Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа 6

По дисциплине «Основы алгоритмизации и программирования»

На тему «Динамические структуры данных. Односвязные списки»

 Выполнила:

Студентка1 курса 6 группы

Альшевская Алина Михайловна

Преподаватель: асс. Андронова М.В.

2023, Минск

Вариант 1

В соответствии со своим вариантом разработать программу с использованием ***односвязного списка*** по данным, представленным в таблице ниже.

Программа должна содержать меню с пунктами: добавление элемента, удаление элемента, поиск элемента, вывод списка в консольное окно, запись списка в файл, считывание списка из файла.



#include <iostream>

#include <fstream>

struct Node {//cтруктура данных

int data;

Node\* next;

};

Node\* head = nullptr;

void addNode(int data) {//функция добавления данных

Node\* newNode = new Node();

newNode->data = data;

newNode->next = head;

head = newNode;

}

void deleteNode(int data) {//функция удаления данных

if (head == nullptr) return;

if (head->data == data) {

Node\* temp = head;

head = head->next;

delete temp;

return;

}

Node\* current = head;

while (current->next != nullptr && current->next->data != data) {

current = current->next;

}

if (current->next == nullptr) return;

Node\* temp = current->next;

current->next = current->next->next;

delete temp;

}

Node\* searchNode(int data) {//функция для поиска

Node\* current = head;

while (current != nullptr) {

if (current->data == data) return current;

current = current->next;

}

return nullptr;

}

void printList() {//фугкция для вывода

Node\* current = head;

while (current != nullptr) {

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

current = current->next;

}

std::cout << std::endl;

}

void writeToFile(const std::string& filename) {//запись в файл

std::ofstream file(filename);

Node\* current = head;

while (current != nullptr) {

file << current->data << " ";

current = current->next;

}

file.close();

}

void readFromFile(const std::string& filename) {//чтения ез файла

std::ifstream file(filename);

int data;

while (file >> data) {

addNode(data);

}

file.close();

}

int sumPositive() {//функция суммирования положительных данных

int sum = 0;

Node\* current = head;

while (current != nullptr) {

if (current->data > 0) sum += current->data;

current = current->next;

}

return sum;

}

int main() {

setlocale(LC\_ALL, "Russian");

int choice;

do {

std::cout << "1-Добавить элемент" << std::endl;

std::cout << "2-Удалить элемент" << std::endl;

std::cout << "3-Найти элемент" << std::endl;

std::cout << "4-Вывести список" << std::endl;

std::cout << "5-Записать список в файл" << std::endl;

std::cout << "6-Считать список из файла" << std::endl;

std::cout << "7-Сумма положительных элементов" << std::endl;

std::cout << "8-Выход" << std::endl;

std::cin >> choice;

switch (choice) {//используем switch для реализации меню

case 1: {

int data;

std::cout << "Введите данные: ";

std::cin >> data;

addNode(data);

break;

}

case 2: {

int data;

std::cout << "Введите данные: ";

std::cin >> data;

deleteNode(data);

break;

}

case 3: {

int data;

std::cout << "Введите данные: ";

std::cin >> data;

Node\* node = searchNode(data);

if (node != nullptr) {

std::cout << "Элемент найден: " << node->data << std::endl;

}

else {

std::cout << "Элемент не найден" << std::endl;

}

break;

}

case 4: {

printList();

break;

}

case 5: {

std::string filename;

std::cout << "Введите имя файла: ";

std::cin >> filename;

writeToFile(filename);

break;

}

case 6: {

std::string filename;

std::cout << "Введите имя файла: ";

std::cin >> filename;

readFromFile(filename);

break;

}

case 7: {

int sum = sumPositive();

if (sum > 0) {

std::cout << "Сумма положительных элементов: " << sum << std::endl;

}

else {

std::cout << "Положительных элементов нет" << std::endl;

}

break;

}

case 8: {

std::cout << "Выход из программы..." << std::endl;

break;

}

default: {

std::cout << "Неверный ввод! Попробуйте еще раз." << std::endl;

