Создайте класс Worker с полями:

```
private int id;
private String surname;
private String name;
private int age;
private int gender;
private Address address;
private Contacts contacts;
```

Добавьте конструкторы по умолчанию и со всеми параметрами. Поле id не должно передаваться в конструктор, а должно автоматически инициализироваться и увеличиваться из статического счетчика. Добавьте методы получения и установки, и метод toString().

Добавьте в класс Worker вложенный или внутренний класс Gender со статическими константами с номерами полов:

```
public static final int NOT_SPECIFIED = 0;
public static final int MALE = 1;
public static final int FEMALE = 2;
//
public static final int MIN = 0; //Минимальное значение пола
public static final int MAX = FEMALE; //Максимальное
значение пола
```

В классе Worker добавить метод, установки который будет проверять переданный пол по константам MIN и MAX из класса Gender. Если устанавливаемый пол меньше или больше минимального или максимального значений, пол должен быть установлен на NOT SPECIFIED.

Добавьте в класс Worker вложенный или внутренний класс Address с полями:

```
private String country;
private String region;
private String town;
```

```
private String house;
private String apartment;
```

Добавьте в этот класс конструктор по умолчанию, со всеми параметрами, методы получения и установки, и метод toString().

Добавьте в класс Worker вложенный или внутренний класс Contacts с полями:

```
private String mail;
private String phone;
```

Добавьте в этот класс конструктор по умолчанию, со всеми параметрами, методы получения и установки, и метод toString().

В main создайте пару объектов класса Worker, используя конструкторы со всеми параметрами. Выведете эти объекты в консоль.

2)

Создайте класс User с приватными полями:

```
String surname;
String name;
int age;
String email;
String phone;
```

Добавьте конструктор по умолчанию и с параметрами. В отдельном пакете exceptions, создайте исключение:

```
UserValidException
```

В пакете exceptions создайте исключения:

```
UserSurnameException
UserNameException
UserAgeException
UserMailException
UserPhoneException
```

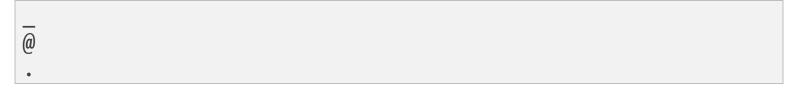
Унаследуйте эти исключения от класса UserValidException.

В классе User создайте статический метод для проверки объекта класса User на валидность с помощью Рефлексии! Метод должен содержать в сигнатуре throws UserValidException.

В методе должны проверятся все поля и если, какое-то поле не соответствует требованиям, должно выбрасываться исключение соответствующего типа. Например, если поле surname не соответствует требованиям, должен выбрасываться UserSurnameException с сообщением о проблеме с полем.

Требования к юзеру:

- поля не должны быть равны null
- имя или фамилия не должны содержать числа или специальные символы
- почта должна содержать 1 или более символов до @, 1 или более символов от собачки до точки, и не меньше двух символов после точки, почта не должна содержать специальных символов, кроме:



 номер должен содержать только цифры, начинаться с ноля и должен содержать 10 цифр.

Вложенные и Внутренние классы:

https://metanit.com/java/tutorial/3.12.php

http://developer.alexanderklimov.ru/android/java/innerclass.php

https://habr.com/ru/post/342090/

http://www.quizful.net/post/inner-classes-java

Исключения:

http://www.quizful.net/post/java-exceptions

https://metanit.com/java/tutorial/2.10.php

https://metanit.com/java/tutorial/4.1.php

https://metanit.com/java/tutorial/4.2.php

https://metanit.com/java/tutorial/4.3.php

http://iais.kemsu.ru/odocs/java/Chapter10.html