CSE 11 **START EARLY!**
Spring 2013
Homework Assignment #3

Due: 8 November 2012, 11:00pm
Covers Chapters: 11, 13.

You will turn in your assignment electronically
Include code – means turn in a printed copy of your code.
Include output – means turn in screen grab of the output of your code.

You will turn in several files for this assignment:

```
PR5.txt
PR5.pdf
Integrate.java
MultTable.java
```

Bundle your homework with the program

```
$ /home/linux/ieng6/cs11e/public/bin/bundleP5
```

For your screen output, create a file called PR5.pdf. It should be portable document format file (See instructions for homework 3, Please make sure to include your

NAME:
LOGIN:
ID:

in every file you turn in. PR5.txt should have just that information in it.

Problem #1 (20 points)

Using the standard Java Scanner class.

Look at http://natch3z.blogspot.com/2008/11/read-text-file-using-javautilscanner.html as an example of using the Scanner class and opening a file for reading. Modify the TextApplet example on page 296 in your book to do the following:

1. Change the applet to a program using the usual technique.
2. The text input box should be relabeled "Filename"
3. When you type a valid filename into the box and hit enter on your keyboard, your program should read the file indicated and display its contents in the JTextArea (scrollable text area next to "Output") using Scanner.
4. If the file is not found (this is what is called "throwing an exception"), the JTextArea should clear the contents and display "File Not Found!". This clear and display logic would replace the e.printStackTrace() in the blogspot example. Exceptions have not yet been covered in class, you should simply mimic the `try { } except { }` logic of the blogSpot example.
5. Pressing the "Clear Output" button should clear the JTextArea.

Include code
Include screen output when what is displayed is the java source code that answers this problem (i.e. your program)
Include screen output when a File is not Found!

Problem #2 (10 points)

Modify your code from Problem #1 to (It is highly suggested to make it a different Program)

1. Add an additional button labeled "Count" placed next to the "clear output" button
2. Add an additional JLabel That will display the number of characters of the text that is currently displayed in the JTextArea. This label should be updated when the "Count" button is pressed by the user. You should use a panel with a FlowLayout for the buttons and label.

Include code

Include screen output when what is displayed is the java source code that answers this problem (i.e. your program). Your screenshot should be taken after the "Count" button has been pressed

Problem #3 (10 points)

JAEA Exercise 13.6.1

Modify so that the corrected `isPrime` method is invoked as part of main program called `Prime`. `Prime` takes an integer argument from the command line and properly determines if a number is prime.

Include code
Include output of running Prime with input values of 17, 27, 37, 47, 57, 81

Problem #4 (10 points) Fun with for loops
In each of the following, create a valid `for( ; ; )` statement to achieve the desired results of the following two-line code segment:

```
for ( ;  ; )
      System.out.println(i);
```

example: print out integers from 1 to 10
ans: `for (i = 1; i <= 10; i++)`

(a) print out integers from 10 to 1
(b) print out 2^k k=0,1,2,3, ... 20
(c) print out every seventh integer starting at 14 ending at 98
(d) print out the numbers from 1 to 41 that are NOT evenly divisible by 3
(e) print out the numbers 1,3,6,10,15,21,28,36,45,55 (Hint: `int j = i = 1;` is a legal statement that initializes both i and j to 1)

Problem #5 (20 points) Trapezoidal Integration (calculating the area under a curve)
In basic calculus, definite integration computes the area under the curve of a function, f(x), when

graphed on the interval of A,B. One way of numerically computing the definite integral is to *sum* the area of all trapezoids defined in a particular way. The following picture shows the area of just one of these trapezoids (if you have had calculus, this should look familiar to you).

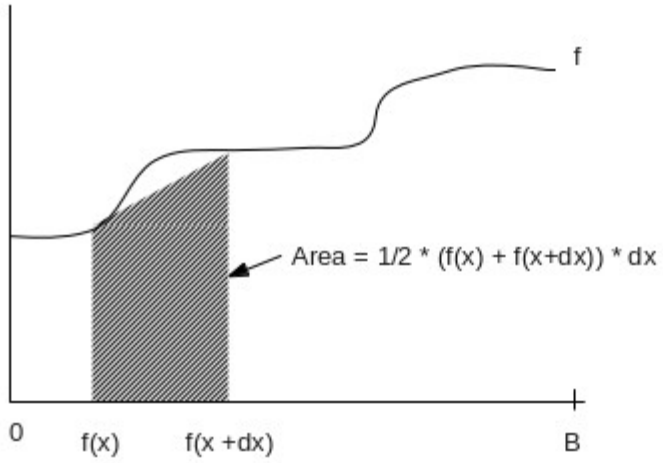

Illustration 1: Area of ONE trapezoid under the curve of a function. The integral sums from 0 to B sums all Trapezoidal areas when f is evaluated a 0,dx,2dx,3dx, ...B

If A(x) is defined as the trapezoidal area of A(x) = 1/2 (f(x) + f(x + dx)) * dx then the integral from 0 to B is A(0) + A(dx) + A(2*dx) + A(3*dx) + ... + A(B – dx)

As dx becomes *smaller* this numerical estimate becomes *better* at computing the actual integral.