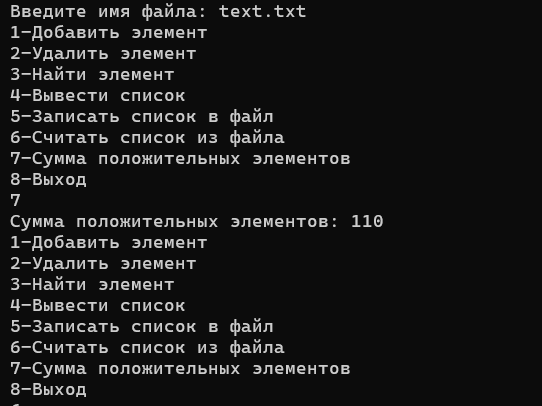
break;

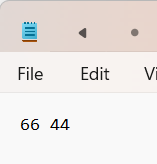
}

}

} while (choice != 8);

return 0;}





Дополнительные варианты

Вариант 15



#include <iostream>

#include <fstream>

#include <Windows.h>

using namespace std;

//односвязный список

struct list {

int number;

list\* next;

};

//прототипы функций

void insertData(list\*&, int);

void printList(list\*);

void getSum(list\*);

int deleteData(list\*&, int);

int searchData(list\*, int);

int isEmpty(list\*);

void toFile(list\*);

void fromFile(list\*&);

// главная функция

int main() {

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

list\* first = nullptr; //указатель на начало односвязного списка

int choice, value;

do {

cout << "Выберите вариант работы приложения:" << endl;

cout << " 1 - Ввод числа" << endl;

cout << " 2 - Вывод списка в консольное окно" << endl;

cout << " 3 - Поиск суммы положительных чисел, оканчивающихся на 7" << endl;

cout << " 4 - Удаление числа" << endl;

cout << " 5 - Поиск числа" << endl;

cout << " 6 - Запись списка в файл" << endl;

cout << " 7 - Считывание списка из файла" << endl;

cout << " 0 - Выход" << endl;

cout << "Ваш выбор: ";

cin >> choice;

switch (choice) {

case 1: { //ввод числа

cout << "Введите число: ";

cin >> value;

insertData(first, value);

break;

}

case 2: { //вывести элементы списка

if (!isEmpty(first)) {

printList(first);

}

else {

cout << "Список пуст" << endl;

}

break;

}

case 3: { //посчитать средние значения положительных чисел списка

if (!isEmpty(first)) {

getSum(first);

}

else {

cout << "Список пуст" << endl;

}

break;

}

case 4: { // удалить число из списка

if (!isEmpty(first)) {

cout << "Введите удаляемое число: ";

cin >> value;

if (deleteData(first, value)) {

cout << "Удалено число " << value << endl;

printList(first);

}

else {

cout << "Число не найдено" << endl;

}

}

else {

cout << "Список пуст" << endl;

}

break;

}

case 5: { //поиск числа в спииске

int search;

cout << "Введите число для поиска: ";

cin >> search;

if (searchData(first, search)) {

cout << "Введенное число есть в списке." << endl;

}

else {

cout << "Введенного числа нет в списке." << endl;

}

break;

}

case 6: { //запись списка в файл

toFile(first);

break;

}

case 7: { //чтение списка из файла

fromFile(first);

break;

}

case 0: { //выход из программы

cout << "Конец программы" << endl;

break;

}

default: { //в случае неверного ввода

cout << "Неправильный выбор" << endl;

break;

}

}

} while (choice != 0);

//очищение памяти, занятой списком

list\* temp;

while (first != nullptr) {

temp = first;

first = first->next;

delete(temp);

}

return 0;

}

// добавление числа value в список

void insertData(list\*& p, int value) {

list\* newP = new list; //выделение места для ячейки списка и создание указателя на нее

if (newP != nullptr) {

newP->number = value;

newP->next = p;

p = newP;

}

else {

cout << "Операция добавления не выполнена" << endl;

