**Домашнее задание по теме «Иерархия исключений в Java. Работа программиста с исключениями»**

Формулировка задания:

1. Напишите приложение, которое будет запрашивать у пользователя следующие данные в произвольном порядке, разделенные пробелом:

Фамилия Имя Отчество датарождения номертелефона пол возраст

Форматы данных:

- фамилия, имя, отчество — строка;
- датарождения — строка формата dd.mm.yyyy;
- номертелефона - целое беззнаковое число без форматирования;
- пол - символ латиницей f или m;
- возраст — целое число.

Данные читаются из консоли.

2. Приложение должно проверить введенные данные. Если количество полей не совпадает с требуемым, вернуть код ошибки, обработать его и показать пользователю сообщение, что в файле меньше и больше данных, чем требуется.

3. Приложение должно попытаться распарсить полученные значения и выделить из них требуемые параметры. Параметры записываются в поля класса Person. Если форматы данных не совпадают, нужно бросить исключение, соответствующее типу проблемы.

Можно использовать встроенные типы java и создать свои. Исключение должно быть корректно обработано, пользователю выведено сообщение с информацией, что именно неверно.

4. Если всё введено и обработано верно, должен создаться файл с названием, равным фамилии, в него в одну строку должны записаться полученные данные, вида

<Фамилия><Имя><Отчество><датарождения> <номертелефона><пол>

5. Дополнительно. Однофамильцы должны записаться в один и тот же файл, в отдельные строки.

---

6. Не забудьте закрыть соединение с файлом. При возникновении проблемы с чтением-записью в файл, исключение должно быть корректно обработано, пользователь должен увидеть стектрейс ошибки.

7. Программа реализуется в отдельной ветке git homeworks/homework012. При сохранении состояния программы (коммиты) пишется сообщение с описанием хода работы по задаче.

