Министерство образования и науки Кыргызской Республики

Кыргызский государственный технический университет

им. И.Раззакова

Факультет информационных технологий

Кафедра «Программное обеспечение компьютерных систем»

Направление: 710400 «Программная инженерия»

Дисциплина: «Объектно – ориентированное программирование»

**ОТЧЕТ**

По лабораторной работе №2.

Тема: «Классы и объекты»

                                                                       Выполнила: студент группы

ПИ(б)-1-22 Кулбачаев Т.К

Проверила: Мусабаев Э.Б

**Бишкек – 2022**

**Лабораторная работа №2.**

**Тема: Классы и объекты**

1. В здании аэропорта на экранах отображается информация о самолетах, а именно: информация о пункте отправления, пункте назначения, номере рейса, времени прибытия, времени отправления, номере секции для регистрации. Экраны – это средство, которое помогает своевременно зарегистрировать и отправить пассажиров. Важно, чтобы информация на экранах была понятной и верной.

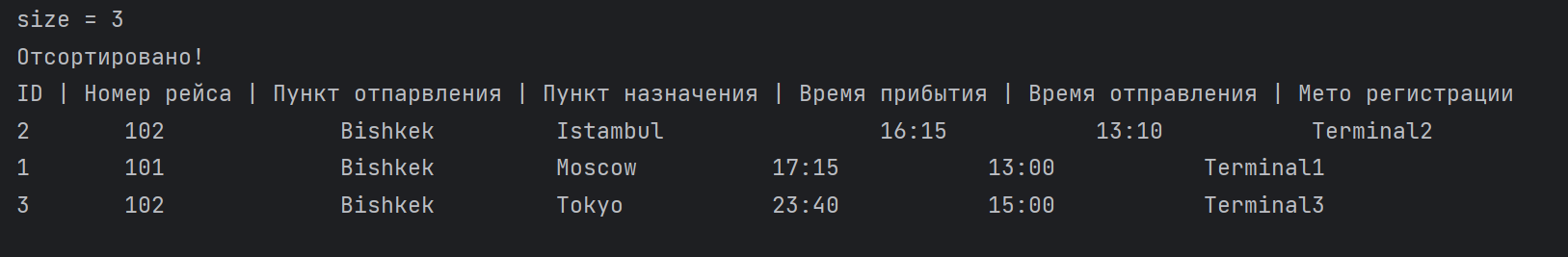
Создайте необходимую информацию в виде таблицы для такого экрана, с помощью класса Aeroflot, содержащего в описании следующие поля: номер рейса; название пункта отправления; название пункта назначения рейса; время прибытия; время отправления; место регистрации.

Напишите код программы, выполняющей следующие действия: ввод с клавиатуры значений полей объектов; сортировку записей в таблице в алфавитном порядке по названию пунктов назначения; вывод на консоль значений полей объектов класса в виде таблицы рейсов; если таких рейсов нет, выдать соответствующее сообщение.

**Код Программы:**

#include <iostream>  
#include "windows.h"  
#include <cmath>  
#include <string>  
#include <algorithm>  
  
using namespace std;  
  
class Aeroflot {  
 static int nextId;  
 int id;  
  
public:  
 int flight\_number;  
 string departure\_point;  
 string flight\_destination;  
 string arrival\_time;  
 string departure\_time;  
 string place\_of\_registration;  
  
 Aeroflot(){  
 id = nextId++;  
 }  
 void entering\_values(){  
 while (true){  
 cout << "Введите номер рейса: ";  
 if (cin >> flight\_number) {  
 break;  
 } else {  
 cin.clear();  
 cin.ignore(numeric\_limits<streamsize>::max(), '\n');  
 cout << "Ошибка ввода! Надо вводить целое число." << endl;  
 }  
 }  
  
 cin.ignore(numeric\_limits<streamsize>::max(), '\n');  
 cout << "Введите пункт отправления: ";  
 getline(cin, departure\_point);  
  
 cout << "Введите пункт назначения: ";  
 getline(cin, flight\_destination);  
  
 cout << "Введите время отправления: ";  
 getline(cin, departure\_time);  
  
 cout << "Введите время прибытия: ";  
 getline(cin, arrival\_time);  
  
 cout << "Введите место регистрации: ";  
 getline(cin, place\_of\_registration);  
  
