Week 11 Lab – **Throwing Exceptions**

Maximum Points = 10

File *Factorials.java* contains a program that calls the *factorial* method of the *MathUtils* class to compute the factorials of integers entered by the user. Save these files to your directory and study the code in both, then compile and run Factorials to see how it works. Try several positive integers, and then try a negative number. You should find that it works for small positive integers (values < 17), but that it returns a large negative value for larger integers and that it always returns 1 for negative integers.

1. Returning 1 as the factorial of any negative integer is not correct—mathematically, the factorial function is not defined for negative integers. To correct this, you could modify your *factorial* method to check if the argument is negative, but then what? The method must return a value, and even if it prints an error message, whatever value is returned could be misconstrued. Instead it should throw an exception indicating that something went wrong so it could not complete its calculation. You could define your own exception class, but there is already an exception appropriate for this situation—IllegalArgumentException, which extends RuntimeException. Modify your program as follows:

    a) Modify the header of the *factorial* method to indicate that *factorial* can throw an IllegalArgumentException.
    b) Modify the body of *factorial* to check the value of the argument and, if it is negative, throw an IllegalArgumentException. Note that what you pass to *throw* is actually an instance of the IllegalArgumentException class, and that the constructor takes a String parameter. Use this parameter to be specific about what the problem is.
    c) Compile and run your Factorials program after making these changes. Now when you enter a negative number an exception will be thrown, terminating the program. The program ends because the exception is not caught, so it is thrown by the main method, causing a runtime error.
    d) Modify the main method in your Factorials class to catch the exception thrown by factorial and print an appropriate message, but then continue with the loop. Think carefully about where you will need to put the *try* and *catch*.

2. Returning a negative number for values over 16 also is not correct. The problem is arithmetic overflow—the factorial is bigger than can be represented by an int. This can also be thought of as an IllegalArgumentException—this *factorial* method is only defined for arguments up to 16. Modify your code in *factorial* to check for an argument over 16 as well as for a negative argument. You should throw an IllegalArgumentException in either case, but pass different messages to the constructor so that the problem is clear.

```
// ****************************************************************
// Factorials.java
//
// Reads integers from the user and prints the factorial of each.
//
// ****************************************************************
import java.util.Scanner;

public class Factorials
{
    public static void main(String[] args)
    {
      String keepGoing = "y";
      Scanner scan = new Scanner(System.in);

      while (keepGoing.equals("y") || keepGoing.equals("Y"))
          {
            System.out.print("Enter an integer: ");
            int val = scan.nextInt();
            System.out.println("Factorial(" + val + ") = "
                               + MathUtils.factorial(val));
            System.out.print("Another factorial? (y/n) ");
            keepGoing = scan.next();
          }
    }
}

// ****************************************************************
// MathUtils.java
//
// Provides static mathematical utility functions.
//
// ****************************************************************
public class MathUtils
{
    //-------------------------------------------------------------
    // Returns the factorial of the argument given
    //-------------------------------------------------------------
    public static int factorial(int n)
    {
      int fac = 1;
      for (int i=n; i>0; i--)
          fac *= i;
      return fac;
    }
}
```

(Due before end of the day on Friday, October 29, 2010) Submit your .java files containing your program to the dropbox in WebCT.

Grades are determined using the following scale:

- Runs correctly.......................:___/3
- Correct output.........................:___/2
- Design of output....................:___/1
- Design of logic.....................:___/2
- Standards............................:___/1
- Documentation.....................:___/1