Учреждение образования

«БЕЛОРУССКИЙГОСУДАРСТВЕННЫЙТЕХНОЛОГИЧЕСКИЙУНИВЕРСИТЕТ»

Н. Н. Пустовалова, Н. В. Пацей

# ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

**в 2-х частях**

**Часть 2**

**Лабораторный практикум**

**Лабораторная работа № 9. Двусвязные списки**

**Лабораторная работа № 9. Двусвязные списки**

|  |  |
| --- | --- |
| **Задание** | **Программа** |
| 1. В правой части записана программа, которая формирует *двусвязный список* для хранения информации, содержащей адреса людей (имя и город).  Написать комментарии к программе. |  |
| 2. В правой части приведены функции, осуществляющие запись информации в файл и считывание из файла.  Добавить эти функции в программу п.1 и внести соответствующие изменения в меню. |  |
| 3. Дополнить программу функцией в соответствии со своим вариантом из таблицы, представленной ниже. |  |
| 3.1. В [приложении 2](#prilogenie2) приведен проект, в котором реализован проект с использованием *двусвязного* *списка*. На основе данного проекта разработать функции, которые предлагается создать в данном приложении.  В проекте используется двусвязный список для хранения информации о телефонных номерах. Проект состоит из трех частей: главная функция с именем **OAP\_List**, файл **List.cpp** с функциями пользователя, заголовочный файл **List.h**.  1. Создать проект, ввести операторы *главной* функции и изучить структуру функции.  Добавить операторы создания списков и операций над ними.  2. Добавить в проект файл **List.cpp**. Изучить функции программы.  Написать комментарии к операторам.  3. Добавить в проект заголовочный файл **List.h**.  Выполнить программы проекта.  4. Дополнить проект, разработав функцию **deleteList** (удаление списка) и функцию **countList** (подсчет числа элементов списка). |  |
| 2. Каждый элемент списка студентов содержит фамилию, имя, отчество, год рождения, курс, номер группы, оценки по пяти предметам. Упорядочить студентов по курсу, причем студенты одного курса должны располагаться в алфавитном порядке. Найти средний балл каждой группы по каждому предмету. Определить самого старшего студента и самого младшего. Для каждой группы найти лучшего с точки зрения успеваемости студента. |  |

**Main.cpp:**

#include <iostream>

#include <windows.h>

#include "function.h"

using namespace std;

void showMenu()

{

printLine();

cout << "Меню\n";

cout << "0 - Выход из программы\n";

cout << "1 - Показать меню\n";

cout << "2 - Записать нового студента в список\n";

cout << "3 - Найти средний бал определенной группы по определенному предемету\n";

cout << "4 - Определить самого старшего студента\n";

cout << "5 - Определить самого младшего сиудента\n";

cout << "6 - Найти лучшего студента с точки зрения успеваемости\n";

cout << "7 - Вывести отсортированный список всех студентов\n";

cout << "8 - Считать список из файла\n";

printLine();

}

int main()

{

int userInput = 1;

setlocale(LC\_CTYPE, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

showMenu();

while (userInput != 0)

{

cin >> userInput;

switch (userInput)

{

case 0:

break;

case 1:

showMenu();

break;

case 2:

addStudent();

break;

case 3:

calculateAverageScore();

break;

case 4:

findOldest();

break;

case 5:

findYoungest();

break;

case 6:

findTheBestStudent();

break;

case 7:

showList();

break;

case 8:

readList();

break;

default:

cout << "Ошибка ввода\n";

break;

}

cout << "Что дальше?\n";

}

}

**Function.cpp:**

#include <iostream>

#include "function.h"

#include <fstream>

using namespace std;

Student\* firstStudent = nullptr, \* lastStudent = nullptr;

void printLine()

{

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

void addStudent()

{

Student\* temp = new Student, \* ptr = firstStudent;

printLine();

cout << "Введите данные студента:\n";

cout << "Фамилия: ";

cin.ignore();

cin.getline(temp->secondname, 20);

cout << "Имя: ";

cin.getline(temp->name, 20);

cout << "Отчество: ";

cin.getline(temp->surname, 20);

cout << "Год рождения: ";

cin >> temp->yearOfBirth;

cout << "На каком курсе учится студент: ";

cin >> temp->course;

cout << "Группа: ";

cin >> temp->group;

cout << "Оценка за экзамен по ОАИП: ";

cin >> temp->oaipGrade;

cout << "Оценка за экзамен по ОПИ: ";

cin >> temp->opiGrade;

cout << "Оценка за экзамен по английскому: ";

cin >> temp->englishGrade;

cout << "Оценка за экзамен по математитке: ";

cin >> temp->mathemathicsGrade;

cout << "Оценка за экзамен по физике: ";

cin >> temp->physicsGrade;

printLine();

temp->next = nullptr;

temp->previous = nullptr;

if (firstStudent == nullptr)

