Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

Пермский национальный исследовательский политехнический университет

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

**ОТЧЕТ**

**Тема:** Лабораторная работа №4

Семестр: 2

Выполнил студент ИВТ-23-2б:

Синицын Владислав Сергеевич

(дата, подпись)

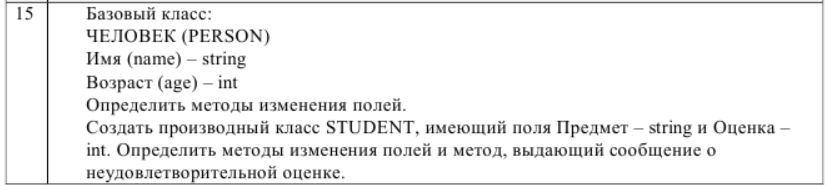
Проверила:

Полякова Ольга Андреевна

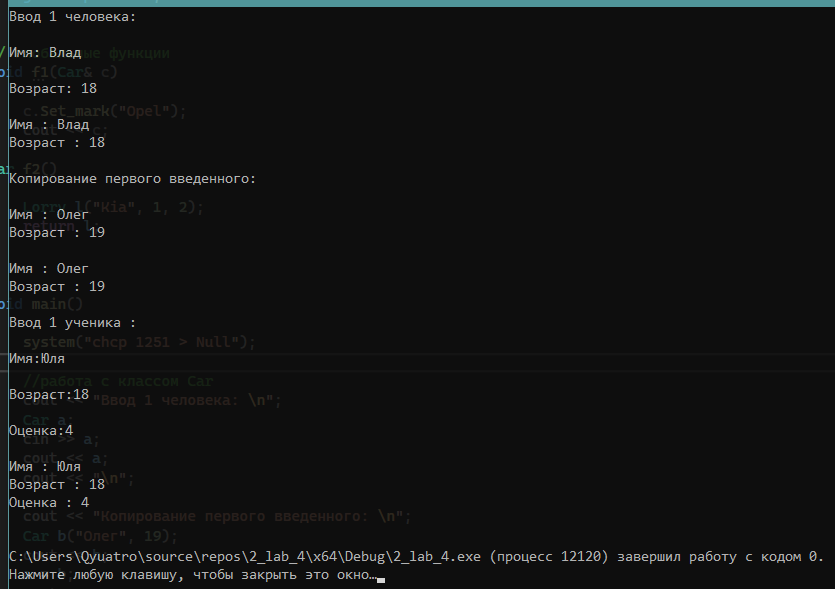
(дата, подпись)

Пермь 2024

Задание:



Программное решение:



***Lab\_4.cpp***

#include <iostream>

#include "Car.h"

#include "Lorry.h"

using namespace std;

// глобальные функции

void f1(Car& c)

{

c.Set\_mark("Opel");

cout << c;

}

Car f2()

{

Lorry l("Kia", 1, 2);

return l;

}

void main()

{

system("chcp 1251 > Null");

//работа с классом Car

cout << "Ввод 1 человека: \n";

Car a;

cin >> a;

cout << a;

cout << "\n";

cout << "Копирование первого введенного: \n";

Car b("Олег", 19);

cout << b;

a = b;

cout << a;

cout << "\n";

//работа с классом Lorry

cout << "Ввод 1 ученика : \n";

Lorry c;

cin >> c;

cout << c;

cout << "\n";

}

***car.cpp***

#include "Car.h"

//конструктор без параметров

Car::Car(void)

{

name = "";

age = 0;

}

//деструктор

Car::~Car(void)

{

}

//конструктор с параметрами

Car::Car(string N, int A)

{

name = N;

age = A;

}

//конструктор копирования

Car::Car(const Car& car)

{

name = car.name;

age = car.age;

}

//модификаторы

void Car::Set\_cyl(int A) // ставим возраст

{

age = A;

}

void Car::Set\_mark(string N) // ставим имя

{

name = N;

}

//перегрузка операции присваивания

Car& Car::operator=(const Car& c)

{

if (&c == this)return \*this;

name = c.name;

age = c.age;

return \*this;

}

//глобальная функция для ввода

istream& operator>>(istream& in, Car& c)

{

cout << "\nИмя: "; in >> c.name;

cout << "\nВозраст: "; in >> c.age;

return in;

}

//глобальная функция для вывода

ostream& operator<<(ostream& out, const Car& c)

{

out << "\nИмя : " << c.name;

out << "\nВозраст : " << c.age;

out << "\n";

return out;

}

***car.h***

#pragma once

#include <string>

#include <iostream>

using namespace std;

class Car

{

//конструктор без параметров

public:

Car(void);

public:

//деструктор

virtual ~Car(void);

//констрктор с параметрами

Car(string, int);

//конструктор копирования

Car(const Car&);

//селекторы

string Get\_mark() { return name; }

int Get\_cyl() { return age; }

//модификаторы

void Set\_mark(string);

void Set\_cyl(int);

void Set\_power(int);

//перегрузка операции присваивания

Car& operator=(const Car&);

//глобальные операторы-функции ввода-вывода

friend istream& operator>>(istream& in, Car& c);

friend ostream& operator<<(ostream& out, const Car& c);

//атрибуты

protected:

string name;

int age;

};

***lorry.cpp***

#include "Lorry.h"

//конструктор без параметров

Lorry::Lorry(void) :Car()

{

mark = 0;

}

//дестрктор

Lorry::~Lorry(void)

{

}

//конструктор с параметрами

Lorry::Lorry(string M, int C, int G) :Car(M, C)

{

mark = G;

}

//конструктор копирования

Lorry::Lorry(const Lorry& L)

{

name = L.name;

age = L.age;

mark = L.mark;

}

//модификатор

void Lorry::Set\_Gruz(int G)

{

mark = G;

}

//оперция присваивания

Lorry& Lorry::operator=(const Lorry& l)

{

if (&l == this)return \*this;

name = l.name;

age = l.age;

mark = l.mark;

return \*this;

}

//операция ввода

istream& operator>>(istream& in, Lorry& l)

{

cout << "\nИмя:"; in >> l.name;

cout << "\nВозраст:"; in >> l.age;

cout << "\nОценка:"; in >> l.mark;

return in;

}

//операция вывода

ostream& operator<<(ostream& out, const Lorry& l)

{

out << "\nИмя : " << l.name;

out << "\nВозраст : " << l.age;

out << "\nОценка : " << l.mark;

out << "\n";

return out;

}

***lorry.h***

#pragma once

#include "car.h"

//класс Lorry наследуется от класса Car

class Lorry :

public Car

{

public:

Lorry(void);//конструктор без параметров

public:

~Lorry(void);//деструктор

Lorry(string, int, int);//конструктор с параметрами

Lorry(const Lorry&);//конструктор копирования

int Get\_gruz() { return mark; }//модификатор

void Set\_Gruz(int);//селектор

Lorry& operator=(const Lorry&);//операция присваивания

friend istream& operator>>(istream& in, Lorry& l);//операция ввода

friend ostream& operator<<(ostream& out, const Lorry& l); //операция вывода

protected:

int mark;//атрибут грузоподъемность

};

