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**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: «Классы и объекты. Инкапсуляция»

Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Желнин Н.А.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

# Постановка задачи

# Реализовать определение нового класса. Для демонстрации работы с объектами написать главную функцию. Продемонстрировать разные способы создания объектов и массивов объектов.

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.

# Алгоритм решения

1. Создаем структуру para.
2. Создаем методы инициализации, ввода, вывода и умножения.
3. Создаем функцию проверки введенных данных на корректность.
4. В главной функции последовательно проверяем работоспособность всех функции.

# Код программы

*#include<iostream>*

*#include<cmath>*

*using namespace std;*

*struct para{*

*double first;*

*double second;*

*void Init(double, double);*

*void Read();*

*void Show();*

*void Multiply(int);*

*};*

*void para::Init(double f, double s){*

*if(f == round(f) and s != round(s)){*

*first = f;*

*second = s;*

*}*

*else{*

*cout << "\nError";*

*first = 0;*

*second = 0;*

*}*

*}*

*void para::Read(){*

*cout << "\nFirst: "; cin >> first;*

*cout << "\nSecond: "; cin >> second;*

*}*

*void para:: Show(){*

*cout << "\nFirst: " << first;*

*cout << "\nSecond: " << second;*

*cout << "\n";*

*}*

*void para::Multiply(int k){*

*first\*=k;*

*second\*=k;*

*}*

*para make\_para(double f, double s){*

*para tmp;*

*tmp.Init(f,s);*

*return tmp;*

*}*

*int main(){*

*para a;*

*para mas[3];*

*for(int i = 0; i < 3; i++) mas[i].Read();*

*for(int i = 0; i < 3; i++) mas[i].Show();*

*for(int i = 0; i < 3; i++) mas[i].Multiply(6);*

*para\* mas2 = new para[3];*

*for(int i = 0; i < 3; i++) mas2[i].Read();*

*for(int i = 0; i < 3; i++) mas2[i].Show();*

*for(int i = 0; i < 3; i++) mas2[i].Multiply(3);*

*a.Init(2, 3.4);*

*a.Show();*

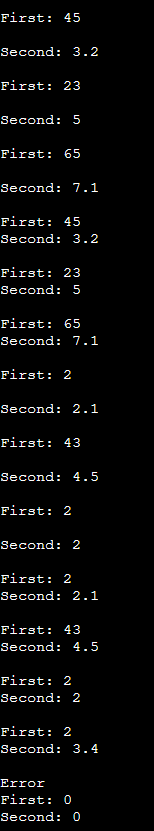
*para tmp = make\_para(4.3, 3.6);*

*tmp.Show();*

*return 0;*

*}*

# Вывод в консоль



Контрольные вопросы

1. Что такое класс?

*Это абстрактный тип данных. Он сочетает в себе два функционала: Первая — это структура, в которой можно хранить различные типы данных: массивы, переменные, функции. Вторая — возможность пользоваться объектно-ориентированным программированием.*

2. Что такое объект (экземпляр) класса?

*В языке программирования объект – это переменная типа «класс». Класс описывает данные и методы (функции), которые будут использоваться объектом этого класса. Каждый класс описывает логически-завершенную единицу программы.*

3. Как называются поля класса?

*Поле класса или атрибут в объектно-ориентированном программировании — переменная, связанная с классом или объектом.*

4. Как называются функции класса?

*Методы.*

5. Для чего используются спецификаторы доступа?

*Чтобы предотвратить изменения «жизненно важных» переменных и методов другими пользователями.*

6. Для чего используется спецификатор public?

*Для доступа к методам и полям через функцию main().*

7. Для чего используется спецификатор private?

*Для работы с этими полями и методами внутри класса.*

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

*Private.*

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

*Public.*

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

*Public, для комфортной работы из любой функции.*

11. Каким образом можно изменить значения атрибутов экземпляра класса?

*С помощью конструкторов копирования, с параметрами, без параметров, с помощью сеттеров(метод присваивания), с помощью обычной инициализации в функции main(a.number = 542562).*

12. Каким образом можно получить значения атрибутов экземпляра класса?

*С помощью методов класса show()(простой cout), либо с использованием геттеров(возвращает значение, обычно полученное от сеттера).*

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?

*s.name;*

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?

*s.name;*

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?

*С помощью сеттера void Set\_name(your\_name){name = your\_name;}, из функции main(){s.Set\_name};*

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?

*С помощью сеттера void Set\_name(your\_name){name = your\_name;}, из функции main(){s.Set\_name};*

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 s?

*s.name;*