Министерство высшего образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ

о лабораторной работе № 1

Тема: «Классы и объекты. Инкапсуляция.»

Выполнил

Студент группы РИС-22-2б

Арапов М. В.

Проверил доц. кафедры ИТАС

Полякова О.А.

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 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.

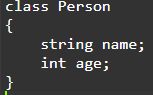
**2 вариант**

Поле first - положительное целое число, номинал купюры; номинал может принимать значения 1, 2, 5. 10, 50, 100, 500, 1000, 5000, поле second - положительное целое число, количество купюр данного достоинства. Реализовать метод summa() - вычисление денежной суммы.

**Контрольные вопросы**

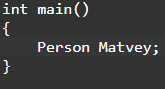
* 1. Что такое класс?

Класс - это пользовательский тип данных, который позволяет объединять данные и функции в одном объекте.



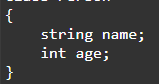
* 1. Что такое объект (экземпляр) класса?

Экземпляр класса – это конкретный экземпляр типа данных, описываемого классом.



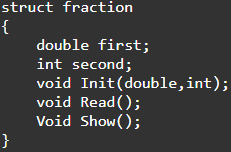
* 1. Как называются поля класса?

Атрибуты.



* 1. Как называются функции класса?

Методы.



* 1. Для чего используются спецификаторы доступа?

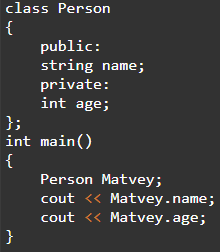
Управление видимостью элементов класса.

* 1. Для чего используется спецификатор public?

Для общедоступных элементов класса

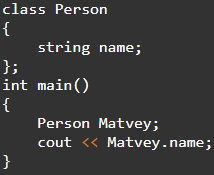
* 1. Для чего используется спецификатор private?

Для закрытых элементов класса.



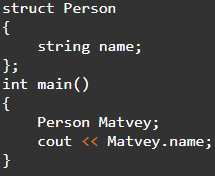
* 1. Если описание класса начинается со спецификатора class, то какой спецификатор доступа будет использоваться по умолчанию?

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?

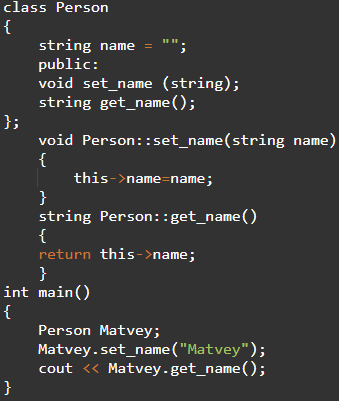
Public. Так как нужно иметь доступ к методу, чтобы его описать.

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

С помощью метода – модификатора.

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

С помощью метода – селектор.





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;



struct Student

{

string name;

int group;

……..

};

Объект класса определен следующим образом

Student s;

Как можно обратиться к полю name объекта s?

s.name;



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



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;

………

};

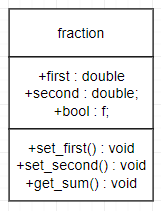
Объект класса определен следующим образом

Student \*s=new Student;

Как можно обратиться к полю name объекта s?

s->name.

**UML – диаграмма**



**Код программы**

Fraction.h:

#pragma once

#include <iostream>

using namespace std;

class Money

{

int first;

int second;

bool f;

public:

void set\_first(int);

void set\_second(int);

void get\_summ();  
}

Fraction.cpp:

#include "Class.h"

void Money::set\_first(int first)

{

this->first = first;

if (first != 1 and first != 2 and first != 5 and first != 10 and first != 50 and first != 100 and first != 1000 and first != 5000)

{

cout << "Error!" << endl;

this->f = 0;

}

}

void Money::set\_second(int second)

{

this->second = second;

}

void Money::get\_summ()

{

if (this->f != 0)

cout << "Sum = " << this->first \* this->second;

}  
Lab1:

#include "Fraction.h"

int main()

{

int x, y;

Money a;

cout << "Enter the denomination of the bill. Can be 1,2,5,10,50,100,500,1000,5000" << endl;

cin >> x;

cout << "Enter the number of bills" << endl;

cin >> y;

a.set\_first(x);

a.set\_second(y);

a.get\_summ();

return(0);

}