Министерство образования и науки Российской Федерации

Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский**

**политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.1**

Дисциплина: «Информатика»

Тема: Объектно-ориентированное программирование. Инкапсуляция

Вариант 8

Выполнил:

Студент группы РИС-20-1б

Тараканов Д. М.

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

Пермь

2021 год

# **Цель работы**

1. Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.
2. Использование классов и объектов в ОО программе.

# **Постановка задачи**

1. Реализовать определение нового класса. Для демонстрации работы с объектами написать главную функцию. Продемонстрировать разные способы создания объектов и массивов классов.
2. Структура-пара – структура с двумя полями, которые обычно имеют имена first и second. Требуется реализовать тип данных с помощью такой структуры. Во всех заданиях должны присутствовать:
3. Метод инициализации Init (метод должен контролировать значения аргументов на корректность);
4. Ввод с клавиатуры Read;
5. Вывод на экран Show;
6. Реализовать внешнюю функцию make\_тип(), где тип – тип реализуемой структуры. Функция должна получать значения для полей структуры как параметры функции и возвращать структуру как результат. При передаче ошибочных параметров следует выводить сообщение и заканчивать работу.

(8 вариант)

1. Поле first – дробное число x, координата точки.
2. Поле second – дробное число y, координата точки.
3. Реализовать метод distance (double x1, double y1) – вычисление расстояния от точки с координатами (first, second) до начала координат.

# **Анализ задачи**

1. Определить какие действия предстоит выполнить:

* Разработать функцию int main(), в которой происходит считывание данных введённых в консоль, а именно координаты точки (X,Y), и дальнейший передачей в различные методы.

coordinates A;

coordinates B;

cout << "\nТочка А\nx = 4.5\ny = 3.6";

A.Init(4.5, 3.6);

cout << "\n\nТочка А";

A.Show();

cout << "\nРасстояние от начала координат до точки А = " << A.distance() << endl;

cout << "\nТочка B";

B.Read();

cout << "\nТочка B";

B.Show();

cout << "\nРасстояние от начала координат до точки B = " << B.distance() << endl;

* Создать заголовочный файл coordinates.h и описать в нем структуру и её методы.

struct coordinates {

double first;

double second;

void Init(double, double);

void Read();

void Show();

double distance();

};

* Создать файл coordinates.cpp и определить в нем методы выше упомянутой структуры.

#include <iostream>

#include "coordinates.h"

using namespace std;

void coordinates::Init(double F, double S) {

first = F;

second = S;

}

void coordinates::Read() {

cout << "\nx = "; cin >> first;

cout << "y = "; cin >> second;

}

void coordinates::Show() {

cout << "\nКоординаты: (" << first;

cout << " ; " << second;

cout << ")\n";

}

double coordinates::distance() {

return sqrt(pow(first, 2) + pow(second, 2));

}

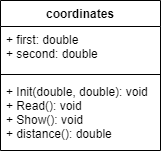
1. С какими типами данных предстояло работать

* Для хранения координаты X, Y точки реализованы переменые first, second типа double.

double first;

double second;

# **UML диаграмма классов**



# **Код программы на языке C++**

Заголовочный файл coordinates.h

#pragma once

struct coordinates {

double first;

double second;

void Init(double, double); //Метод инициализации полей

void Read(); //Метод чтения значений полей

void Show(); //Метод вывода значений полей

double distance(); //Метод вычисления расстояния

};

Обьявление класса в coordinates.cpp

#include <iostream>

#include "coordinates.h"

using namespace std;

//Реализация метода инициализации полей структуры

void coordinates::Init(double F, double S) {

first = F;

second = S;

}

//Реализация метода для чтения значнений полей структуры

void coordinates::Read() {

cout << "\nx = "; cin >> first;

cout << "y = "; cin >> second;

}

//Реализация метода для вывода значений полей структуры

void coordinates::Show() {

cout << "\nКоординаты: (" << first;

cout << " ; " << second;

cout << ")\n";

}

//Реализация метода нахождения расстояния

double coordinates::distance() {

return sqrt(pow(first, 2) + pow(second, 2));

}

Main файл

#include <iostream>

#include "coordinates.h"

using namespace std;

coordinates make\_coordinates(double F, double S) {

coordinates temp;

temp.Init(F,S);

return temp;

}

void main() {

system("chcp 1251");

//Через main

coordinates A;

coordinates B;

cout << "\nТочка А\nx = 4.5\ny = 3.6";

A.Init(4.5, 3.6);

cout << "\n\nТочка А";

A.Show();

cout << "\nРасстояние от начала координат до точки А = " << A.distance() << endl;

cout << "\nТочка B";

B.Read();

cout << "\nТочка B";

B.Show();

cout << "\nРасстояние от начала координат до точки B = " << B.distance() << endl;