 }  
  
 void print\_values(){  
 cout << id <<"\t" << flight\_number <<"\t\t" << departure\_point << "\t\t"<< flight\_destination <<"\t\t" << arrival\_time;  
 cout << "\t\t" << departure\_time << "\t\t" << place\_of\_registration << endl;  
 }  
  
 int getId(){  
 return id;  
 }  
  
};  
  
int Aeroflot::nextId = 1;  
  
bool compareByDestination(const Aeroflot& a, const Aeroflot& b) {  
 return a.flight\_destination < b.flight\_destination;  
}  
  
void print\_table(Aeroflot\* table, size\_t &size);  
  
int main() {  
 SetConsoleOutputCP(CP\_UTF8);  
  
 Aeroflot flight1;  
 flight1.flight\_number = 101;  
 flight1.departure\_point = "Bishkek";  
 flight1.flight\_destination = "Moscow";  
 flight1.departure\_time = "13:00";  
 flight1.arrival\_time = "17:15";  
 flight1.place\_of\_registration = "Terminal1";  
  
 Aeroflot flight2;  
 flight2.flight\_number = 102;  
 flight2.departure\_point = "Bishkek";  
 flight2.flight\_destination = "Istambul";  
 flight2.departure\_time = "13:10";  
 flight2.arrival\_time = "16:15";  
 flight2.place\_of\_registration = "Terminal2";  
  
 Aeroflot flight3;  
 flight3.flight\_number = 102;  
 flight3.departure\_point = "Bishkek";  
 flight3.flight\_destination = "Tokyo";  
 flight3.departure\_time = "15:00";  
 flight3.arrival\_time = "23:40";  
 flight3.place\_of\_registration = "Terminal3";  
  
  
 Aeroflot example;  
 example.entering\_values();  
 cout << endl;  
  
 Aeroflot all\_flights[]{flight1, flight2, flight3};  
 size\_t size = sizeof(all\_flights) / sizeof(all\_flights[0]);  
 cout << "size = " << size << endl;  
  
  
 if(size > 0){  
 sort(all\_flights, all\_flights + size, compareByDestination);  
 cout << "Отсортировано!" << endl;  
 print\_table(all\_flights, size);  
  
 }  
 else{  
 cout << "Ошибка!!! Нет объектов в таблице." << endl;  
 }  
  
  
 return 0;  
}  
  
  
  
void print\_table(Aeroflot\* table, size\_t &size){  
 cout << "ID | Номер рейса | Пункт отпарвления | Пункт назначения | Время прибытия | Время отправления | Мето регистрации" << endl;  
 for(size\_t i = 0; i < size; i++){  
 table[i].print\_values();  
 }  
}

**Результат программы:**



1. **Опишите класс «студенческая группа».**

Предусмотрите возможность: работы с переменным числом студентов; поиска студента по какому-либо признаку (например, по фамилии, дате рождения или номеру телефона); добавления или удаления записей; сортировки по разным полям.

Разработайте программу, демонстрирующую работу с этим классом.