{

firstStudent = lastStudent = temp;

}

else

{

while (ptr->next != nullptr && ptr->course < temp->course)

{

ptr = ptr->next;

}

if (ptr->next == nullptr)

{

temp->previous = ptr;

ptr->next = temp;

lastStudent = lastStudent->next;

}

else

{

while (ptr->next != nullptr && ptr->secondname[0] < temp->secondname[0])

{

ptr = ptr->next;

}

if (ptr->next == nullptr)

{

temp->previous = ptr;

ptr->next = temp;

lastStudent = lastStudent->next;

}

else if (ptr->previous == nullptr)

{

temp->next = ptr;

ptr->previous = temp;

firstStudent = firstStudent->previous;

}

else

{

temp->previous = ptr->previous;

temp->previous->next = temp;

temp->next = ptr;

ptr->previous = temp;

}

}

}

cout << "Элемент успешно записан";

}

void showList()

{

Student\* ptr = firstStudent;

for (int i = 1; i <= 4; i++)

{

cout << "##################### " << i << " Курс #####################\n";

while (ptr != nullptr)

{

if (ptr->course == i)

{

printLine();

cout << "Фамилия: " << ptr->secondname << endl;

cout << "Имя: " << ptr->name << endl;

cout << "Отчество: " << ptr->surname << endl;

cout << "Год рождения: " << ptr->yearOfBirth << endl;

cout << "Курс: " << ptr->course << endl;

cout << "Группа: " << ptr->group << endl;

cout << "Оценка за экзамен по ОАИП: " << ptr->oaipGrade << endl;

cout << "Оценка за экзамен по ОПИ: " << ptr->opiGrade << endl;

cout << "Оценка за экзамен по английскому: " << ptr->englishGrade << endl;

cout << "Оценка за экзамен по математитке: " << ptr->mathemathicsGrade << endl;

cout << "Оценка за экзамен по физике: " << ptr->physicsGrade << endl;

printLine();

}

ptr = ptr->next;

}

ptr = firstStudent;

}

}

void readList()

{

ifstream fdata("data.txt");

Student\* temp = new Student, \* ptr;

string buf;

if (fdata.is\_open())

{

while (!fdata.eof())

{

ptr = firstStudent;

fdata >> buf;

strcpy\_s(temp->secondname, sizeof(temp->secondname), buf.c\_str());

fdata >> buf;

strcpy\_s(temp->name, sizeof(temp->name), buf.c\_str());

fdata >> buf;

strcpy\_s(temp->surname, sizeof(temp->surname), buf.c\_str());

fdata >> temp->yearOfBirth;

fdata >> temp->course;

fdata >> temp->group;

fdata >> temp->oaipGrade;

fdata >> temp->opiGrade;

fdata >> temp->englishGrade;

fdata >> temp->mathemathicsGrade;

fdata >> temp->physicsGrade;

temp->next = nullptr;

temp->previous = nullptr;

if (firstStudent == nullptr)

{

firstStudent = lastStudent = temp;

}

else

{

while (ptr->next != nullptr && ptr->course < temp->course)

{

ptr = ptr->next;

}

if (ptr->next == nullptr)

{

temp->previous = ptr;

ptr->next = temp;

lastStudent = lastStudent->next;

}

else

{

while (ptr->next != nullptr && ptr->secondname[0] < temp->secondname[0])

{

ptr = ptr->next;

}

if (ptr->next == nullptr)

{

temp->previous = ptr;

ptr->next = temp;

lastStudent = lastStudent->next;

}

else if (ptr->previous == nullptr)

{

temp->next = ptr;

ptr->previous = temp;

firstStudent = firstStudent->previous;

}

else

{

temp->previous = ptr->previous;

temp->previous->next = temp;

temp->next = ptr;

ptr->previous = temp;

}

}

}

temp = new Student;

}

fdata.close();

}

else

{

cout << "Не удалось открыть файл data.txt для чтения\n";

}

}

void calculateAverageScore()

