МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

##### «МИНСКИЙ ГОСУДАРСТВЕННЫЙ КОЛЛЕДЖ ЦИФРОВЫХ ТЕХНОЛОГИЙ»

##### ***Группа 82 ТП***

## **Отчет**

## **по лабораторной работе № 28**

# "Классы для работы с динамическими структурами данных"

учебный предмет

«Основы алгоритмизации и программирования»

**Исполнитель: Шатуха А.К.**

**Руководитель: Д.А.Федкевич**

Минск 2025

# Задание для самостоятельного решения

При решении задач необходимо описать класс, который используется для представления элементов динамической структуры данных. Затем разрабатывается класс для работы с используемой динамической структурой данных, которая при тестировании класса может быть построена путем ввода данных: a) с клавиатуры; б) из файла.

Возможны два варианта решения:

а) динамическая структура данных постоянно хранится в памяти;

б) динамическая структура данных хранится в файле.

1. Построить класс для работы с односвязным списком. Элементы списка – целые числа. Сформировать список, упорядочить элементы списка по возрастанию, используя сортировку: a) методом выбора; б) методом пузырька; в) методом вставки.

Листинг или ссылка на проект:

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

struct Node {

int data;

Node\* next;

Node(int val) : data(val), next(nullptr) {}

};

class LinkedList {

private:

Node\* head;

public:

LinkedList() : head(nullptr) {}

~LinkedList() {

Node\* current = head;

while (current != nullptr) {

Node\* temp = current;

current = current->next;

delete temp;

}

}

void append(int data) {

Node\* newNode = new Node(data);

if (!head) {

head = newNode;

return;

}

Node\* current = head;

while (current->next) {

current = current->next;

}

current->next = newNode;

}

void inputFromKeyboard() {

int n;

cout << "Введите количество элементов: ";

cin >> n;

cout << "Введите элементы:\n";

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

int data;

cin >> data;

append(data);

}

}

void inputFromFile(const string& filename) {

ifstream file(filename);

if (!file.is\_open()) {

cout << "Ошибка открытия файла!\n";

return;

}

int data;

while (file >> data) {

append(data);

}

file.close();

}

void display() {

Node\* current = head;

if (!current) {

cout << "Список пуст\n";

return;

}

while (current) {

cout << current->data << " ";

current = current->next;

}

cout << endl;

}

void selectionSort() {

if (!head || !head->next) return;

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

Node\* min = i;

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

if (j->data < min->data) {

min = j;

}

}

if (min != i) {

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

}

}

}

void bubbleSort() {

if (!head || !head->next) return;

bool swapped;

do {

swapped = false;

Node\* current = head;

while (current->next) {

if (current->data > current->next->data) {

swap(current->data, current->next->data);

swapped = true;

}

current = current->next;

}

} while (swapped);

}

void insertionSort() {

if (!head || !head->next) return;

Node\* sorted = nullptr;

Node\* current = head;

while (current) {

Node\* next = current->next;

if (!sorted || sorted->data >= current->data) {

current->next = sorted;

sorted = current;

}

else {

Node\* temp = sorted;

while (temp->next && temp->next->data < current->data) {

temp = temp->next;

}

current->next = temp->next;

temp->next = current;

}

current = next;

}

head = sorted;

}

};

int main() {

setlocale(LC\_ALL, "ru");

LinkedList list;

int choice;

cout << "1. Ввод с клавиатуры\n2. Ввод из файла\n";

cin >> choice;

if (choice == 1) {

list.inputFromKeyboard();

}

else {

string filename;

cout << "Введите имя файла: ";

cin >> filename;

list.inputFromFile(filename);

}

while (true) {

cout << "\nМеню:\n";

cout << "1. Показать список\n";

cout << "2. Сортировка выбором\n";

cout << "3. Сортировка пузырьком\n";

cout << "4. Сортировка вставками\n";

cout << "5. Выход\n";

cout << "Выберите действие: ";

cin >> choice;

switch (choice) {

case 1:

cout << "Список: ";

list.display();

break;

case 2:

list.selectionSort();

cout << "Отсортировано выбором\n";

break;

case 3:

list.bubbleSort();

cout << "Отсортировано пузырьком\n";

break;

case 4:

list.insertionSort();

cout << "Отсортировано вставками\n";

break;

case 5:

return 0;

default:

cout << "Неверный выбор\n";

}

}

return 0;

}

Скриншот:

