Nguyen Van Cu" (nếu có).
3. Danh sách sau khi thêm sinh viên "Tran Thi Mo".

* **Mã nguồn:**

#include <iostream>

#include <cstring>

using namespace std;

struct SinhVien {

    char hoTen[50];

    char diaChi[70];

    char lop[10];

    int khoa;

    SinhVien\* next;

};

SinhVien\* createSinhVien(const char\* hoTen, const char\* diaChi, const char\* lop, int khoa) {

    SinhVien\* sv = new SinhVien;

    for (int i = 0; hoTen[i] != '\0'; i++) sv->hoTen[i] = hoTen[i];

    sv->hoTen[strlen(hoTen)] = '\0';

    for (int i = 0; diaChi[i] != '\0'; i++) sv->diaChi[i] = diaChi[i];

    sv->diaChi[strlen(diaChi)] = '\0';

    for (int i = 0; lop[i] != '\0'; i++) sv->lop[i] = lop[i];

    sv->lop[strlen(lop)] = '\0';

    sv->khoa = khoa;

    sv->next = NULL;

    return sv;

}

void addSinhVien(SinhVien\*& head, SinhVien\* sv) {

    if (!head) {

        head = sv;

    } else {

        SinhVien\* temp = head;

        while (temp->next) {

            temp = temp->next;

        }

        temp->next = sv;

    }

}

void displayList(SinhVien\* head) {

    SinhVien\* temp = head;

    while (temp) {

        cout << "Họ Tên: " << temp->hoTen << ", Địa Chỉ: " << temp->diaChi

             << ", Lớp: " << temp->lop << ", Khóa: " << temp->khoa << endl;

        temp = temp->next;

    }

}

void removeSinhVien(SinhVien\*& head, const char\* key, bool byName) {

    SinhVien\* current = head;

    SinhVien\* prev = NULL;

    while (current) {

        if ((byName && strcmp(current->hoTen, key) == 0) ||

            (!byName && strcmp(current->diaChi, key) == 0)) {

            if (prev) {

                prev->next = current->next;

            } else {

                head = current->next;

            }

            delete current;

            return;

        }

        prev = current;

        current = current->next;

    }

}

void deleteList(SinhVien\*& head) {

    while (head) {

        SinhVien\* temp = head;

        head = head->next;

        delete temp;

    }

}

int main() {

    SinhVien\* list = NULL;

    cout << "Nhập 10 sinh viên:" << endl;

    for (int i = 0; i < 10; i++) {

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

        int khoa;

        cout << "Nhập họ tên: "; cin.getline(hoTen, 50);

        cout << "Nhập địa chỉ: "; cin.getline(diaChi, 70);

        cout << "Nhập lớp: "; cin.getline(lop, 10);

        cout << "Nhập khóa: "; cin >> khoa;

        cin.ignore();

        addSinhVien(list, createSinhVien(hoTen, diaChi, lop, khoa));

    }

    cout << "Danh sách sinh viên:" << endl;

    displayList(list);

    removeSinhVien(list, "Nguyen Van Teo", true);

    removeSinhVien(list, "Nguyen Van Cu", false);

    addSinhVien(list, createSinhVien("Tran Thi Mo", "25 Hong Bang", "TT0901", 2009));

    cout << "Danh sách sau khi cập nhật:" << endl;

    displayList(list);

    deleteList(list);

    return 0;

}