В корне папки с программой должен быть файл .gitignore.

```java
package homeworks.homework12;

public interface ExceptionMessages {  1 usage  1 implementation  new *

    static final String INCORRECT_NUMBER_ARGS = "Неверное " +  1 usage
            "количество аргументов!";

    static final String INCORRECT_PHONE_NUMBER = "Недопустимый формат номера " +  1 usage
            "телефона!";

    static final String INCORRECT_GENDER = "Пол должен быть \"f\" или \"m\"!";  1 usage

    static final String INCORRECT_DATE = "Дата рождения задаётся в формате " +  1 usage
            "dd.mm.yyyy";

    static final String INCORRECT_NAME = "Имя, фамилия и отчество не могут " +  3 usages
            "содержать цифры или специальные символы!";

    static final String INCORRECT_AGE = "Возраст не соответствует " +  1 usage
            "действительности, проверьте!";
}
```

```java
package homeworks.homework12;

import java.util.Objects;

public class Person implements ExceptionMessages{ 4 usages  new *

    private String lastName; 8 usages
    private String firstName; 8 usages
    private String middleName; 8 usages
    private String date; 8 usages
    private Integer phoneNumber; 8 usages
    private char gender; 8 usages
    private Integer age; 8 usages

    public Person() { no usages  new *
    }

@   public Person(String str) { 1 usage  new *

        String[] enteredString = str.split(" ");

        if (enteredString.length != 7) {
            throw new IllegalArgumentException(INCORRECT_NUMBER_ARGS);
        }

        if (enteredString[0].chars().allMatch(Character::isLetter)) {
            this.lastName = enteredString[0];
        } else {
            throw new IllegalArgumentException(INCORRECT_NAME);
```

```
30          }
31
32          if (enteredString[1].chars().allMatch(Character::isLetter)) {
33              this.firstName = enteredString[1];
34          } else {
35              throw new IllegalArgumentException(INCORRECT_NAME);
36          }
37
38          if (enteredString[2].chars().allMatch(Character::isLetter)) {
39              this.middleName = enteredString[2];
40          } else {
41              throw new IllegalArgumentException(INCORRECT_NAME);
42          }
43
44          if (enteredString[3].matches("^((0?[1-9]|[12][0-9]|3[01])\\." +
45                  "(0?[1-9]|1[012])\\.(19|20)\\d\\d)$")) {
46
47              this.date = enteredString[3];
48          } else {
49              throw new IllegalArgumentException(INCORRECT_DATE);
50          }
51
52          if ((enteredString[4].chars().allMatch(Character::isDigit)
53                  && Integer.parseInt(enteredString[4]) > 0)) {
54
55              this.phoneNumber = Integer.parseInt(enteredString[4]);
56          } else {
```

```
            throw new IllegalArgumentException(INCORRECT_PHONE_NUMBER);
        }

        if (enteredString[5].length() == 1 && (enteredString[5].equals("f")
                || enteredString[5].equals("m"))) {

            this.gender = enteredString[5].charAt(0);
        } else {
            throw new IllegalArgumentException(INCORRECT_GENDER);
        }

        if (enteredString[6].chars().allMatch(Character::isDigit)
                && Integer.parseInt(enteredString[6]) > 0
                && Integer.parseInt(enteredString[6]) < 125) {

            this.age = Integer.parseInt(enteredString[6]);
        } else {
            throw new IllegalArgumentException(INCORRECT_AGE);
        }
    }

    public Person(String lastName, String firstName, String middleName, String date  no usages  new *
            , Integer phoneNumber, char gender, Integer age) {

        this.lastName = lastName;
        this.firstName = firstName;
        this.middleName = middleName;
```

```
84          this.date = date;
85          this.phoneNumber = phoneNumber;
86          this.gender = gender;
87          this.age = age;
88      }
89
90      public String getFirstName() {  no usages  new *
91          return firstName;
92      }
93
94      public void setFirstName(String firstName) {  no usages  new *
95          this.firstName = firstName;
96      }
97
98      public String getLastName() {  1 usage  new *
99          return lastName;
100     }
101
102     public void setLastName(String lastName) {  no usages  new *
103         this.lastName = lastName;
104     }
105
106     public String getMiddleName() {  no usages  new *
107         return middleName;
108     }
109
110     public void setMiddleName(String middleName) {  no usages  new *
```

```
111         this.middleName = middleName;
112     }
113
114     public String getDate() {  no usages  new *
115         return date;
116     }
117
118     public void setDate(String date) {  no usages  new *
119         this.date = date;
120     }
121
122     public Integer getPhoneNumber() {  no usages  new *
123         return phoneNumber;
124     }
125
126     public void setPhoneNumber(Integer phoneNumber) {  no usages  new *
127         this.phoneNumber = phoneNumber;
128     }
129
130     public char getGender() {  no usages  new *
131         return gender;
132     }
133
134     public void setGender(char gender) {  no usages  new *
135         this.gender = gender;
136     }
137
138     public Integer getAge() {  no usages  new *
```

```
139         return age;
140     }
141
142     public void setAge(Integer age) {  no usages  new *
143         this.age = age;
144     }
145
146     @Override  new *
147     public boolean equals(Object o) {
148         if (o == null || getClass() != o.getClass()) return false;
149         Person person = (Person) o;
150         return gender == person.gender && Objects.equals(firstName
151                 , person.firstName) && Objects.equals(lastName, person.lastName)
152                 && Objects.equals(middleName, person.middleName)
153                 && Objects.equals(date, person.date) && Objects.equals(phoneNumber
154                 , person.phoneNumber) && Objects.equals(age, person.age);
155     }
156
157     @Override  new *
158     public int hashCode() {
159         return Objects.hash(firstName, lastName, middleName, date, phoneNumber, gender, age);
160     }
161
162     @Override  new *
163     public String toString() {
164         return "Person{" +
165                 "firstName='" + firstName + '\'' +
166                 ", lastName='" + lastName + '\'' +
```

