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 ПО ЛАБОРАТОРНОЙ РАБОТЕ №1**

Дисциплина: «Основы алгоритмизации и программирования»

Тема : "Классы и объекты. Инкапсуляция."

Выполнила работу

Студентка группы РИС-22-1Б

Лихачев Д.А.

Проверила

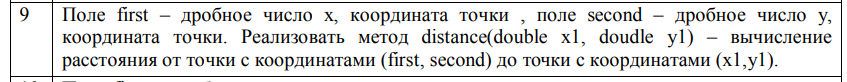
Доцент кафедры ИТАС

Полякова О.А.

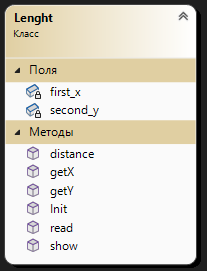
Г. Пермь-2023

**Постановка задачи**

1. Реализовать определение нового класса. Для демонстрации работы с объектами написать главную функцию. Продемонстрировать разные способы создания объектов и массивов объектов.  
2. Структура-пара - структура с двумя полями, которые имеют имена first и second. Требуется реализовать тип данных с помощью такой структуры. Во всех заданиях должны присутствовать:  
a. метод инициализации Init (метод должен контролировать значения  
аргументов на корректность);  
b. Ввод с клавиатуры Read;  
с. Вывод на экран Show.  
3. 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 класса**



**Код программы**

Lenght.h:

#pragma once

class Lenght

{

private:

double first\_x;

double second\_y;

public:

double getX() {

return this->first\_x;

}

double getY() {

return this->second\_y;

}

double distance(double x, double y);

void Init(double, double);

void read();

void show();

};

Lenght.cpp:

#include <iostream>

#include <math.h>

#include "Lenght.h"

using namespace std;

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

first\_x = F;

second\_y = S;

}

void Lenght::read() {

cout << "Enter X";

cin >> first\_x;

cout << "Enter Y";

cin >> second\_y;

}

void Lenght::show() {

cout << "\nX = " << first\_x << "\nY = " << second\_y;

cout << endl;

}

double Lenght::distance(double x, double y) {

return pow(pow(x-first\_x,2)+pow(y-second\_y,2),0.5);

}

Lab1\_main.cpp:

#include <iostream>

#include "Lenght.h"

using namespace std;

Lenght make\_object(double F, double S) {

Lenght t;

t.Init(F, S);

return t;

}

int main()

{

double first, second;

cout << "Enter x coordinate for first object: "; cin >> first;

cout << "Enter y coordinate for first object: "; cin >> second;

Lenght p = make\_object(first, second);

cout << "Enter x coordinate for second object: "; cin >> first;

cout << "Enter y coordinate for second object: "; cin >> second;

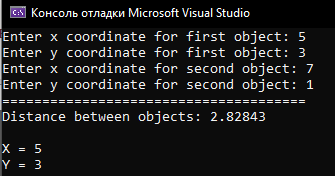
cout << "======================================"<<endl;

cout << "Distance between objects: " << p.distance(first, second)<<endl;

p.show();

}

**Результат работы программы**



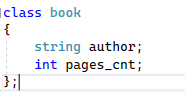
Введя координаты, пользователь получит ответ - расстояние между точками с введенными им коорданатами.

**Ответы на контрольные вопросы.**

1. Что такое класс?

Класс – это абстрактный тип данных, определяемый пользователем. Представляет собой модель реального объекта в виде данных и функций для работы с ними.

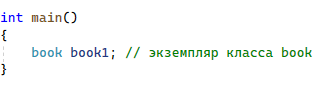
Пример:



2. Что такое объект (экземпляр) класса?

Объект (экземпляр) класса - это переменная типа класса. Их может быть бесконечное количество.

Пример:

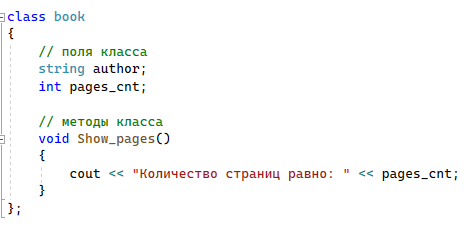


3. Как называются поля класса?