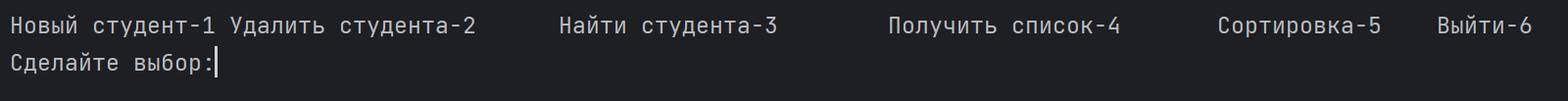
Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

**Код программы:**

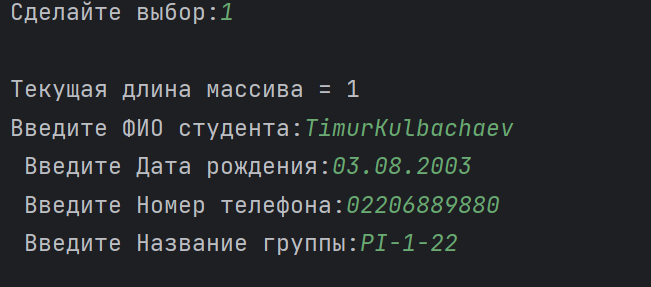
#include <iostream>  
#include "windows.h"  
#include <cmath>  
#include <string>  
#include <algorithm>  
  
using namespace std;  
  
class Student {  
 static int nextId;  
 int id;  
 string fullname;  
 string birthdate;  
 string phone;  
 string group\_name;  
  
public:  
 Student() {  
 id = nextId++;  
 }  
 int getID() {  
 return id;  
 }  
 string getPhone(){  
 return phone;  
 }  
 string getFullname(){  
 return fullname;  
 };  
 bool is\_valid() const {  
 return !fullname.empty();  
 }  
  
 void print\_values();  
 void enter\_values();  
};  
int Student::nextId = 1;  
bool compareByDestination(Student& a, Student& b) {  
 return a.getFullname() < b.getFullname();  
}  
  
  
void search\_by\_phone(Student\* object, size\_t &size);  
void add\_newStudent(Student\*& mas, size\_t &size);  
void print\_all\_object(Student\* mas, size\_t& size);  
void delete\_student(Student\*& object, size\_t &size);  
  
  
int main() {  
 SetConsoleOutputCP(CP\_UTF8);  
 size\_t size = 0;  
 Student \*pi\_one;  
 pi\_one = new Student[size];  
  
 while(true){  
 cout << "Новый студент-1\tУдалить студента-2\tНайти студента-3\tПолучить список-4";  
 cout << "\tСортировка-5\tВыйти-6" << endl;  
 cout << "Сделайте выбор: ";  
  
 int choice;  
 while (true){  
 if (cin >> choice) {  
 break;  
 } else {  
 cin.clear();  
 cin.ignore(numeric\_limits<streamsize>::max(), '\n');  
 cout << "Ошибка ввода! Надо вводить целое число. [1, 6]." << endl;  
 }  
 }  
 bool end = false;  
 switch(choice){  
 case 1:  
 cin.ignore(numeric\_limits<streamsize>::max(), '\n');  
 add\_newStudent(pi\_one, size);  
 break;  
 case 2:  
 cin.ignore(numeric\_limits<streamsize>::max(), '\n');  
 delete\_student(pi\_one, size);  
 break;  
 case 3:  
 cin.ignore(numeric\_limits<streamsize>::max(), '\n');  
 search\_by\_phone(pi\_one, size);  
 break;  
 case 4:  
 print\_all\_object(pi\_one, size);  
 break;  
 case 5:  
 sort(pi\_one, pi\_one + size, compareByDestination);  
 break;  
 case 6:  
 end = true;  
 break;  
 default:  
 cout << "Нет такого варианта!!!" << endl;  
 break;  
 }  
 if (end) {  
 break;  
 }  
 }  
  
 delete [] pi\_one;  
 return 0;  
}  
  
  
void Student::print\_values() {  
 cout << endl;  
 cout << "ID: " << id << endl;  
 cout << "ФИО студента: " << fullname << endl;  
 cout << "Дата рождения: " << birthdate << endl;  
 cout << "Номер телефона: " << phone << endl;  
 cout << "Название группы: " << group\_name << endl;  
}  
  
