Федеральное государственное автономное образовательное учреждение высшего образования «Пермский национальный исследовательский

политехнический университет»

Лабораторная работа №1

«Классы и объекты. Инкапсуляция»

Выполнил:

студент первого курса

ЭТФ группы РИС-23-3б

Коротаев Александр Дмитриевич

Проверила:

Доцент кафедры ИТАС О. А. Полякова

Пермь, 2024

Классы и объекты. Инкапсуляция

**Цель задания**

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 объектов.
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 :
   1. метод инициализации Init (метод должен контролировать значения аргументов на корректность);
   2. ввод с клавиатуры Read;
   3. 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.

**Задание**

Элемент арифметической прогрессии aj вычисляется по формуле:

aj=a0·rj.Поле first – дробное число, первый элемент прогрессии a0, поле second

– положительное целое число, постоянное отношение r. Реализовать метод

element(int j) –вычисление j-го элемента прогрессии.

**Анализ задачи**

1. Вводятся значения для первого элемента прогрессии (first), коэффициента прогрессии (second) и номера элемента (j\_elem). Вводятся эти значения с помощью функций cin и cout.
2. После ввода значений создается объект класса Progression с параметрами 1 и 1. Затем значения first и second устанавливаются в объекте prog с помощью методов setFirst и setSecond.
3. Наконец, вычисляется значение элемента прогрессии с помощью метода element объекта prog и выводится на экран с помощью функции cout.
4. конструктор класса Progression. Конструктор принимает два параметра - first и second, которые присваиваются соответствующим полям объекта.
5. Затем объявляется метод element, который вычисляет j-ый элемент прогрессии по формуле aj = a0 \* r^j.
6. Далее объявляются методы setFirst и setSecond, которые устанавливают значения полей first и second соответственно.
7. директива #pragma once, которая предотвращает повторную загрузку заголовочного файла, если он уже был включен
8. класс Progression. Класс содержит два private поля - first и second,

которые являются double значениями.

1. Также объявляются четыре public метода - конструктор класса Progression, который принимает два double параметра, метод element, который принимает int параметр и возвращает double значение, и два метода setFirst и setSecond, которые принимают double параметры.

**Код**

Файл ООП1.cpp

#include <iostream>

#include "Progression.h"

using namespace std;

int main()

{

system("chcp 1251>null");

cout << "Введите a0 и r: ";

double first;

int second, index;

cin >> first >> second;

Progression progressia(first, second);

do

{

cout << "Для какого j-го элемента вывести значение функции: ";

cin >> index;

} while (index < 0);

cout << "Значение функции = " << progressia.element(index) << '\n';

return 0;

}

Progression::Progression(double first, int second)

{

this->first = first;

this->second = second;

}

double Progression::element(int j)

{

return first \* pow(second, j);

}

Файл Progression.h

#pragma once

class Progression

{

private:

double first;

unsigned int second;

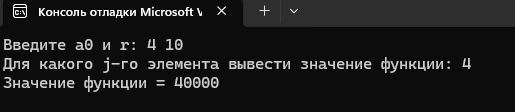
public:

Progression(double, int);

double element(int);

};

**Результат работы**



**GitHub**

<https://github.com/Korovay4ik/Laboratory-works>

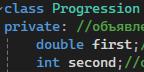
**Контрольные вопросы**

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. Он содержит данные (поля) и функции (методы), которые определяют поведение объекта.

1. **Что такое объект (экземпляр) класса?**

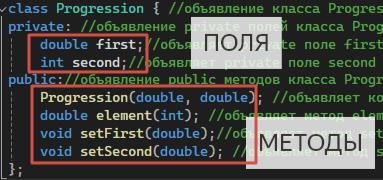
Объект (экземпляр) класса - это переменная типа класса. Их может быть бесконечное количество.



1. **Как называются поля класса?**

Поля - переменные внутри класса, также называются атрибутами класса.

Они хранят информацию, специфичную для каждого объекта класса.



1. **Как называются функции класса?**

Функции класса обычно называются методами. Они определяют поведение объекта и могут работать с его полями.

1. **Для чего используются спецификаторы доступа?**

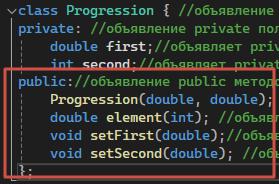
Спецификаторы доступа используются для управления доступом к членам класса. Они определяют, какие части класса могут быть доступны

извне и внутри самого класса. Всего есть 3 спецификатора public, private, protected.

* классе (class) все поля и методы по умолчанию имеют модификатор доступа private.

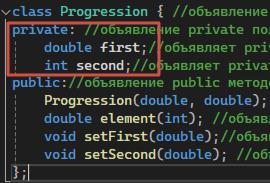
1. **Для чего используется спецификатор public?**

Спецификатор public используется для определения членов класса, которые доступны другим объектам и функциям вне класса. Это позволяет другим частям программы использовать и изменять эти члены.



1. **Для чего используется спецификатор private?**

Спецификатор private используется для определения членов класса, которые доступны только внутри самого класса. Это предотвращает несанкционированный доступ к этим членам извне класса и обеспечивает инкапсуляцию, что помогает поддерживать целостность и безопасность данных класса. Функции друзья могут использовать заприваченные поля и методы.



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

Если описание класса начинается со спецификатора class, то по умолчанию используется спецификатор доступа private. Это означает, что все члены класса будут закрытыми, если явно не указать другой спецификатор доступа.

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

Если описание класса начинается со спецификатора struct, то по умолчанию используется спецификатор доступа public. Все члены класса будут общедоступными, если явно не указать другой спецификатор доступа.

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

При описании интерфейса класса следует использовать спецификатор доступа public. Интерфейс класса определяет публичные методы и свойства, которые должны быть доступны для использования другими частями программы. Использование private ограничило бы доступ к этим членам, что противоречит цели интерфейса.

1. **Каким образом можно изменить значения атрибутов экземпляра класса?**

Напрямую, через ссылку, через указатель, через методы классы (сеттеры)

1. **Каким образом можно получить значения атрибутов экземпляра класса?**

Напрямую, через указатель, через методы классы (геттеры)

1. **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?**

s->name = “Olga”

1. **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?**

s.name = “Olga”

1. **Класс описан следующим образом class Student**

**{**

**string name; int group;**

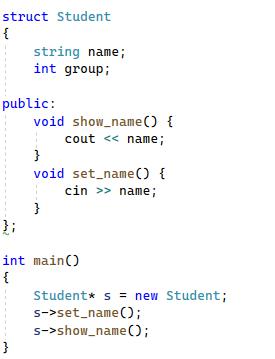
**…….**

**};**

**Объект класса определен следующим образом Student \*s=new Student;**

**Как можно обратиться к полю name объекта s?**

s->name = “Olga”



1. **Класс описан следующим образом class Student**

**{**

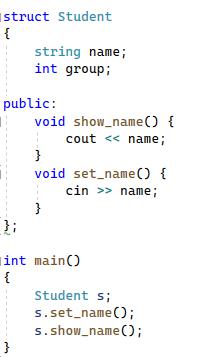
**string name; int group; public:**

**…..**

**};**

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

s.name = “Olga”



1. **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 s?**

s->name = “Olga”