}

}

// вывод списка

void printList(list\* p) {

if (p == nullptr) { //если дошли до начала списка

cout << "Список пуст" << endl;

}

else {

cout << "Список:" << endl;

while (p != nullptr) { //пока не дошли до начала списка,

cout << p->number << " "; //выводим элементы

p = p->next; //переходим к новому элементу по адресу

}

cout << "NULL" << endl;

}

}

// поиск суммы положительных чисел, оканчивающихся на 7

void getSum(list\* p) {

int sum = 0, counter\_pos = 0;

while (p != nullptr) {

if (p->number > 0 && p->number % 10 == 7) {

sum += p->number;

counter\_pos++;

}

p = p->next;

}

if (counter\_pos == 0) {

cout << "В списке нет положительных чисел, оканчивающихся на 7." << endl;

}

else {

cout << "Сумма положительных чисел, оканчивающихся на 7: " << sum << endl;

}

}

// Удаление числа

int deleteData(list\*& p, int value) {

list\* previous = nullptr, \* current = p, \* temp;

while (current != nullptr && current->number != value) { //ищем нужный элемент, пока не найдем или не доберемся до нуль-элемента

previous = current;

current = current->next;

}

if (current != nullptr) {

temp = current;

if (previous != nullptr)

previous->next = current->next;

else

p = current->next;

delete(temp);

return value;

}

return 0;

}

//Список пустой? (1-да, 0-нет)

int isEmpty(list\* p) {

return p == nullptr;

}

// поиск числа в списке

int searchData(list\* p, int value) {

while (p != nullptr) {

if (p->number == value) {

return 1;

}

p = p->next;

}

return 0;

}

//запись чисел в файл

void toFile(list\* p) {

ofstream ofile("numbers.txt"); //создание файла для записи

if (ofile.fail()) {

cout << "Ошибка открытия файла" << endl;

exit(1);

}

if (!isEmpty(p)) { //проверка, не пустой ли список

while (p != nullptr) {

ofile << p->number << " "; //запись элементов списка в файл

p = p->next;

}

}

else {

cout << "Список пуст" << endl;

}

ofile.close();

cout << "Список записан в файл" << endl;

}

//чтение списка из файла

void fromFile(list\*& p) {

int temp;

ifstream ifile("numbers.txt"); //создание файла для чтения

if (ifile.fail()) {

cout << "Ошибка открытия файла" << endl;

exit(1);

