МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В. Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа № 8**

по дисциплине: Объектно-ориентированное программирование

тема: «Создание шаблонов классов в С++»

Выполнил: ст. группы ПВ-223

Игнатьев Артур Олегович

Проверил:

асс. Черников Сергей Викторович

Белгород 2024г.

**Лабораторная работа №8**

**«Создание шаблонов классов в С++»**

**Цель работы:** Получение теоретических знаний о шаблонах классов в С++. Получение практических навыков по созданию классов-шаблонов С++.

**Вариант 3**

**Задание 1**

Реализовать шаблон класса в соответствии с указанным вариантом. Предусмотреть необходимые методы для работы со структурой данных, указанной в варианте. Предусмотреть исключительные ситуации, которые могут возникнуть в процессе работы.

**Двусвязный список**

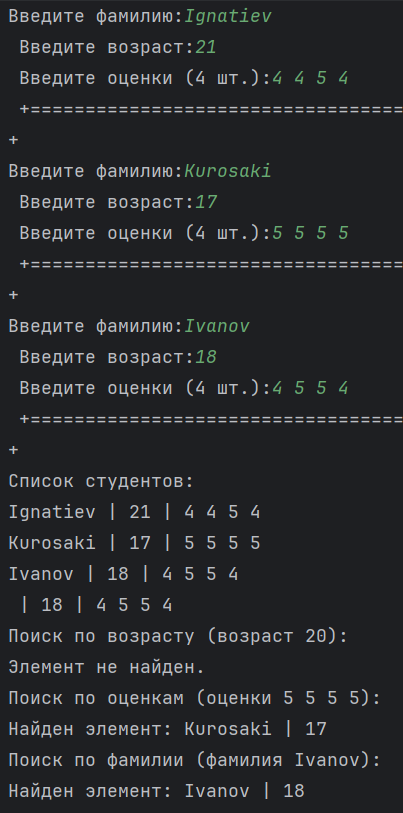
**Задание 2**

Дан файл структур с тремя полями: фамилия, возраст, оценка. Реализовать структуру для упорядоченного поиска элемента из файла по возрасту, оценки, фамилии. В качестве метода сравнения двух элементов обязательно использовать template функцию.

Код программы:

#include <iostream>  
#include <fstream>  
#include <stdexcept>  
#include <string>  
#include <windows.h>  
  
using namespace std;  
  
struct Person {  
 string name;  
 int age;  
 int marks[4];  
};  
  
template<class Type>  
class DList {  
 struct Node {  
 Type data;  
 Node\* next;  
 Node\* prev;  
 };  
  
 Node\* head;  
 Node\* tail;  
  
public:  
 DList() : head(nullptr), tail(nullptr) {}  
  
 ~DList() {  
 while (head) {  
 Node\* temp = head;  
 head = head->next;  
 delete temp;  
 }  
 }  
  
 void addNode(Type data) {  
 Node\* newNode = new Node{data, nullptr, nullptr};  
 if (!head) {  
 head = tail = newNode;  
 } else {  
 tail->next = newNode;  
 newNode->prev = tail;  
 tail = newNode;  
 }  
 }  
  
 void show() {  
 Node\* current = head;  
 while (current) {  
 cout << current->data.name << " | " << current->data.age << " | ";  
 for (int i = 0; i < 4; ++i) {  
 cout << current->data.marks[i] << " ";  
 }  
 cout << endl;  
 current = current->next;  
 }  
 }  
  
 Type\* find(bool (\*cmp)(const Type&, const Type&), const Type& key) {  
 Node\* current = head;  
 while (current) {  
 if (cmp(current->data, key)) {  
 return &(current->data);  
 }  
 current = current->next;  
 }  
 return nullptr;  
 }  
};  
  
bool compareByAge(const Person& p1, const Person& p2) {  
 return p1.age == p2.age;  
}  
  
bool compareByMarks(const Person& p1, const Person& p2) {  
 for (int i = 0; i < 4; ++i) {  
 if (p1.marks[i] != p2.marks[i]) {  
 return false;  
 }  
 }  
 return true;  
}  
  
bool compareByName(const Person& p1, const Person& p2) {  
 return p1.name == p2.name;  
}  
  
void personsToFile(const string& filename, size\_t n) {  
 ofstream file(filename);  
 if (!file.is\_open()) {  
 throw runtime\_error("Не удалось открыть файл для записи.");  
 }  
  
 for (size\_t i = 0; i < n; ++i) {  
 Person p;  
 cout << "Введите фамилию: ";  
 cin >> p.name;  
 cout << "Введите возраст: ";  
 cin >> p.age;  
 cout << "Введите оценки (4 шт.): ";  
 for (int j = 0; j < 4; ++j) {  
 cin >> p.marks[j];  
 }  
 file << p.name << " " << p.age << " ";  
 for (int j = 0; j < 4; ++j) {  
 file << p.marks[j] << " ";  
 }  
 file << endl;  
 cout << "+=============================================================+" << endl;  
 }  
 file.close();  
}  
  
int main() {  
 SetConsoleCP(1251);  
 SetConsoleOutputCP(CP\_UTF8);  
 setlocale(LC\_ALL, "Russian");  
  
 try {  
 personsToFile("persons.txt", 3);  
  
 ifstream file("persons.txt");  
 if (!file.is\_open()) {  
 throw runtime\_error("Не удалось открыть файл для чтения.");  
 }  
  
 DList<Person> list;  
 while (!file.eof()) {  
 Person p;  
 file >> p.name >> p.age;  
 for (int i = 0; i < 4; ++i) {  
 file >> p.marks[i];  
 }  
 list.addNode(p);  
 }  
 file.close();  
  
 cout << "Список студентов:" << endl;  
 list.show();  
  
 cout << "Поиск по возрасту (возраст 20):" << endl;  
 Person key\_age{"", 20, {0, 0, 0, 0}};  
 Person\* found\_age = list.find(compareByAge, key\_age);  
 if (found\_age) {  
 cout << "Найден элемент: " << found\_age->name << " | " << found\_age->age << endl;  
 } else {  
 cout << "Элемент не найден." << endl;  
 }  
  
 cout << "Поиск по оценкам (оценки 5 5 5 5):" << endl;  
 Person key\_marks{"", 0, {5, 5, 5, 5}};  
 Person\* found\_marks = list.find(compareByMarks, key\_marks);  
 if (found\_marks) {  
 cout << "Найден элемент: " << found\_marks->name << " | " << found\_marks->age << endl;  
 } else {  
 cout << "Элемент не найден." << endl;  
 }  
  
 cout << "Поиск по фамилии (фамилия Ivanov):" << endl;  
 Person key\_name{"Ivanov", 0, {0, 0, 0, 0}};  
 Person\* found\_name = list.find(compareByName, key\_name);  
 if (found\_name) {  
 cout << "Найден элемент: " << found\_name->name << " | " << found\_name->age << endl;  
 } else {  
 cout << "Элемент не найден." << endl;  
 }  
 } catch (const exception& e) {  
 cerr << "Ошибка: " << e.what() << endl;  
 return 1;  
 }  
  
 return 0;  
}

Результат работы:



**Вывод:** в ходе лабораторной работы получили теоретических знаний о шаблонах классов в С++. Получение практических навыков по созданию классов-шаблонов С++.