//Указатель

coordinates\* X = new coordinates;

cout << "\nТочка X\nx = 2.4\ny = 0.5";

X->Init(2.4, 0.5);

cout << "\n\nТочка X";

X->Show();

cout << "\nРасстояние от начала координат до точки X = " << X->distance();

delete X;

//Массив

coordinates mas[3];

for (int i = 0; i < 3; i++) {

cout << "\n\nТочка C" << i + 1;

mas[i].Read();

cout << "\nТочка C" << i + 1;

mas[i].Show();

cout << "\nРасстояние от начала координат до точки C" << i+1 << " = " << mas[i].distance();

}

//Динамический массив

coordinates\* p\_mas = new coordinates[3];

for (int i = 0; i < 3; i++) {

cout << "\n\nТочка D" << i + 1;

p\_mas[i].Read();

cout << "\nТочка D" << i + 1;

p\_mas[i].Show();

cout << "\nРасстояние от начала координат до точки D" << i + 1 << " = " << p\_mas[i].distance();

}

delete[] p\_mas;

//Через функцию make\_coordinates

double x, y;

cout << "\n\nТочка F";

cout << "\nx = "; cin >> x;

cout << "y = "; cin >> y;

coordinates F = make\_coordinates(x, y);

cout << "\nТочка F";

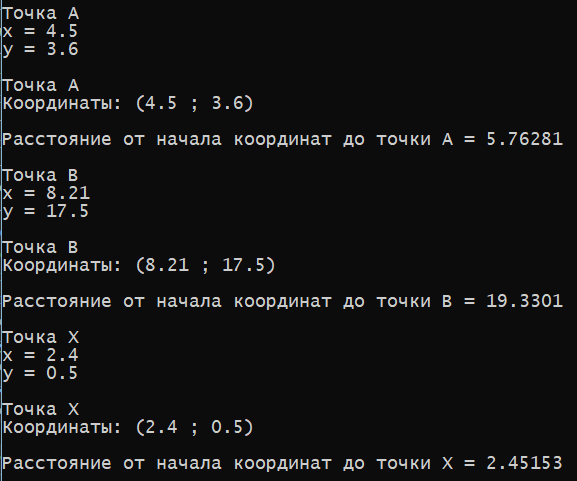
F.Show();

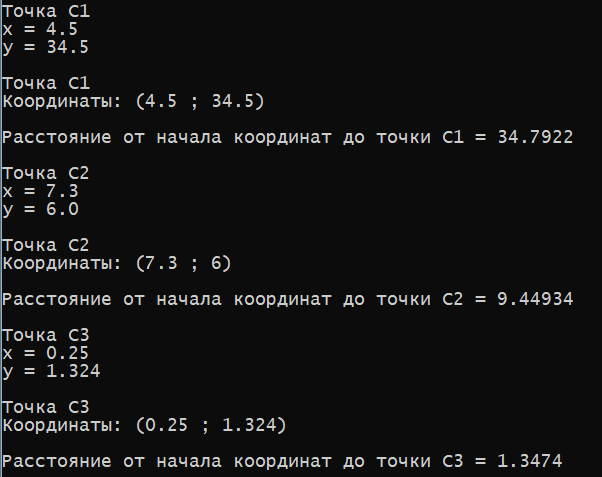
cout << "\nРасстояние от начала координат до точки F = " << F.distance() << endl;

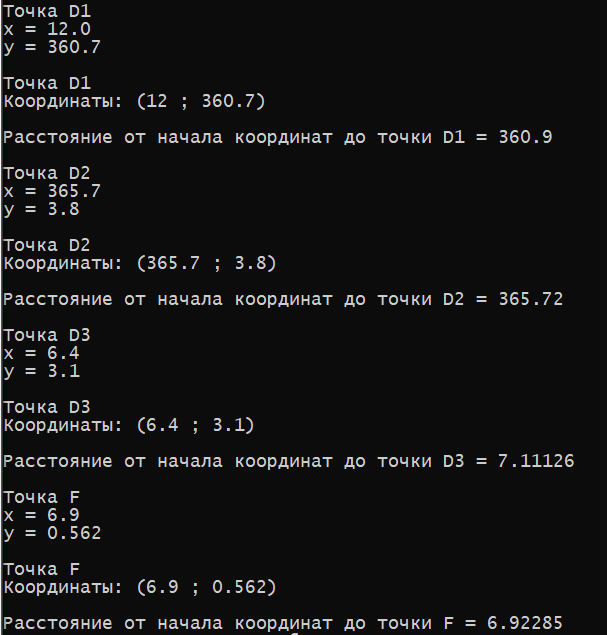
system("pause");

}

# **Скриншоты тестов**





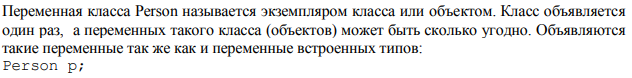


**Контрольные вопросы**

1. Что такое класс?