}

while (ifile >> temp) { //чтение списка из файла

insertData(p, temp); //создание списка

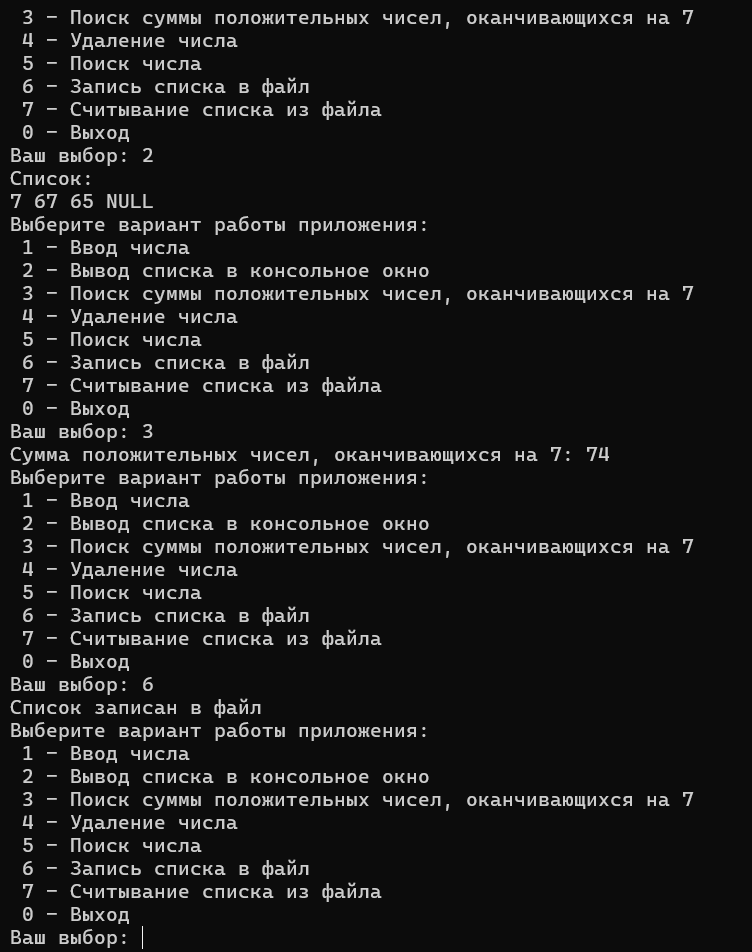
}

ifile.close();

cout << "Список прочитан из файла" << endl;

printList(p); //вывод полученного списка

}



Вариант 12

#include <iostream>

#include <fstream>

struct Node {//cтруктура данных

float data;

Node\* next;

};

Node\* head = nullptr;

void addNode(int data) {//функция добавления данных

Node\* newNode = new Node();

newNode->data = data;

newNode->next = head;

head = newNode;

}

void deleteNode(int data) {//функция удаления данных

if (head == nullptr) return;

if (head->data == data) {

Node\* temp = head;

head = head->next;

delete temp;

return;

}

Node\* current = head;

while (current->next != nullptr && current->next->data != data) {

current = current->next;

}

if (current->next == nullptr) return;

Node\* temp = current->next;

current->next = current->next->next;

delete temp;

}

Node\* searchNode(int data) {//функция для поиска

Node\* current = head;

while (current != nullptr) {

if (current->data == data) return current;

current = current->next;

}

return nullptr;

}

void printList() {//фугкция для вывода

Node\* current = head;

while (current != nullptr) {

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

current = current->next;

}

std::cout << std::endl;

}

void writeToFile(const std::string& filename) {//запись в файл

std::ofstream file(filename);

Node\* current = head;

while (current != nullptr) {

file << current->data << " ";

current = current->next;

}

file.close();

}

void readFromFile(const std::string& filename) {//чтения ез файла

std::ifstream file(filename);

int data;

while (file >> data) {

addNode(data);

}

file.close();

}

int sumPositive() {//функция суммирования положительных данных

int sum = 0;

Node\* current = head;

while (current != nullptr) {

if (current->data > 0) sum += current->data;

current = current->next;

}

return sum;

}

int main() {

setlocale(LC\_ALL, "Russian");

int choice;

do {

std::cout << "1-Добавить элемент" << std::endl;

std::cout << "2-Удалить элемент" << std::endl;

std::cout << "3-Найти элемент" << std::endl;

std::cout << "4-Вывести список" << std::endl;

std::cout << "5-Записать список в файл" << std::endl;

std::cout << "6-Считать список из файла" << std::endl;

std::cout << "7-Сумма положительных элементов" << std::endl;

std::cout << "8-Выход" << std::endl;

std::cin >> choice;

switch (choice) {//используем switch для реализации меню

case 1: {

int data;

std::cout << "Введите данные: ";

std::cin >> data;

addNode(data);

break;

}

case 2: {

int data;

std::cout << "Введите данные: ";

std::cin >> data;

deleteNode(data);

break;

}

case 3: {

int data;

std::cout << "Введите данные: ";

std::cin >> data;

Node\* node = searchNode(data);

if (node != nullptr) {

std::cout << "Элемент найден: " << node->data << std::endl;

}

else {

std::cout << "Элемент не найден" << std::endl;

}

break;

}

case 4: {

printList();

break;

}

case 5: {

std::string filename;

std::cout << "Введите имя файла: ";

std::cin >> filename;

writeToFile(filename);

break;

}

case 6: {

std::string filename;

std::cout << "Введите имя файла: ";

std::cin >> filename;

readFromFile(filename);

break;

}

case 7: {

int sum = sumPositive();

if (sum > 0) {

std::cout << "Сумма положительных элементов: " << sum << std::endl;