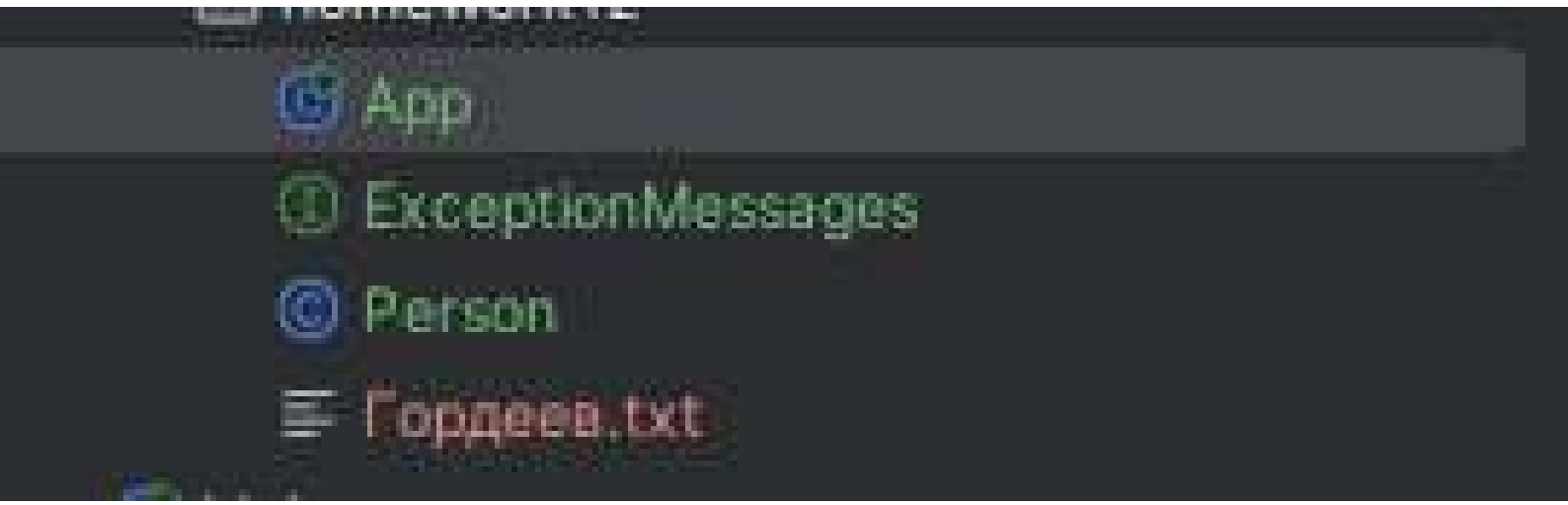
```
                ", middleName='" + middleName + '\'' +
                ", date='" + date + '\'' +
                ", phoneNumber=" + phoneNumber +
                ", gender=" + gender +
                ", age=" + age +
                '}';
    }
}
```

```java
package homeworks.homework12;

import java.io.*;
import java.util.Scanner;

public class App {
    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);
        String enteredString = scanner.nextLine();

        try {
            Person person = new Person(enteredString);

            try (BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(
                    "src/homeworks/homework12/" + person.getLastName() + ".txt"))) {

                bufferedWriter.write(enteredString);
            } catch (Exception e) {
                e.printStackTrace();
            }
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

```
"C:\Program Files\Java\jdk-21.0.2\bin\java.exe" -javaagent:G:\JetBrains\IntelliJIdea2025.1\lib\idea_rt
 .jar=59215 -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath
 D:\JavaProject\JavaDevHomeworks\out\production\JavaDevHomeworks homeworks.homework12.App
Гордеев Мак-сим Леонидович 05.04.1984 650525 m 41
Имя, фамилия и отчество не могут содержать цифры или специальные символы!
```

```
"C:\Program Files\Java\jdk-21.0.2\bin\java.exe" -javaagent:G:\JetBrains\IntelliJIdea2025.1\lib\idea_rt
 .jar=59658 -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath
 D:\JavaProject\JavaDevHomeworks\out\production\JavaDevHomeworks homeworks.homework12.App
Гордеев Максим Леонидович 05-04-1984 650525 m 41
Дата рождения задаётся в формате dd.mm.yyyy
```

```
"C:\Program Files\Java\jdk-21.0.2\bin\java.exe" -javaagent:G:\JetBrains\IntelliJIdea2025.1\lib\idea_rt
 .jar=59152 -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath
 D:\JavaProject\JavaDevHomeworks\out\production\JavaDevHomeworks homeworks.homework12.App
Гордеев Максим Леонидович 05.04.1984 562343 m 41

Process finished with exit code 0
```


App
ExceptionMessages
Person
Гордеев.txt


Person.java
App.java
Гордеев.txt
ExceptionMessages.java
1
Гордеев Максим Леонидович 05.04.1984 650525 m 41