Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

Тема: «Классы и объекты. Инкапсуляция.»

Выполнил работу

Студент группы РИС-22-2Б

Вахрушева А.В.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

Постановка задачи.

Поле first - дробное число x, координата точки, поле second - дробное число y, координата точки. Реализовать метод distance(double x1, double y1) – вычисление расстояния от точки с координатами (first, second) до точки с координатами (x1,y1).

1. Реализовать определение нового класса. Для демонстрации работы с

объектами написать главную функцию. Продемонстрировать разные

способы создания объектов и массивов объектов.

1. 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:

* метод инициализации Init (метод должен контролировать

значения аргументов на корректность);

* ввод с клавиатуры Read;
* вывод на экран Show.

1. Реализовать внешнюю функцию make\_тип(), где тип – тип

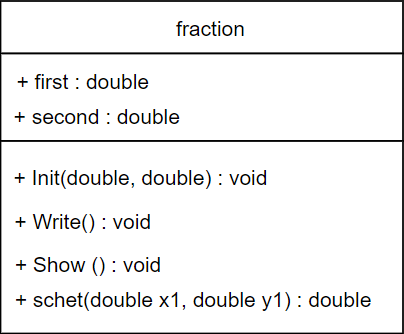
реализуемой структуры. Функция должна получать значения для

полей структуры как параметры функции и возвращать структуру

как результат. При передаче ошибочных параметров следует

выводить сообщение и заканчивать работу.

UML диаграмма.



Код программы.

Sourse.cpp

#include"Header.h"

fraction make\_fraktion(double F, double S)

{

fraction temp;

temp.Init(F,S);

return temp;

}

int main()

{

system("chcp 1251 >> null");

fraction one;

double x1, y1;

one.Write();

one.Show();

cout << "\nВведите координаты доп.точки: ";

cin >> x1 >> y1;

cout << "\nРасстояние между точками: " << one.schet(x1, y1);

return 0;

}

Header.h

#pragma once

#include <iostream>

using namespace std;

struct fraction

{

double first;

double second;

void Init(double, double);//метод для инициализации полей

void Write();//метод для чтения значений полей

void Show();//метод для вывода значений полей

double schet(double x1, double y1);//вычисление расстояния

};

Header.cpp

#include"Header.h"

void fraction::Init(double F, double S)

{

first = F;

second = S;

}

void fraction::Write()

{

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

cout << "\nsecond = "; cin >> second;

}

void fraction::Show()

{

cout << "\nfirst = " << first;

cout << "\nsecond = " << second;

cout << "\n";

}

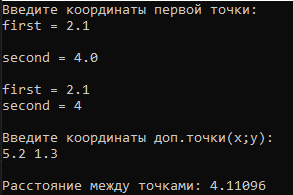
double fraction::schet(double x1, double y1)

{

return (sqrt(pow(x1 - first, 2) + pow(y1 - second, 2)));

}

Вывод.



Контрольные вопросы

1. Что такое класс?

Класс - абстрактный тип данных, определяемым пользователем.

Представляет собой модель реального объекта в виде данных и функций для работы с ними.

1. Что такое объект (экземпляр) класса?

Экземпляр класса – это описание конкретного объекта класса. Допустим:

Класс – человек, объект – Данил.

1. Как называются поля класса?

Атрибуты.

1. Как называются функции класса?

Методы.

1. Для чего используются спецификаторы доступа?

Управление видимостью элементов класса (чтение/запись).

1. Для чего используется спецификатор public?

Для общедоступных элементов класса

1. Для чего используется спецификатор private?

Для закрытых элементов класса.

1. Если описание класса начинается со спецификатора class, то

какой спецификатор доступа будет использоваться по умолчанию?

Private

1. Если описание класса начинается со спецификатора struct, то

какой спецификатор доступа будет использоваться по умолчанию?

Public

1. Какой спецификатор доступа должен использоваться при

описании интерфейса класса? Почему?

Public. Так как нужно иметь доступ к методу, чтобы его описать в

дальнейшем.

1. Каким образом можно изменить значения атрибутов экземпляра

класса?

С помощью метода – модификатора (сеттер).

1. Каким образом можно получить значения атрибутов экземпляра

класса?

С помощью метода – селектор (геттер).

1. struct Student

{

string name;

int group;

………

};

Объект класса определен следующим образом

Student \*s=new Student;

Как можно обратиться к полю name объекта s?

s->name;

1. struct Student

{

string name;

int group;

……..

};

Объект класса определен следующим образом

Student s;

Как можно обратиться к полю name объекта s?

s.name;

1. class Student

{

string name;

int group;

…….

};

Объект класса определен следующим

образом Student \*s=new Student;

Как можно обратиться к полю name объекта s?

с помощью геттера

1. class Student

{

string name;

int group;

public:

Объект класса определен следующим

образом Student s;

Как можно обратиться к полю name объекта s?

с помощью геттера

class Student

{

public:

char\* name;

int group;

………

};

Объект класса определен следующим образом

Student \*s=new Student;

Как можно обратиться к полю name объекта s?

s->name.