***Лабораторная работа 5***

***АБСТРАКТНЫЕ КЛАССЫ И ИНТЕРФЕЙСЫ***

1. Создайте абстрактный класс **Engine** с параметрами:

- double engineVolume;

- int cylinderAmount;

- double engineWeight;

Конструкторы с параметрами и конструктор по умолчанию

Геттеры и сеттеры

+ abstract double efficiency();

+ abstract double throttleEnergy();

+ abstract double breakEnergy();

+ double **getMaxSpeed(){**

        return (throttleEnergy()-breakEnergy())\*efficiency();

 }

Создайте класс **FerrariEngine** который наследует от класса **Engine**.

Конструктор с параметрами и конструктор по умолчанию

Реализуйте все абстрактные методы

В **FerrariEngine** вы вычислите абстрактные методы по формуле:

Efficiency: 0.25

Throttle energy: engineVolume\*cylinderAmount\*100

Break energy: engineWeight\*2

Создайте класс **RenaultEnigine** который наследует от класса **Engine**.

Он имеет дополнительный параметр:

- double extraTurboEnergy;

Конструктор с параметрами и конструктор по умолчанию

Реализуйте все абстрактные методы

В **RenaultEnigine** вы вычислите абстрактные методы по формуле:

Efficiency: 0.27

Throttle energy: engineVolume\*cylinderAmount\*110 + extraTurboEnergy

Break energy: engineWeight\*2.1

В основном классе Main, создайте по 5 объектов класса **FerrariEngine** и **RenaultEnigine**с разными значениями.

Добавьте все в массив класса **Engine**.

Выведите максимальные скорости каждого объекта

1. Создайте интерфейс **Workers.java** в котором есть абстрактные методы:

*String getWorkerData();*

*int getSalary();*

Получается, вы создаете несколько классов, которые наследуют от интерфейса **Workers.java:**

**Staff.java**(Рабочие)

- int id;

- String name;

- String surname;

- int salary;

**HRManagers.java**(Отдел кадров)

- int id;

- String fullName;

- int salary;

**Programmers.java** (Программисты)

- int id;

- String nickname;

- int salary;

- int bonusSalary;

- double KPIValue;

Получается у каждого класса есть свои особенности, но ваша задача реализовать абстрактные методы.

В принципе при вызове метода getSalary() у всех классов возвращается значение зарплаты, но у программистов подсчет зарплаты идет таким образом: salary + KPIValue\*bonusSalary. KPIValue - это коэффициент объема работы, выполненный за определенный период, и его значение от 0 до 1 включительно.

В основном классе Main.java создайте по 5 объектов рабочих, отдела кадров и программистов.

a. Выведите данные рабочего, у которого самая высокая зарплата.

b. Отсортируйте рабочих, по зарплате, по убыванию.

***HOMEWORK***

1. Создайте специальный интерфейс **Restaurant.java** в котором будут абстрактные методы:

public double getTotalIncome();

public double getTotalOutcome();

Эти два метода возвращают расход и доход ресторанов.

Далее, создадим специальный класс **City.java** в котором я имею список всех ресторанов моего города.

**City.java**

private Restaurant[] allRestaurants = new Restaurant[50];

private int restaurantSize = 0;

private String cityName; // *геттер и сеттер для данного поля*

public City(String cityName); // *Конструктор*

public City(); // *Конструктор по умолчанию*

public void addRestaurant(Restaurant restaurant); // *Данный метод добавляет ресторан в список*

public double getTotalTax(); //*Данный метод возвращает общую сумму разницы дохода и расхода всех ресторанов умноженную на 15 % как налог.*

Получается наш город будет реализованным. Но вот для добавления любого ресторана в наш город, мы должны реализовать наш интерфейс ресторана.

Создайте два класса **MCDonalds.java** и **KFC.java** которые реализовывают интерфейс **Restaurant.java**

**MCDonalds.java**

private int burgersPrices[] = new int[100];

private int workersSalary[] = new int[20];

private int burgersSize = 0;

private int workersSize = 0;

public void addBurger(int burgerPrice); // *Данный метод добавляет цену бургера в список*

public void addWorker(int workerSalary); // *Данный метод добавляет зарплату рабочего*

Реализуйте два абстрактных метода из интерфейса, в котором вы возвращаете расход (сумма зарплат всех рабочих) и доход (сумма всех цен на бургер)

**KFC.java**

private int basketPrices[] = new int[100];

private int basketSize = 0;

public void addBasket(int basketPrice); // *Данный метод добавляет цену баскета в список*

Реализуйте два абстрактных метода из интерфейса, в котором вы возвращаете расход (в данном примере расход равен нулю) и доход (сумма всех цен на баскет)

В главном классе создайте два объекта класса **MCDonalds** и один объект класса **KFC**.

Добавьте в каждую из них несколько бургеров, рабочих и баскетов соответственно.

Создайте объект класса **City** и назовите его Almaty.

Добавьте туда все выше созданные заведения.

Выведите общий налог от всех заведении.

1. Программно промоделировать стрельбу по мишени группой человек. Каждый человек имеет свое имя, возраст (в годах) и стаж обучения стрельбе (в годах).  
   Люди делятся на новичков, опытных и ветеранов (потомки класса человек). Для каждого человека определите полиморфный метод. «Стрелять» без параметров, возвращающих логическое значение (попал – true, не попал – false). Попадание определяется случайным образом, причем для новичка вероятность попасть равна 0,01\*стаж обучения; для опытно- го = 0,05\*стаж обучения стрельбе; для ветерана = 0,9 – 0,01\*возраст. Люди стреляют по очереди, начиная с первого, пока кто-то не попадет в мишень. Стрельба прекращается после того, как кто-то попал или все выстрелили по одному разу. После каждого выстрела нужно выводить на экран всю информацию о стреляющем и результат стрельбы.

В главной программе создайте массив из 7 людей в таком порядке: новичок, опытный, ветеран, опытный, новичок и произведите стрельбу с выводом ее результатов.