Workshop #2: Exceptions

Upon successful completion of this workshop, you will have demonstrated the abilities to:

- Practice handling errors in your program.
- Describe to your instructor what you have learned in completing this workshop.

CQ6.1 - Write a Java program to accept a number and print out it. If the number is below 1 then a message “the number is invalid” is printed out. Using do..while to input again, try-catch block to handle errors.

The user interface may be:
Enter the number: - 2
The number is invalid
Enter the number: world
The number is invalid
Enter the number: 2
The number is 2

Step by step workshop instructions:

- Create a new project named “**workshop2**”
- In the project, create a new file named “**Part1.java**”
- In the method main, you type:

```
1
2  import java.util.Scanner;
3  public class test {
4      public static void main(String[] args) {
5          boolean cont=false;
6          do{
7              try{
8                  int num;
                   Scanner sc=new ...
10                 System.out.println("enter the number:");
                   num=sc.....
12                 if( num<1)
13                     throw new Exception();
14                 System.out.println("The number is "+ num);
15                 cont=false;
16             }catch(Exception e){
17                 System.out.println("The number is invalid");
18                 cont=true;
19             }
20         }while(cont);
21     }
22 }
```

CQ6.2 - Write a Java program to accept a string and print out it. If the string does not match SExxx (x is digit) then a message “the string is invalid” is printed out. Using do..while to input again.

Hint: In library class **String**, you should use the method **matches**() to do this, use **try-catch** block and use **throws** to handle errors.

The user interface may be:
Input the string 1: I love u
the string is invalid
Input the string 1: SE123
the string is SE123

Step by step workshop instructions:

Background: In this workshop, you will use the pattern string(also called regular expression, see more What is a Regular Expression? - Definition from Techopedia). You should read the document to complete the exercise.

Task 1: use try-catch

- In the project “**Workshop2**”, create a new file named “**Part2**.java”
- In the method main, you type:

```
1
2  import java.util.Scanner;
3  public class test {
4      public static void main(String[] args) {
5          boolean cont=false;
6          do{
7              try{
               String s="";
               String pattern=....
               Scanner sc=new ...
11             System.out.println("enter the string:");
               s=sc.....
13             if( ! s.matches( pattern))
14                 throw new Exception();
15             System.out.println("The string is "+ s);
16             cont=false;
17         }catch(Exception e){
18             System.out.println("The string is invalid");
19             cont=true;
20         }
21     }while(cont);
22 }
23 }
```

- At the row 9, use rules of the regular expression to create a pattern string “SExxx”, x is digit

Task 2: use throws keyword

- create a new file named “**Part2_2.java**”
- in the method main, type:

```
1
2  import java.util.Scanner;
3  public class Part2_2 {
4      public String inputString() throws Exception
5      {
6          String pattern="";//use rules of regular expression to c
           String s="";
8          Scanner sc=new Scanner(System.in);
9          System.out.println("input the string:");
10         s=sc.nextLine();
11         if(!s.matches(pattern))
12             throw new Exception();
13         return s;
14     }
15     public static void main(String[] args) {
16         Part2_2 obj=new Part2_2();
           boolean cont=false;
18         do{
19             try{
20                 String s=obj.inputString();
21                 System.out.println("the string is " +s);
22                 cont=false;
23             }catch(Exception e){
24                 System.out.println("The string is invalid");
25                 cont=true;
26             }
27         }while(cont);
28     }
29 }
```

CQ6.3- Write a Java program that uses the Scanner class to read and then print out a list of employees from a text file named employee.txt. If the specified pathname does not exist or the file is not accessible then a message “The system cannot find the file specified” is printed out. Using useDelimiter() to specify the delimiter, try-catch block to handle errors.

- The employee.txt file is:

```
E0001;NGUYEN HUNG DUNG;400
E0003;LUONG TAM QUANG;500
E0005;VU MINH CHAU;450
E0009;NGUYEN VAN LINH;300
E0010;NGUYEN LONG;280
```

- The user interface may be:

```
E0001, NGUYEN HUNG DUNG, 400
E0003, LUONG TAM QUANG, 500
E0005, VU MINH CHAU, 450
E0009, NGUYEN VAN LINH, 300
E0010, NGUYEN LONG, 280,
BUILD SUCCESSFUL (total time: 0 seconds)
```

Step by step workshop instructions:

- You create a new class named "**Part3.java**" and add the code:

```
7  import java.io.File;
8  import java.util.Scanner;
9
10 public class Part3 {
11     public static void main(String[] args) {
           Scanner sc = new Scanner(new File("employee.txt"));
           sc.useDelimiter ...
           while(sc.hasNext()) ... sc.next() ...
15         System.out.println("");
16     }
17 }
```

- You must insert try-catch block and add your code to get the required result.

CQ6.4 - Write a Java program to accept a vehicle and print out its information with the following additional requirements:

- VIN, model, brand, colour: String (not blank)
- engine power: int (>0)
- convertible, parking brake: boolean (TRUE or FLASE)

Note: Each vehicle is identified by a vehicle identification number (VIN). Each individual vehicle is a particular model of a particular brand offered by the company (e.g., the XF is a model of the car brand Jaguar of Tata Motors).

Hint: Using **do..while** to input again, **try-catch** block and use **throws** to handle errors.

