

CSC 1109 LAB 7

Woon Jun Wei, 2200624

February 14, 2023

Question 1

Listing 1: CircleWithException.java

```
1 import java.util.Scanner;
2
3 public class CircleWithException {
4     private double radius;
5
6     public CircleWithException(double radius) throws ↵
7         IllegalArgumentException {
8         if (radius < 0) {
9             throw new IllegalArgumentException("Radius cannot be negative"↵
10             );
11         }
12         this.radius = radius;
13     }
14
15     public double getRadius() {
16         return radius;
17     }
18
19     public void setRadius(double radius) throws IllegalArgumentException {
20         if (radius < 0) {
21             throw new IllegalArgumentException("Radius cannot be negative"↵
22             );
23         }
24         this.radius = radius;
25     }
26
27     public double getArea() throws Exception{
28         double area = Math.PI * radius * radius;
29
30         if (area > 1000){
31             throw new Exception("Area cannot be greater than 1000");
32         }
33         return area;
34     }
35 }
```

```

33     // Exceptions not stated in Lab Instructions
34     public double getDiameter() {
35         return radius * 2;
36     }
37
38     public static void main(String[] args) {
39         CircleWithException c1 = new CircleWithException(5);
40         Scanner sc = new Scanner(System.in);
41         System.out.print("Enter a radius: ");
42         double radius = sc.nextDouble();
43         try {
44             CircleWithException c2 = new CircleWithException(radius);
45             System.out.println("Radius: " + c2.getRadius());
46             System.out.println("Area: " + c2.getArea());
47             System.out.println("Diameter: " + c2.getDiameter());
48         } catch (IllegalArgumentException e) {
49             System.out.println(e.getMessage());
50         } catch (Exception e) {
51             System.out.println(e.getMessage());
52         }
53     }
54
55 }

```

Listing 2: CircleWithException.java Output

```

1  // Negative radius
2  Enter a radius: -10
3  Radius cannot be negative
4
5  // Area Greater than 1000
6  Enter a radius: 1000
7  Radius: 1000.0
8  Area cannot be greater than 1000
9
10 // Valid Circle
11 Enter a radius: 5
12 Radius: 5.0
13 Area: 78.53981633974483
14 Diameter: 10.0

```

Question 2

Listing 3: InsufficientFundsException.java

```
1 public class InsufficientFundsException extends Exception{
2     private double amount;
3     public InsufficientFundsException(double amount){
4         this.amount = amount;
5     }
6     public double getAmount(){
7         return amount;
8     }
9 }
```

Listing 4: CheckingAccount.java

```
1 public class CheckingAccount extends InsufficientFundsException{
2
3     private double balance;
4     private int number;
5
6     public CheckingAccount(double amount) {
7         super(amount);
8     }
9
10    public CheckingAccount(){
11        this(0);
12    }
13
14    public void deposit(double amount){
15        if (amount > 0){
16            this.balance += amount;
17        }
18    }
19
20    public void withdraw(double amount) throws InsufficientFundsException{
21        if (amount > this.balance){
22            throw new InsufficientFundsException(amount);
23        }
24    }
25    public double getBalance(){
26        return this.balance;
27    }
28
29    public int getNumber(){
30        return this.number;
31    }
32 }
```

```
31     }
32
33
34 }
```

Listing 5: BankDemoTest.java

```
1  import java.util.Scanner;
2  public class BankDemoTest {
3      public static void main(String[] args){
4          CheckingAccount c = new CheckingAccount();
5          Scanner sc = new Scanner(System.in);
6
7          System.out.print("Enter an amount to deposit: ");
8          double amount = sc.nextDouble();
9          c.deposit(amount);
10         System.out.print("Enter an amount to withdraw: ");
11         amount = sc.nextDouble();
12         try{
13             c.withdraw(amount);
14         }catch(InsufficientFundsException e){
15             if (e.getAmount() > c.getBalance()){
16                 System.out.println("Sorry, but your account is short by: $↵
17                                     " + (e.getAmount() - c.getBalance()));
18             }
19         }
20         if (amount <= c.getBalance())
21             System.out.println("The balance after withdrawal is: $" + (c.↵
22                                 getBalance() - amount));
21     }
22 }
```

Listing 6: BankDemoTest.java Output

```
1  // $yy < $xx
2  Enter an amount to deposit: 100
3  Enter an amount to withdraw: 20
4  The balance after withdrawal is: $80.0
5
6  // $yy > $xx
7  Enter an amount to deposit: 100
8  Enter an amount to withdraw: 254
9  Sorry, but your account is short by: $154.0
```
