Лабораторная работа #4.
Наследование и полиморфизм

Задание 1

Создайте класс **User** с параметрами:
protected int id
protected String login
protected String password
protected String name
protected String surname

+ User()
+ User(int id, String login, String password, String name, String surname)

Геттеры и сеттеры

+String **getData()** // Данный метод возвращает все данные пользователя

Создайте 2 класса которые наследуют от класса **User:**

1 – **Staff**
private double salary;
private String **subjects[] = new String[100];**
private int indexOfSubject = 0;
Геттеры и сеттеры
Переопределите метод **getData()**, относительно их параметрам

2 – **Student**
private double gpa
private String **courses[] = new String[100];**
private int indexOfCourses = 0;
Геттеры и сеттеры
Переопределите метод **getData()**, относительно их параметрам

Подсказка*
Для массивов subjects & courses вам не нужно создавать геттеры и сеттеры, а также вам нельзя его указывать в конструкторе. Если мы укажаем массив в конструкторе - нам придется создавать массив на мейне и передавать уже готовый массив в наш конструктор, а мы этого не хотим! Нам нужен хардкор!
Что тогда делать?
Создаем специальную переменную index, которая следит за тем, сколько на самом деле было добавлено курсов/уроков. А это означает, что нужно создать методы **addSubject(String subject) & addCourse(String course),** которые будут добавлять по одному subject & course, в наши массивы, а еще будет фиксировать сколько на самом деле было добавлено курсов. Для вывода всех данных из массиовов создайте специальные методы, которые запускают цикл и выводят на экран все данные. И да, в цикле должен использоваться **index,** вдруг у вас там всего 5 книг, а не все 100!

В вашем основном классе **Main**, вы должны создать как минимум по 5 объектов класса **Student**, **Staff** и **Users**, и добавить их в массив из класса **Users**.
Чтобы добавить курсы пользователям, просто вызовите методы **addSubject() & addCourse()** для каждого объекта!

Задание 2

Создайте меню для первого задания, где вы управляете студентами, рабочими и пользователями.

PRESS [1] ADD USER

    PRESS [1] TO ADD STUDENT
    PRESS [2] TO ADD STAFF

PRESS [2] TO LIST USERS

    PRESS [1] TO LIST STUDENTS
    PRESS [2] TO LIST STAFF

PRESS [0] TO EXIT

(Подсказка: Фильтр вывода студента или рабочего нужно реализовать с помощью ключевого слова: **instanceof**)

instanceof - специальное ключевое слово, которая возвращает true, если объект является типом данного класса.

Например:

```
Dog d = new Dog();

if(d instanceof Dog){
   System.out.println("I am a Dog");
}else{
   System.out.println("I am not a Dog");
}
```

В данном примере, объект **d** является экземпляром класса **Dog**, соответственно возвращает **true**.

Домашняя работа

Задание 1

Создайте класс **Book** с параметрами:

```
protected String name;
protected String code;
protected int pages;
```

+**Book()** // Конструктор по умолчанию
+**Book(String name, String code, int pages)** // Конструкторы с параметрами

Геттеры и сеттеры

+String **getBookData()**; // Данный метод возвращает данные о книге

Создайте класс **ScientificBook** которая наследует от **Book**:

private int price;
private String publisher;

+ **ScientificBook()**
+ **ScientificBook(String name, String code, int pages, int price, String publisher)**

Геттеры и сеттеры

Переопределите метод **getBookData()**

Создайте класс **LiteratureBook** Которая наследует от класса **Book**:

private String author;
private int publishedYear;

+ **LiteratureBook()**
+ **LiteratureBook(String name, String code, int pages, String author, int publishedYear)**

Геттеры и сеттеры
Переопределите метод **getBookData()**

Создайте класс **Library** с параметрами:

private String name;
private String city;
private String country;
private **Book[] books = new Book[100]**; //Для данного массива не создавайте геттер и сеттер
private **int sizeOfBooks = 0**; //Для данного типа не создавайте геттер и сеттер

+**Library()** // Конструктор по умолчанию
+**Library(String name, String city, String country)**

Геттеры и сеттеры

+ void **addBook(Book b)** // данный метод добавляет книгу в массив используя курсор **sizeOfBooks**

+ void **printLibraryData()** //Данный метод выведет данные о библиотеке и выведет данные о каждой книге

In Main class:

Создайте объект класса Library.

Добавьте туда разных по 5 объектов класса **ScientificBook** и **LiteratureBook**.
Выведите все данные о библиотеке с ее книгами используя метод **printLibraryData()**

Задание 2

Создайте программу которая будет хранить книги в вашем библиотеке.

Программа позволит искать книги по определенным критериям:

PRESS [1] TO SEARCH BOOK BY NAME
    INSERT NAME OF THE BOOK:
PRESS [2] TO SEARCH BOOK BY CODE
    INSERT CODE OF THE BOOK:
PRESS [3] TO SEARCH BOOK BY PAGES AMOUNT
    INSERT MINIMUM AMONT OF PAGES:
    INSERT MAXIMUM AMOUNT OF PAGES: