**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**КЫРГЫЗСКОЙ РЕСПУБЛИКИ**

**КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. И. Раззакова**

**ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

Кафедра: **Программное обеспечение компьютерных систем**

Дисциплина «**Объектно-Ориентированное Программирование**»

Отчет

Лабораторная работа №4

Выполнила: студент группы ПИ-2-21

Тойчубекова Айчурок

Проверил: Мусабаев Э. Б.

**Бишкек 2024**

**Задания:**

1. Для участия в ежегодном трансконтинентальном Ралли-марафоне «Дакар» по усложненной трассе в Южной Америке выбраны грузовики [КАМАЗ](https://ru.wikipedia.org/wiki/%D0%9A%D0%90%D0%9C%D0%90%D0%97) и [Tatra](https://ru.wikipedia.org/wiki/Tatra), которые после 2000 года лидируют в зачете грузовиков.

Решите задачу сравнения скоростей движения грузовиков по разным по проходимости участкам трассы, а именно: по равнине, горам, пустыне. Создайте и выдайте на экран таблицу результатов ралли - марафона. Определите победителя.

Для решения задачи используйте классы **Kamaz** и **Tatra,** а такжефункциюсравнения скоростей **FrCreater (**). Функция  **FrCreater (**) возвращает число +1, если объект **kamaz** движется быстрее объекта **tatra**; нуль, если их скорости одинаковы; число -1, если объект **kamaz** движется медленнее объекта **tatra** .

Оба класса содержат поля: «скорость» и «наименование» грузовой машины, а также методы: инициализация и отображение полей на экране. Определитесь с идентификаторами доступа к членам класса, не нарушая принципа инкапсуляции.

1. Сделайте функцию **FrCreater(**) дружественной: классу **Kamaz,** обоим классам **Kamaz** и **Tatra.**
2. Сделайте класс **Tatra** дружественным классу **Kamaz.**

#include <iostream>

#include <string>

#include <ctime>

using namespace std;

class Tatra;

class Kamaz {

private:

string name;

double speed;

public:

Kamaz(const string& name, double speed) : name(name), speed(speed) {}

friend int FrCreater(const Kamaz& kamaz, const Tatra& tatra);

void display() const {

cout << "Камаз " << name << ", скорость: " << speed << " км/ч" << endl;

}

};

class Tatra {

private:

string name;

double speed;

public:

Tatra(const string& name, double speed) : name(name), speed(speed) {}

friend int FrCreater(const Kamaz& kamaz, const Tatra& tatra);

void display() const {

cout << "Tatra " << name << ", скорость: " << speed << " км/ч" << endl;

}

friend class Kamaz;

};

int FrCreater(const Kamaz& kamaz, const Tatra& tatra) {

if (kamaz.speed > tatra.speed)

return 1;

else if (kamaz.speed == tatra.speed)

return 0;

else

return -1;

}

int main() {

setlocale(LC\_ALL, "Russian");

srand(time(0));

Kamaz kamaz("Мастер", rand() % 100 + 100); // Скорость от 100 до 200 км/ч

Tatra tatra("Phoenix", rand() % 80 + 80); // Скорость от 80 до 160 км/ч

kamaz.display();

tatra.display();

int result = FrCreater(kamaz, tatra);

if (result == 1)

cout << "Камаз быстрее Tatra на участке трассы" << endl;

else if (result == 0)