Поля - переменные внутри класса, также называются атрибутами класса.

4. Как называются функции класса?

Функции класса называются методами класса.

  
5. Для чего используются спецификаторы доступа?

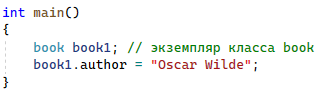
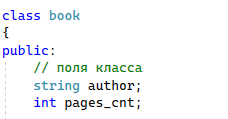
Спецификаторы доступа используются для изменения видимости компонентов.

Всего есть 3 спецификатора public, private, protected.

В структуре (struct) все поля и методы по умолчанию имеют модификатор доступа public, а в классе (class) все поля и методы по умолчанию имеют модификатор доступа private.

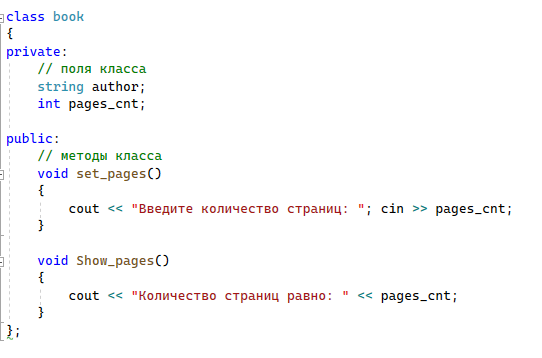
6. Для чего используется спецификатор public?

При спецификаторе доступа public общедоступные компоненты доступны в любой части программы. Они могут быть использованы любой функцией как внутри класса, так и вне его. Доступ из вне осуществляется через имя объекта.



7. Для чего используется спецификатор private?

При спецификаторе доступа private собственные компоненты локализованы в классе и не доступны извне. Они могут использоваться функциями - членами данного класса и функциями - "друзьями" того класса, в котором они описаны.



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

По умолчанию будет исполльзоваться private.

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

По умолчанию будет использоваться public.

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

11. Каким образом можно изменить значения атрибутов экземпляра класса?

1. Напрямую

2. Через ссылку

3. Через указатель

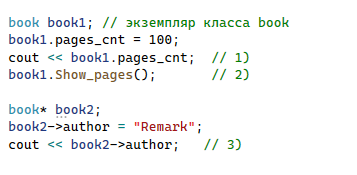


12. Каким образом можно получить значения атрибутов экземпляра класса?

1. Напрямую

2. Через public методы классса

3. Через указатель.



13. Класс описан следующим образом  
struct Student  
{  
string name;  
int group;  
};  
Объект класса определен следующим образом Student \*s=new Student;  
Как можно обратиться к полю name объекта s?

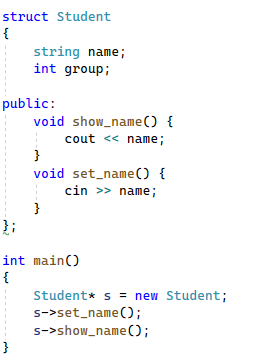


14. Класс описан следующим образом struct Student  
{  
string name;  
int group;  
};  
Объект класса определен следующим образом Student s;  
Как можно обратиться к полю name объекта s?

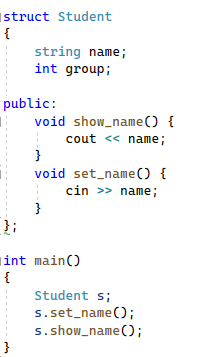


15. Класс описан следующим образом  
class Student  
{  
string name;  
int group;  
};  
Объект класса определен следующим образом Student \*s=new Student;  
Как можно обратиться к полю name объекта s?

Через public методы:

  
16. Класс описан следующим образом  
class Student  
{  
string name;  
int group; public:

};  
Объект класса определен следующим образом Student s;  
Как можно обратиться к полю name объекта s?

  
17. Класс описан следующим образом  
class Student  
{  
public:  
char\* name;  
int group;  
};  
Объект класса определен следующим образом Student \*s=new Student;  
Как можно обратиться к полю name объекта я?

