**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**КЫРГЫЗСКОЙ РЕСПУБЛИКИ**

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

**ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

Кафедра: **Программное обеспечение компьютерных систем**

Курс «Объектно-ориентированное программирование»

**ОТЧЕТ**

**Лабораторная работа №4**

Выполнил: студент группы ПИ-3-21

**Икрамов Абдул Муним**

Проверил: Мусабаев Э.Б.

**Бишкек 2023**

# **Лабораторная работа №4**

**Тема: Дружественные функции и дружественные классы**

**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.**

**Задания:**

*#include* <iostream>

*#include* <string>

using namespace std;

class Tatra;

class Kamaz {

private:

int speed;

string name;

public:

Kamaz(int *s*, string *n*) : speed(*s*), name(*n*) {}

void display() {

cout << "Kamaz - Название: " << name << ", Скорость: " << speed << " км/ч" << endl;

}

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

friend class Tatra;

};

class Tatra {

private:

int speed;

string name;

public:

Tatra(int *s*, string *n*) : speed(*s*), name(*n*) {}

void display() {

cout << "Tatra - Название: " << name << ", Скорость: " << speed << " км/ч" << endl;

}

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

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* -1;

*else*

*return* 0;

}

int main() {

Kamaz kamaz(120, "Грузовик Камаз");

Tatra tatra(110, "Грузовик Tatra");

kamaz.display();

tatra.display();

int result = FrCreater(kamaz, tatra);

*if* (result == 1)

cout << "Грузовик Камаз быстрее, чем грузовик Tatra." << endl;

*else* *if* (result == -1)

cout << "Грузовик Tatra быстрее, чем грузовик Камаз." << endl;

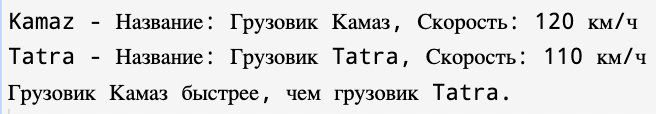
*else*

cout << "Грузовик Камаз и грузовик Tatra имеют одинаковую скорость." << endl;

*return* 0;

}

Тесты:



**Ответы на вопросы:**

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

**Дружественная функция**—это функция, не являющаяся членом класса, но обеспечивающая доступ ко всем членам класса, для которого она дружественна. Независима от класса **дружественная функция**находится вне класса. Но она объявляется внутри класса как обычная функция и перед ней указывается ключевое слово **friend**.  Вне класса перед описанием дружественной функции это ключевое слово не ставится**.**

**2. Как она объявляется, каков её синтаксис?**

Синтаксис прототипа дружественной функции

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

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

Так как дружественная функция не является методом класса, для которого она дружественна, нельзя вызвать дружественную функцию обычным способом в точечном формате с помощью команды **a.show()**, а можно только командой **show(a).**Так как, хотя дружественная функция "знает" о закрытом поле класса, для которого она является дружественной, доступ к нему она может получить только через объект этого класса, который объявлен внутри функции и передан ей в качестве аргумента. ДФ вызывается для объекта класса.

**4. Какой вариант вызова ДФ правильный:**

a.show() или show(a)?

Если функция дружественная, то второй вариант.

**5. Наследуется ли дружественная функция?**

Дружественная функция не наследуется. То есть, если в базовый класс дружественная функция включается, то она не является таковой для производных классов.

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

Дружественная функция может быть дружественной более чем к одному классу.

**7. Для решения каких задачах есть смысл использовать функцию, дружественную двум и более классам?**

Такая дружественная функция полезна тогда, когда у двух и более разных классов имеется нечто общее, с чем необходимо произвести какие-то действия, например, сравнение или арифметические действия закрытых полей объектов, принадлежащих разным классам.

**8. Когда нужно использовать дружественные классы?**

С одной стороны, дружественные функции повышают гибкость языка, но, с другой стороны, они не соответствуют принципу ограничения доступа к данным, в соответствии с которым только методы класса могут иметь доступ к сокрытым данным класса. Множество дружественных функций фактически стирают границы между классами, поэтому они должны быть разбросаны по программе и встречаться как можно реже.

**9. Какому принципу ООП не соответствуют дружественные функции?**

Инкапсуляция.

**10. Как можно ограничить использование ДФ?**

Не использовать их, использовать открытые методы класса. Или каждый раз наследуя один класс от дружественного, в таком случае, ДФ не наследуется.

**11. В каких крайних случаях нужно использовать ДФ?**

Если требуется для базового класса сделать доступ к приватным данным, а для наследуемых нет. В этом случае играет роль то, что дружественные функции не наследуются.