The user interface may be:

```
    - Enter VIN:
    - Enter VIN: 12345
    - Enter model:
    - Enter model: CAMRY 2.5Q 2019
    - Enter brand:
    - Enter brand: TOYOTA
    - Enter colour:
    - Enter colour: Black
    - Enter engine power:
      The engine power is invalid!
    - Enter engine power: 0
      The engine power is invalid!
    - Enter engine power: 135
    - Enter convertible (TRUE or FALSE):
    - Enter parking brake (TRUE or FALSE): true
Vehicle information: 12345, CAMRY 2.5Q 2019, TOYOTA, Black, 135, false, true
BUILD SUCCESSFUL (total time: 3 minutes 20 seconds)
```

Step by step workshop instructions:

- You create a new class named "**Part4.java**" and add the code:

```
import java.util.Scanner;

public class Part4 {
    public static Scanner sc = new Scanner(System.in);
    public static String inputNonBlankStr(String msg) {...8 lines }
    public static int inputInt(String msg) throws Exception {...9 lines }
    public static boolean inputBoolean(String msg) {...5 lines }
    public static void main(String[] args) {
        String vin = inputNonBlankStr("   - Enter VIN: ");
        String model = inputNonBlankStr("   - Enter model: ");
        String brand = inputNonBlankStr("   - Enter brand: ");
        String colour = inputNonBlankStr("   - Enter colour: ");
        int enginePower = 0; boolean cont = false;
        do {
            try {
                enginePower = inputInt("   - Enter engine power: "); cont = false;
            } catch (Exception ex) {
                System.out.println("     The engine power is invalid!"); cont = true;
            }
        } while (cont);
        boolean convertible = inputBoolean("   - Enter convertible (TRUE or FALSE): ");
        boolean parkingBrake = inputBoolean("   - Enter parking brake (TRUE or FALSE): ");
        System.out.print("Vehicle information: ");
//        ...
    }
}
```

CQ6.5 - Details of a student including code, name, gender, address, phone, email with the following additional requirements:

- code, name, gender, address, phone, email are not allowed null,
- phone is number string which has length from 10 to 12,
- the code format is SExxxxxx (x is digit),

- the format of email is w+@w+[.w]+ (w: any number or letter, +: number of occurrences >= 1),
- input value of gender is 1 or 0 (if value is 1: gender=”male”, value is 0: gender=”female”).

Task 1: Construct the class Validations containing input validation methods: inputNonBlankStr(String msg), inputPattern(String msg, String pattern), inputGender(String msg, String pattern) for the Student:

- **inputNonBlankStr**(String msg)
  Accepts a string, converts it to uppercase, and returns. if input is blank then use **do..while** to input again.
- **inputPattern**(String msg, String pattern)
  Checks whether str matches pattern or not. Use **String.matches** (regEx) to implement and **do..while** to input again.
- **inputGender**(String msg, String pattern)
  Checks whether str matches pattern or not. Set gender="male" if input is 1, gender="female" if input is 0 and return. Other throws out an exception.

Task 2: Develop the class StudentDemo in which a main method is implemented.

The user interface may be:

```
Input Student's information:
   - Enter code (SExxxxxx) :
   - Enter code (SExxxxxx) : SE136
   - Enter code (SExxxxxx) : SE123456
   - Enter name :
   - Enter name : nguyen tien luan
   - Enter gender (1 Or 0) :
     male: input is 1, female: input is 0
   - Enter gender (1 Or 0) : male
     male: input is 1, female: input is 0
   - Enter gender (1 Or 0) : 1
   - Enter address :
   - Enter address : binh thanh
   - Enter phone (10 to 12) :
   - Enter phone (10 to 12) : 012345678
   - Enter phone (10 to 12) : 01234567890
   - Enter email :
   - Enter email : LuanNTSE123456$fpt.edu.vn
   - Enter email : LuanNTSE123456@fpt.edu.vn
Student information:
SE123456, NGUYEN TIEN LUAN, male, BINH THANH, 01234567890, LuanNTSE123456@fpt.edu.vn
BUILD SUCCESSFUL (total time: 3 minutes 30 seconds)
```

Step by step workshop instructions:

- You create a new class named “**Validations.java**” and add the code:

```java
package student;

import java.util.Scanner;

public class Validations {

    public static Scanner sc = new Scanner(System.in);

    //Accepts a string, converts it to uppercase, and returns. if input is
    //blank then use do..while to input again.
    public static String inputNonBlankStr(String msg) {...8 lines }

    //Checks whether str matches pattern or not. Use String.matches (regEx) to
    //implement and do..while to input again.
    public static String inputPattern(String msg, String pattern) {...9 lines }

    //Checks whether str matches pattern or not. Set gender="male" if input is 1,
    //gender="female" if input is 0 and return. Other throws out an exception.
    public static String inputGender(String msg, String pattern) throws Exception {...10 lines }

}
```

- You create a new class named "**StudentDemo.java**" and add the code:

```java
package student;

public class StudentDemo {
    public static void main(String[] args) {
        System.out.println("Input Student's information: ");
        String code = Validations.inputPattern("   - Enter code (SExxxxxx) ", "^SE\\d{6}");
        String name = Validations.inputNonBlankStr("   - Enter name ");
        String gender = ""; boolean cont = false;
        do {
            try {
                gender = Validations.inputGender("   - Enter gender (1 Or 0) ", "[0,1]{1}");
                cont = false;
            } catch (Exception ex) {
                System.out.println("     male: input is 1, female: input is 0");
                cont = true;
            }
        } while (cont);
        String address = Validations.inputNonBlankStr("   - Enter address ");
        String phone = Validations.inputPattern("   - Enter phone (10 to 12) ", "[0-9]{10,12}");
        String email = Validations.inputPattern("   - Enter email ", "\\w+@\\w+[\\.\\w]+");
        System.out.println("Student information: ");
        System.out.println(code + ", " + name + ", " + gender + ", " + address
        + ", " + phone + ", " + email);
    }

}
```