Object-Oriented Programming Lab#8, Fall 2019

Today's Topics- "*Exception*"

Problem 1:

1. Write the following Java application.

```java
import java.util .*;
public class TestException {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter 2 integers.");
        int a = scan.nextInt();
        int b = Integer.parseInt(scan.nextLine().trim());
        int c = a/b;
        System.out.println("Result: " + c);

        scan.close();
    }
}
```

Now run the application 4 times with the following input and save the screenshot of output.

Run1: 6, 3
Run2: 6, a
Run3: a, 6
Run4: 6, 0

2. Add appropriate try/catch to handle the exceptions of problem 1.

Problem 2:

3. Write the following Java application and update main as below.

    a. Call ***throwException(105)*** and run it. Take the screen shot of the output and save it.
    b. Update the parameter to -5 and run it. Take the screen shot of the output and save it.
    c. Call ***throwException*** again with 200 as parameter. Add appropriate try/catch and print the message.

```java
public class TestException {
    public static void main(String[] args) {
        // follow instruction above
    }

public static void throwException(int n) throws InvalidParameterException, IOException{
    Integer.parseInt("2");
    if (n<=0){
        // throw InvalidParameterException with message set to "Parameter must be greater
    than 0.";
    }
```

```
    if (n>100){
       // throw IOException with message set to "Parameter must be smaller than 100.";

    }

    System.out.println(n*n);
    }
}
```

Problem 3(Project Related):

4. Update the **BankAccount** project and throw exceptions for invalid inputs.
    a. Update **withdraw**() to throw **InsufficientBalanceException**
        i. In ***BankAccount*** class, throw the InsufficientBalanceException (use `InsufficientBalanceException(double amt)` constructor) if withdraw amount results the balance to go below **minimumBalance**. Pass (accountBalance - minimumBalance) as the parameter of ***InsufficientBalanceException*** constructor.

        ii. In ***SavingAccount*** class, throw the InsufficientBalanceException (use `InsufficientBalanceException(double amt)` constructor) if withdraw amount is more than maximum withdraw limit. Pass maximum withdraw limit as the parameter of ***InsufficientBalanceException*** constructor.

    b. In **Bank** class, do not handle the ***InsufficientBalanceException*** in **withdraw()** method, just pass it in the method header.
    c. In **main** method, handle the ***InsufficientBalanceException*** and show a pop-up message with appropriate message.

Code of InsufficientBalanceException:

```
public class InsufficientBalanceException extends Exception {

	public InsufficientBalanceException(String message) {
		super(message);
	}

	public InsufficientBalanceException(double amt) {
		super("Can't withdraw more than " + amt + " taka.");
	}
}
```