  
void Student::enter\_values() {  
 cout << "Введите ФИО студента: "; getline(cin, fullname);  
 cout << "Введите Дата рождения: "; getline(cin, birthdate);  
 cout << "Введите Номер телефона: "; getline(cin, phone);  
 cout << "Введите Название группы: "; getline(cin, group\_name);  
 cout << endl;  
}  
  
  
void search\_by\_phone(Student\* object, size\_t& size){  
 if(size == 0){  
 cout << "В массиве нет объектов!!!" << endl;  
 return;  
 }  
 string phone\_number;  
 cout << "Введите номер телефона: ";  
 getline(cin, phone\_number);  
  
 for(int i = 0; object[i].is\_valid(); i++){  
 if(object[i].getPhone() == phone\_number){  
 cout << "Student found: " << endl;  
 object[i].print\_values();  
 break;  
 }  
 }  
}  
  
  
void add\_newStudent(Student\*& mas, size\_t &size){  
 Student \*new\_pi\_one = new Student[size+1];  
 for(size\_t i = 0; i < size; i++){  
 new\_pi\_one[i] = mas[i];  
 }  
 size++;  
 cout << endl << "Текущая длина массива = " << size << endl;  
 new\_pi\_one[size - 1].enter\_values();  
  
 delete [] mas;  
 mas = new\_pi\_one;  
}  
  
  
void print\_all\_object(Student\* mas, size\_t& size){  
 if(size == 0){  
 cout << "В массиве нет объектов!!!" << endl;  
 return;  
 }  
 for(size\_t i = 0; i < size; i++){  
 mas[i].print\_values();  
 }  
 cout << endl;  
}  
  
  
void delete\_student(Student\*& object, size\_t &size){  
 if(size == 0){  
 cout << "В массиве нет объектов!!!" << endl;  
 return;  
 }  
 string student\_name;  
 cout << "Введите ФИО студента для удалния: ";  
 getline(cin, student\_name);  
  
 cout << "size = " << size << endl;  
  
 for(size\_t i = 0; i < size; i++) {  
 if (object[i].getFullname() == student\_name) {  
 size\_t index\_to\_remove = i;  
 Student \*new\_pi\_one = new Student[size - 1];  
  
 for (size\_t t = 0, j = 0; t < size; ++t) {  
 if (t != index\_to\_remove) {  
 new\_pi\_one[j++] = object[t];  
 }  
 }  
  
 delete[] object;  
 object = new\_pi\_one;  
 size--;  
 cout << "выход из delete!" << endl;  
 }  
 }  
}

**Результат программы:**

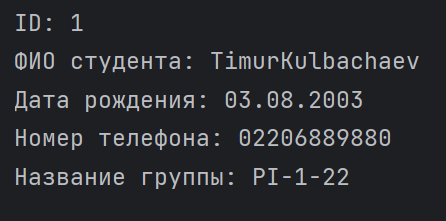
**Menu**



**Добавление студента**



**Получить список**

****

1. **Опишите класс, реализующий стек (Stack).**

Разработать программу, использующую этот класс для моделирования Т-образного сортировочного узла на железной дороге. Программа должна разделять на два направления состав, состоящий из вагонов двух типов (на каждое направление формируется состав из вагонов одного типа). Предусмотреть возможность формирования состава из файла или с клавиатуры. Возможно использование стандартных функций при работе со стеком в виде контейнера из библиотеки STL:

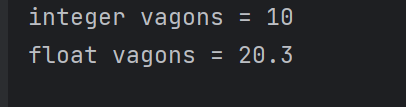
**push()** - поместить элемент в вершину стека;