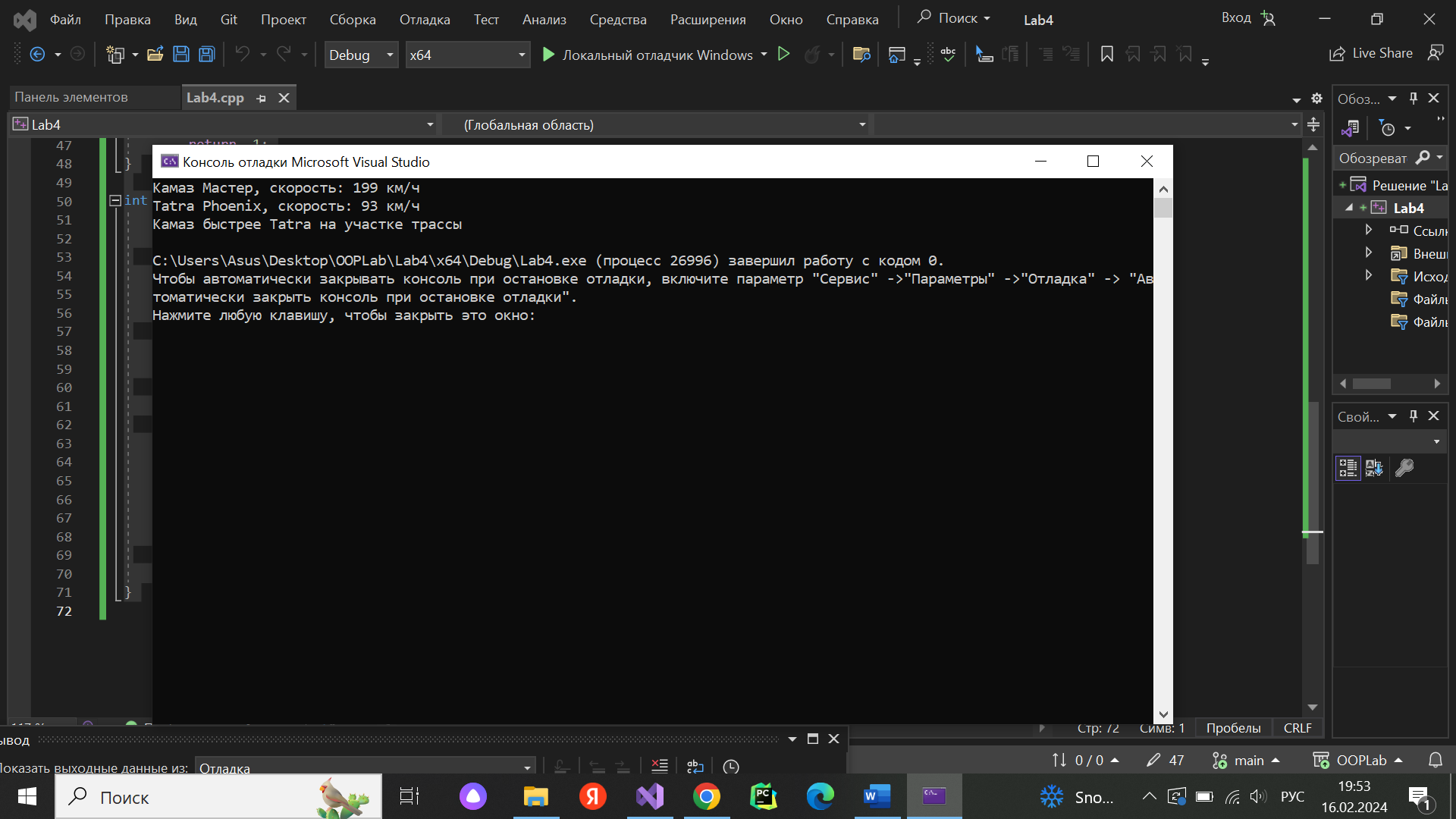
cout << "Скорости Kamaz и Tatra равны на участке трассы" << endl;

else

cout << "Tatra быстрее Kamaz на участке трассы" << endl;

return 0;

}



**ВОПРОСЫ К ЗАЩИТЕ ЛАБОРАТОРНОЙ РАБОТЫ №4**

1. Что такое дружественная функция, для чего она нужна?

Дружественная функция — это функция, объявленная вне тела класса, но имеющая доступ ко всем его членам, включая закрытые. Она может быть использована для предоставления внешним функциям доступа к закрытым членам класса.

2. Как она объявляется, каков её синтаксис?

- Дружественная функция объявляется внутри тела класса с использованием ключевого слова `friend` перед её объявлением.

***friend*** тип имя ([список\_ параметров])

3. Через что дружественная функция может получить доступ к закрытому полю класса?

- Дружественная функция получает доступ к закрытым полям класса через использование ключевого слова `friend`. Она рассматривается как "друг" класса и, следовательно, имеет доступ к его закрытым членам.

4. Какой вариант вызова ДФ правильный:

5. a.show() или show(a)?

- Правильный вариант вызова дружественной функции – `show(a)`.

6. Наследуется ли дружественная функция?

- Нет, дружественная функция не наследуется. Она имеет доступ только к членам класса, для которого была объявлена как друг.

7. Может ли дружественная функция быть дружественной более чем к одному классу?

- Да, дружественная функция может быть дружественной более чем к одному классу. Для этого её нужно объявить с использованием ключевого слова `friend` внутри каждого класса.

8. Для решения каких задач есть смысл использовать функцию, дружественную двум и более классам?

- Дружественные функции могут быть полезны, если есть необходимость взаимодействия между несколькими классами, при этом необходим доступ к закрытым членам каждого из них.

9. Когда нужно использовать дружественные классы?

- Дружественные классы могут использоваться, когда один класс должен иметь доступ к закрытым членам другого класса, а также когда требуется обмен информацией между классами.

10. Какому принципу ООП не соответствуют дружественные функции?

- Дружественные функции могут нарушать принцип инкапсуляции, так как предоставляют внешнему коду доступ к закрытым членам класса.

11. Как можно ограничить использование ДФ?

- Используйте ключевое слово `friend` только для тех функций, которым действительно необходим доступ к закрытым членам класса. Минимизируйте количество дружественных функций, чтобы уменьшить потенциальные нарушения инкапсуляции.

12. В каких крайних случаях нужно использовать ДФ?

- Дружественные функции следует использовать осторожно и, как правило, в крайних случаях, когда другие методы взаимодействия между классами оказываются недостаточными. Они могут быть полезны, например, при реализации сложных операций, которые требуют доступа к закрытым членам нескольких классов.