#include <iostream>

#include <string>

using namespace std;

// Определение структуры

struct Node {

string name;

int age;

double salary;

Node\* next;

};

// Функция для добавления элемента в начало списка

void addToBeginning(Node\*& head, string name, int age, double salary) {

Node\* newNode = new Node();

newNode->name = name;

newNode->age = age;

newNode->salary = salary;

newNode->next = head;

head = newNode;

}

// Функция для добавления элемента в конец списка

void addToEnd(Node\*& head, string name, int age, double salary) {

Node\* newNode = new Node();

newNode->name = name;

newNode->age = age;

newNode->salary = salary;

newNode->next = nullptr;

if (head == nullptr) {

head = newNode;

} else {

Node\* temp = head;

while (temp->next != nullptr) {

temp = temp->next;

}

temp->next = newNode;

}

}

// Функция для добавления элемента после заданного элемента

void addAfter(Node\*& head, string afterName, string name, int age, double salary) {

Node\* temp = head;

while (temp != nullptr && temp->name != afterName) {

temp = temp->next;

}

if (temp != nullptr) {

Node\* newNode = new Node();

newNode->name = name;

newNode->age = age;

newNode->salary = salary;

newNode->next = temp->next;

temp->next = newNode;

} else {

cout << "Элемент с именем " << afterName << " не найден." << endl;

}

}

// Функция для добавления элемента перед заданным элементом

void addBefore(Node\*& head, string beforeName, string name, int age, double salary) {

if (head == nullptr) {

cout << "Список пуст." << endl;

return;

}

if (head->name == beforeName) {

addToBeginning(head, name, age, salary);

return;

}

Node\* temp = head;

while (temp->next != nullptr && temp->next->name != beforeName) {

temp = temp->next;

}

if (temp->next != nullptr) {

Node\* newNode = new Node();

newNode->name = name;

newNode->age = age;

newNode->salary = salary;

newNode->next = temp->next;

temp->next = newNode;

} else {

cout << "Элемент с именем " << beforeName << " не найден." << endl;

}

}

// Функция для удаления элемента с заданным именем

void deleteByName(Node\*& head, string name) {

if (head == nullptr) {

cout << "Список пуст." << endl;

return;

}

if (head->name == name) {

Node\* temp = head;

head = head->next;

delete temp;

return;

}

Node\* temp = head;

while (temp->next != nullptr && temp->next->name != name) {

temp = temp->next;

}

if (temp->next != nullptr) {

Node\* nodeToDelete = temp->next;

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

delete nodeToDelete;

} else {

cout << "Элемент с именем " << name << " не найден." << endl;

}

}

// Функция для вывода списка на экран

void printList(Node\* head) {

Node\* temp = head;

while (temp != nullptr) {

cout << "Имя: " << temp->name << ", Возраст: " << temp->age << ", Зарплата: " << temp->salary << endl;

temp = temp->next;

}

}

// Основная функция

int main() {

Node\* head = nullptr;

// Добавление элементов в список

addToEnd(head, "Иван", 30, 50000);

addToBeginning(head, "Алексей", 25, 45000);

addToEnd(head, "Мария", 28, 55000);

addAfter(head, "Иван", "Ольга", 22, 48000);

addBefore(head, "Мария", "Петр", 35, 60000);

// Вывод списка

cout << "Список после добавления элементов:" << endl;

printList(head);

// Удаление элемента

deleteByName(head, "Иван");

cout << "Список после удаления элемента с именем Иван:" << endl;

printList(head);

return 0;

}