**pop()** - удалить элемент из вершины стека;

**Код программы:**

#include <iostream>  
#include "windows.h"  
#include <stdexcept>  
#include <iterator>  
#include <typeinfo>  
#include <cstring>  
  
using namespace std;  
  
template<typename T>  
class Stack {  
 struct Node {  
 T value;  
 Node \*next;  
 };  
 Node \*base;  
public:  
 Stack() : base(nullptr) {}  
 ~Stack() {  
 clear();  
 }  
 void push(const T &value) {  
 base = new Node{value, base};  
 }  
 void pop() {  
 if (isEmpty()) {  
 underflow\_error("stack is empty");  
 }  
 Node \*current = base;  
 base = base->next;  
 delete current;  
 }  
  
 bool isEmpty() const {  
 return base == nullptr;  
 }  
  
 const T &top() const {  
 if (isEmpty()) {  
 underflow\_error("stack is empty");  
 }  
 return base->value;  
 }  
 void clear() {  
 while (base != nullptr) {  
 Node \*current = base;  
 base = base->next;  
 delete current;  
 }  
 }  
  
};  
  
template <class T>  
void add\_to\_stack(T value, Stack<float> \*&fl, Stack<int> \*&integer);  
  
  
int main() {  
 SetConsoleOutputCP(CP\_UTF8);  
 Stack<int> first\_stack;  
 Stack<float> second\_stack;  
 Stack<int> \*first = &first\_stack;  
 Stack<float> \*second = &second\_stack;  
  
 float t = 20.3;  
 int f = 10;  
  
 add\_to\_stack(t, second, first);  
 add\_to\_stack(f, second, first);  
  
 cout << "integer vagons = " << first\_stack.top() << endl;  
 cout << "float vagons = " << second\_stack.top() << endl;  
  
 return 0;  
}  
  
template <class T>  
void add\_to\_stack(T value, Stack<float> \*&fl, Stack<int> \*&integer){  
 const char \*variable\_type = typeid(value).name();  
 if(strcmp(variable\_type, "f") == 0){  
 fl->push(value);  
 }  
 else{  
 if(strcmp(variable\_type, "i") == 0){  
 integer->push(value);  
 }  
 }  
 variable\_type = nullptr;  
  
}

**Результат программы:**

****

**Спецификации требования к ПО**

1. Общие сведения:
   1. Наименование системы.

**Полное название системы:** Программное обеспечение Студенческая группа

**Перечень документов, на основании которых ведутся работы:**

Основанием для программной обеспечении для учета студентов является договор между исполнителем и заказчиком.

* 1. Наименование организаций заказчика и разработчика.

**Заказчик (преподаватель):**

Мусабаев Эмильбек Бахытжанович

**Разработчик (студент):**

Кулбачаев Тимур

Адрес: Ташкумырская 6

Телефон: +996 220 68 89 80

* 1. Плановые сроки начала и окончания работы.

Дата начало работ: 26.01.2024.

Дата окончания работ: 30.01.2024.

1. Назначение и цели создания системы
   1. Назначение:

Программа предназначена для учета студентов использованием объектно-ориентированным программированием.

* 1. **Цели разработки:**

1. Перенос учета студентов в электронный формат
2. Защитить от потери данных.
   1. Требования к функциям:

Программа должна обеспечивать возможность выполнения следующих функций:

1. Пользователь должен иметь возможность добавить нового студента.
2. ПО должно выводить список студентов.
3. ПО должно обеспечить возможность поиска по номеру телефона.
4. ПО должно обрабатывать возможные ошибки, такие как сортировка пустого списка и т.п. Также сообщать об них пользователю.
5. Пользователь должен иметь возможность удалить определенного студента.
   1. Требования к надежности:

Предусмотреть контроль входной информации. Предусмотреть блокирование некорректных действий пользователя при работе с системой.

* 1. Требования к видам обеспечения:
     1. Требования к информационному обеспечению системы:
     2. **Требования к лингвистическому обеспечению системы:**

ПО системы для взаимодействия с пользователем будет использовать русский язык.

* + 1. **Требования к информационной системе:**

В качестве основы для разработки будет использоваться среда разработки CLion, язык с++

* + 1. **Тип разрабатываемой системы:**

**Консольное приложение.**