}

else {

std::cout << "Положительных элементов нет" << std::endl;

}

break;

}

case 8: {

std::cout << "Выход из программы..." << std::endl;

break;

}

default: {

std::cout << "Неверный ввод! Попробуйте еще раз." << std::endl;

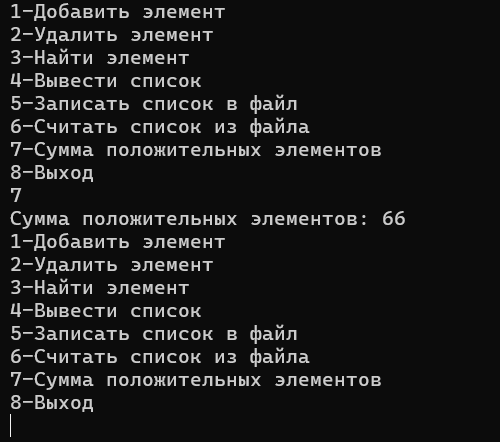
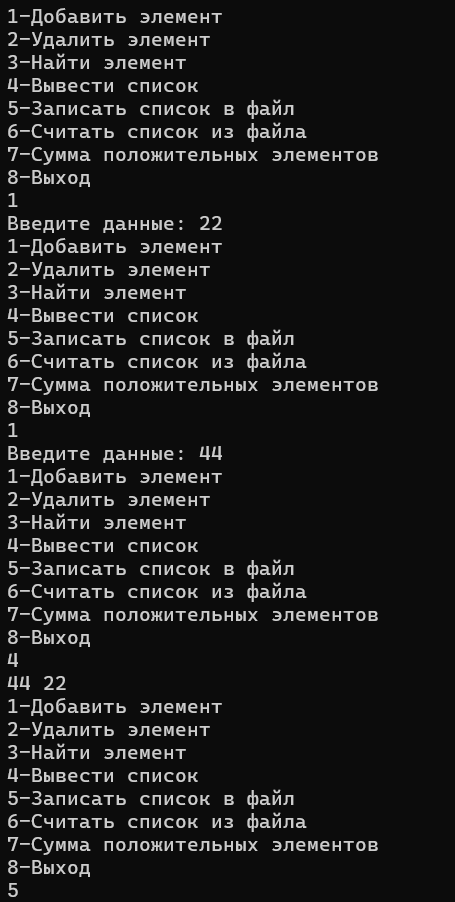
break;

}

}

} while (choice != 8);

return 0;

}

Вариант 13



#include <iostream>

#include <fstream>

#include <Windows.h>

using namespace std;

//односвязный список

struct list {

int number;

list\* next;

};

//прототипы функций

void insertData(list\*&, int);

void printList(list\*);

void getSum(list\*);

int deleteData(list\*&, int);

int searchData(list\*, int);

int isEmpty(list\*);

void toFile(list\*);

void fromFile(list\*&);

// главная функция

int main() {

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

list\* first = nullptr; //указатель на начало односвязного списка

int choice, value;

do {

cout << "Выберите вариант работы приложения:" << endl;

cout << " 1 - Ввод числа" << endl;

cout << " 2 - Вывод списка в консольное окно" << endl;

cout << " 3 - Поиск суммы отрицательных двузначных чисел" << endl;

cout << " 4 - Удаление числа" << endl;

cout << " 5 - Поиск числа" << endl;

cout << " 6 - Запись списка в файл" << endl;

cout << " 7 - Считывание списка из файла" << endl;

cout << " 0 - Выход" << endl;

cout << "Ваш выбор: ";

cin >> choice;

switch (choice) {

case 1: { //ввод числа

cout << "Введите число: ";

cin >> value;

insertData(first, value);

break;

}

case 2: { //вывести элементы списка

if (!isEmpty(first)) {

printList(first);

}

else {

cout << "Список пуст" << endl;

}

break;

}

case 3: { //посчитать средние значения положительных чисел списка

if (!isEmpty(first)) {

getSum(first);

}

else {

cout << "Список пуст" << endl;