1. Что такое объект(экземпляр) класса?



1. Как называются поля класса?

Полями (атрибутами) называются данные класса.

1. Как называются функции класса?

Функции класса называются методами.

1. Для чего используется спецификатор доступа?

Спецификаторы доступа управляют видимостью элементов класса

1. Для чего используется спецификатор public?

Спецификатор public используется, чтобы задать общедоступную видимость.

1. Для чего используется спецификатор доступа private?

Спецификатор private используется, чтобы задать видимость только внутри класса для полей или методов под этим спецификатором

1. Если описание класса начинается со спецификатора class, то какой спецификатор доступа будет использоваться по умолчанию?

Для спецификатора по умолчанию используется private.

1. Если описание класса начинается со спецификатора struct, то какой спецификатор доступа будет использоваться по умолчанию?

Для спецификатора struct по умолчанию используется public.

1. Какой спецификатор доступа должен использоваться при описании интерфейса класса? Почему?

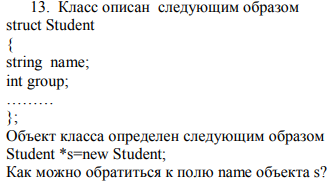
Интерфейс класса описывается после спецификатора public, чтобы к методам класса можно было обратится из других функций.

1. Каким образом можно изменить значения атрибутов экземпляра класса?

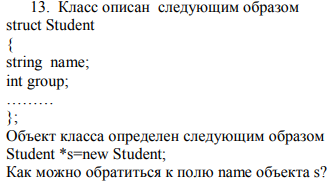
Значения атрибутов экземпляра класса можно изменить с помощью методов класса – модификаторов.

1. Каким образом можно получить значения атрибутов экземпляра класса?

Значения атрибутов экземпляра класса можно получить с помощью методов класса – селекторов.

1. Класс описан следующим образом

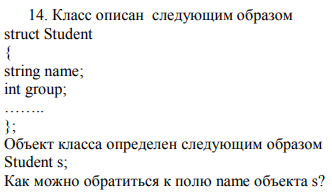
Объект класса определен следующим образом



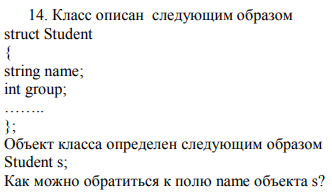
Как можно обратится к полю name объекта s?

К полю name можно обратится через “->”, так как оно имеет открытый доступ public.

1. Класс описан следующим образом



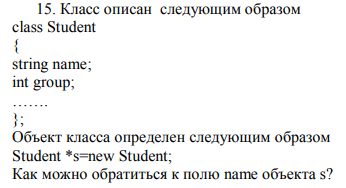
Объект класса определён следующим образом



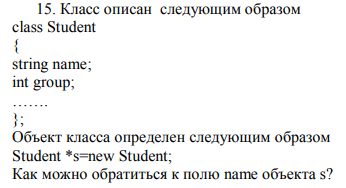
Как можно обратится к полю name объекта s?

К полю name можно обратится через “.”, так ка оно имеет открытый доступ public.

1. Класс описан следующим образом



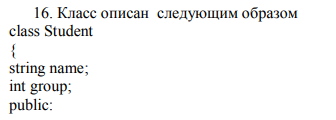
Объект класса определен следующим образом

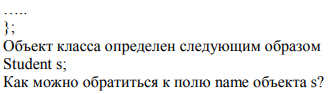


Как можно обратится к полю name объекта s?

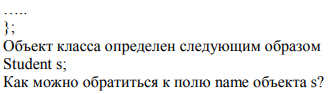
Необходимо прописать селектор и модификатор для класса Student.

1. Класс описан следующим образом





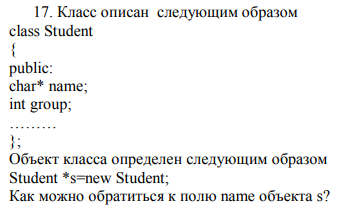
Объект класса определен следующим образом



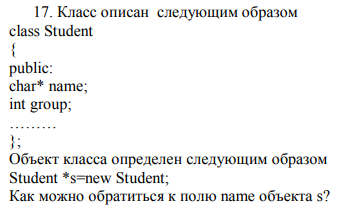
Как можно обратится к полю name объекта s?

Необходимо прописать селектор и модификатор для класса Student.

1. Класс описан следующим образом



Объект класса определен следующим образом



Как можно обратится к полю name объекта s?

К полю name можно обратится через “->”, так как оно имеет открытый доступ public.