{

Student\* ptr = firstStudent;

double quantity = 0, sum = 0;

int userInput[3]; //0 - курс, 1 группа, 2 - предемет

cout << "Введите курс студентов, чей средний балл хотите посчитать: ";

cin >> userInput[0];

cout << "Введите группу студентов, чей средний балл хотите посчитать: ";

cin >> userInput[1];

cout << "Введите предемет, по которому хотите вычислить средний балл ";

cout << "(1 - ОАИП, 2 - ОПИ, 3 - ангийский, 4 - математика, 5 - физика):";

cin >> userInput[2];

while (ptr != nullptr)

{

if (userInput[0] == ptr->course && userInput[1] == ptr->group)

{

switch (userInput[2])

{

case 1:

sum += ptr->oaipGrade;

break;

case 2:

sum += ptr->opiGrade;

break;

case 3:

sum += ptr->englishGrade;

break;

case 4:

sum += ptr->mathemathicsGrade;

break;

case 5:

sum += ptr->physicsGrade;

break;

default:

cout << "Ошибка ввода";

break;

}

quantity++;

}

ptr = ptr->next;

}

cout << "Средний балл группы по заданному предмету: " << sum / quantity << endl;

}

void findYoungest()

{

Student\* ptr = firstStudent, \* youngestStudent = firstStudent;

int age = 0;

while (ptr != nullptr)

{

if (age < ptr->yearOfBirth)

{

age = ptr->yearOfBirth;

youngestStudent = ptr;

}

ptr = ptr->next;

}

cout << "Младшим студентом является: " << youngestStudent->secondname << " " << youngestStudent->name << " " << youngestStudent->surname;

cout << " (" << 2025 - age << " лет)\n";

}

void findOldest()

{

Student\* ptr = firstStudent, \* oldestStudent = firstStudent;

int age = 3000;

while (ptr != nullptr)

{

if (age > ptr->yearOfBirth)

{

age = ptr->yearOfBirth;

oldestStudent = ptr;

}

ptr = ptr->next;

}

cout << "Старшим студентом является: " << oldestStudent->secondname << " " << oldestStudent->name << " " << oldestStudent->surname;

cout << " (" << 2025 - age << " лет)\n";

}

void findTheBestStudent()

{

Student\* ptr = firstStudent, \* bestStudent = firstStudent;

int sum = 0, maxSum = 0;

int userInput[3]; //0 - курс, 1 группа

cout << "Введите курс, на котором хотите найти лучшего с точки зрения успеваемости, студента: ";

cin >> userInput[0];

cout << "Введите группу, в которой хотите найти лучшего с точки зрения успеваемости, студента: ";

cin >> userInput[1];

while (ptr != nullptr)

{

if (userInput[0] == ptr->course && userInput[1] == ptr->group)

{

sum = ptr->englishGrade + ptr->mathemathicsGrade + ptr->oaipGrade + ptr->opiGrade + ptr->physicsGrade;

if (sum > maxSum)

{

maxSum = sum;

bestStudent = ptr;

}

}

ptr = ptr->next;

}

printLine();

cout << "Лучший студент с точки зрения успеваемости:\n";

cout << "Фамилия: " << bestStudent->secondname << endl;

cout << "Имя: " << bestStudent->name << endl;

cout << "Отчество: " << bestStudent->surname << endl;

cout << "Год рождения: " << bestStudent->yearOfBirth << endl;

cout << "Курс: " << bestStudent->course << endl;

cout << "Группа: " << bestStudent->group << endl;

cout << "Оценка за экзамен по ОАИП: " << bestStudent->oaipGrade << endl;

cout << "Оценка за экзамен по ОПИ: " << bestStudent->opiGrade << endl;

cout << "Оценка за экзамен по английскому: " << bestStudent->englishGrade << endl;

cout << "Оценка за экзамен по математитке: " << bestStudent->mathemathicsGrade << endl;

cout << "Оценка за экзамен по физике: " << bestStudent->physicsGrade << endl;

printLine();

}

**Function.h:**

struct Student

{

char secondname[20];

char name[20];

char surname[20];

int yearOfBirth;

int group;

int course;

int oaipGrade;

int opiGrade;

int englishGrade;

int mathemathicsGrade;

int physicsGrade;

Student\* next;

Student\* previous;

};

void readList();

void showList();

void addStudent();

void printLine();

void calculateAverageScore();

void findYoungest();

void findOldest();

void findTheBestStudent();