}

break;

}

case 4: { // удалить число из списка

if (!isEmpty(first)) {

cout << "Введите удаляемое число: ";

cin >> value;

if (deleteData(first, value)) {

cout << "Удалено число " << value << endl;

printList(first);

}

else {

cout << "Число не найдено" << endl;

}

}

else {

cout << "Список пуст" << endl;

}

break;

}

case 5: { //поиск числа в спииске

int search;

cout << "Введите число для поиска: ";

cin >> search;

if (searchData(first, search)) {

cout << "Введенное число есть в списке." << endl;

}

else {

cout << "Введенного числа нет в списке." << endl;

}

break;

}

case 6: { //запись списка в файл

toFile(first);

break;

}

case 7: { //чтение списка из файла

fromFile(first);

break;

}

case 0: { //выход из программы

cout << "Конец программы" << endl;

break;

}

default: { //в случае неверного ввода

cout << "Неправильный выбор" << endl;

break;

}

}

} while (choice != 0);

//очищение памяти, занятой списком

list\* temp;

while (first != nullptr) {

temp = first;

first = first->next;

delete(temp);

}

return 0;

}

// добавление числа value в список

void insertData(list\*& p, int value) {

list\* newP = new list; //выделение места для ячейки списка и создание указателя на нее

if (newP != nullptr) {

newP->number = value;

newP->next = p;

p = newP;

}

else {

cout << "Операция добавления не выполнена" << endl;

}

}

// вывод списка

void printList(list\* p) {

if (p == nullptr) { //если дошли до начала списка

cout << "Список пуст" << endl;

}

else {

cout << "Список:" << endl;

while (p != nullptr) { //пока не дошли до начала списка,

cout << p->number << " "; //выводим элементы

p = p->next; //переходим к новому элементу по адресу

}

cout << "NULL" << endl;

}

}

// поиск суммы отрицательных двузначных чисел

void getSum(list\* p) {

int sum = 0, counter\_neg = 0;

while (p != nullptr) {

if (p->number > -100 && p->number < -9) {

sum += p->number;

counter\_neg++;

}

p = p->next;

}

if (counter\_neg == 0) {

cout << "В списке нет отрицательных двузначных чисел" << endl;

}

else {

cout << "Сумма отрциательных двузначных чисел: " << sum << endl;

}

}

// Удаление числа

int deleteData(list\*& p, int value) {

list\* previous = nullptr, \* current = p, \* temp;

while (current != nullptr && current->number != value) { //ищем нужный элемент, пока не найдем или не доберемся до нуль-элемента

previous = current;

current = current->next;

}

if (current != nullptr) {

temp = current;

if (previous != nullptr)

previous->next = current->next;

else

p = current->next;

delete(temp);

return value;

}

return 0;

}

//Список пустой? (1-да, 0-нет)

int isEmpty(list\* p) {

return p == nullptr;

}

// поиск числа в списке

int searchData(list\* p, int value) {

while (p != nullptr) {

if (p->number == value) {

return 1;

}

p = p->next;

}

return 0;

}

//запись чисел в файл

void toFile(list\* p) {

ofstream ofile("numbers.txt"); //создание файла для записи

if (ofile.fail()) {

cout << "Ошибка открытия файла" << endl;

exit(1);

}

if (!isEmpty(p)) { //проверка, не пустой ли список

while (p != nullptr) {

ofile << p->number << " "; //запись элементов списка в файл

p = p->next;

}

}

else {

cout << "Список пуст" << endl;

}

ofile.close();

cout << "Список записан в файл" << endl;

}

//чтение списка из файла

void fromFile(list\*& p) {

int temp;

ifstream ifile("numbers.txt"); //создание файла для чтения

if (ifile.fail()) {

cout << "Ошибка открытия файла" << endl;

exit(1);

}

while (ifile >> temp) { //чтение списка из файла

insertData(p, temp); //создание списка

}

ifile.close();

cout << "Список прочитан из файла" << endl;

printList(p); //вывод полученного списка

}

