CSCD211 Lab – Exception Handling

Using Java exception handling.

Key in the following Java program that will use exception handling to observe random number generation:

```
import java.util.Random;
public class ExceptionTest1
{
    public static void main(String[] args)
    {
        int a = 0, b = 0, c = 0;
        int good = 0, bad = 0;
        Random r = new Random();

        for(int i = 0; i < 32000; i++)
        {
            try
            {
                b = r.nextInt();
                c = r.nextInt();
                a = 12345 / (b/c);
                System.out.println("OK at " + i + "  b: " +
                        b + "   c: " + c);
                good++;
            }
            catch (ArithmeticException e)
            {
                System.out.println("/ by zero at " + i + "  b: " +
                        b + "   c: " + c);
                bad++;
            }
        }
        System.out.println("Good: " + good + "  Bad: " + bad);
    }
}
```

Run the program three times, noting the number of good and bad results:

1a. Number of occurrences:

1. Good_____ Divide by zero_____
2. Good_____ Divide by zero_____
3. Good_____ Divide by zero_____

1b. What is the deciding factor for division by zero? _____
_____

---

2. Create a new function called throwOne() that will throw an arithmeticException:

```
private static void throwOne()
{
     System.out.println("Begin throwOne.");
     if (3 == 3)
          throw new ArithmeticException("demo");
     System.out.println("End throwOne.");
}
```

Comment out the existing content of main() and add a call to throwOne() plus a call to s.o.p to indicate a normal end-of-job:

```
public static void main(String[] args)
{
     throwOne();
     System.out.println("Exiting normally...");
}
```

2a. Run your program. What is the output? _____
_____
_____

Add a try-catch block to surround the call to throwOne():

```
public static void main(String[] args)
{
    try
    {
        throwOne();
    }
    catch (Exception e)
    {
        System.out.println(e);
    }
    System.out.println("Exiting...");
}
```

2b. Run your program again. Now what is the output? _____
_____
_____

---

3. Change the exception that is thrown in throwOne() to `IllegalAccessException`:

```
private static void throwOne()
{
    System.out.println("Begin throwOne.");
    if (3 == 3)
        throw new IllegalAccessException("demo");
    System.out.println("End throwOne.");
}
```

3a. Try to compile your program. Why do you get an error? _____
_____

3b. How would you fix the program's problem? _____
_____

3c. What is your program's output after you've fixed the problem? _____
_____