Write a `Function` class that provides a static double method called `eval(double x)` that returns returns the value of the Function evaluated at x (E.g. Function(x))

Write a java program called `Integrate` that takes two command-line arguments B and nTrapezoid. B is a double the represents the right hand side of the integral, nTrapezoid is an integer that is the number of trapezoids to compute on the interval [0,B]. It should compute the integral of function defined in the class `Function` It should print out just the computed integral as a double. The basic idea is that one can redefine the function in the Function class and integrate will perform the definite integral over that function

Use your program and class to compute the integrals of

- $f(x) = x^2 + 3$, B = 1,nTrapezoid = 1
- $f(x) = x^2 + 3$, B = 1,nTrapezoid = 10
- $f(x) = x^2 + 3$, B = 1,nTrapezoid = 100
- $f(x) = -x^2 + x$, B = 1,nTrapezoid = 1
- $f(x) = -x^2 + x$, B = 1,nTrapezoid = 10
- $f(x) = -x^2 + x$, B = 1,nTrapezoid = 100

Turnin: `Integrate.java`

Problem #6 (10 points)

(a) what is the fundamental difference between `do {... } while();` and a `while() { }` loops?

(b) JAEA Exercise 13.6.4

(c) Look up the definition of the `break;` statement. What does the following code print out? (attempt this without compiling the code!)

```
int n = 0;
for (int i = 1; i <= 5; i++)
{
   System.out.print( i + ": ");
  for (int j = 1 ; j <= 5 ; j++)
  {
     n++;
     if (j * i  >= 15)
        break;
     System.out.print(j * i + " ");
  }
  System.out.println("");
}
System.out.println("Iterations: " + n);
```

(d) Look up the definition of the continue; statement. What does the following code print out? (attempt this without compiling the code!)

```
int n = 0;
for (int i = 1; i <= 5; i++)
{
  System.out.print( i + ": ");
  for (int j = 1 ; j <= 5 ; j++)
  {
     n++;
     if (j * i  >= 15)
           continue;
     System.out.print(j * i + " ");
  }
  System.out.println("");
}
System.out.println("Iterations: " + n);
```

(e) What is different about the two outputs? Explain why one line of code results in very similar but not identical output?

Put your answer in your PR5.pdf file.

Problem #7 (20 points) Times table with graphical interface.

Java has a function called printf() and format() that enables one to have finer control over print formatting that println(). printf() is a used across C, Java, C++, Python and other languages. Printf has the concept of a format string with replaceable parameters. In this exercise you will use the %d format descriptor to print integers in fixed-width field. The String class defines the format() method that behaves nearly identically to printf(). For example

```
int = 10
String S = “”.format(“% 4d”,i);
```

will create a String S that has integer i printed in using exactly 4 spaces and is right justified. E.g, “ 10”

Create a program call MultTable that uses JSliders, JTextArea, and JLabel to create a multiplication table that is NxM where N and M are controlled by two different sliders, that range independently from 1 ... 12. You should not extend WindowController, you should use native JFrames (you could start with HelloWordSwing.java). A JLabel should print out the current value of each slider. The following screen shots show the various configurations that you should be able to support.

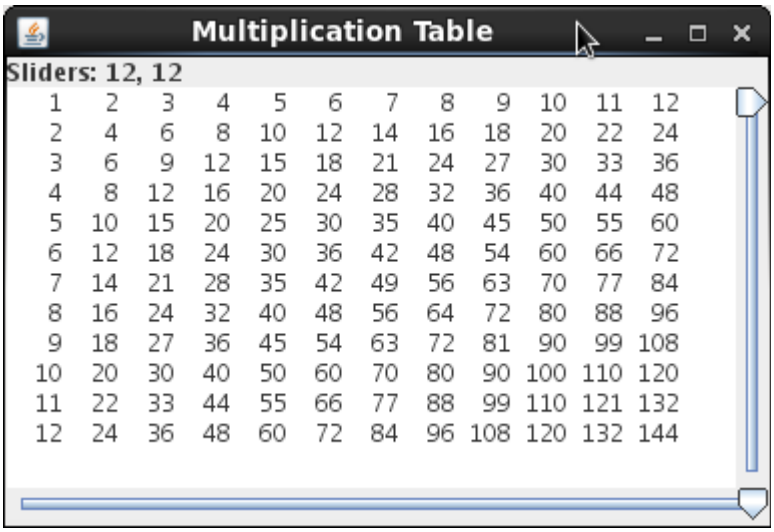

Illustration 2: Initial Configuration

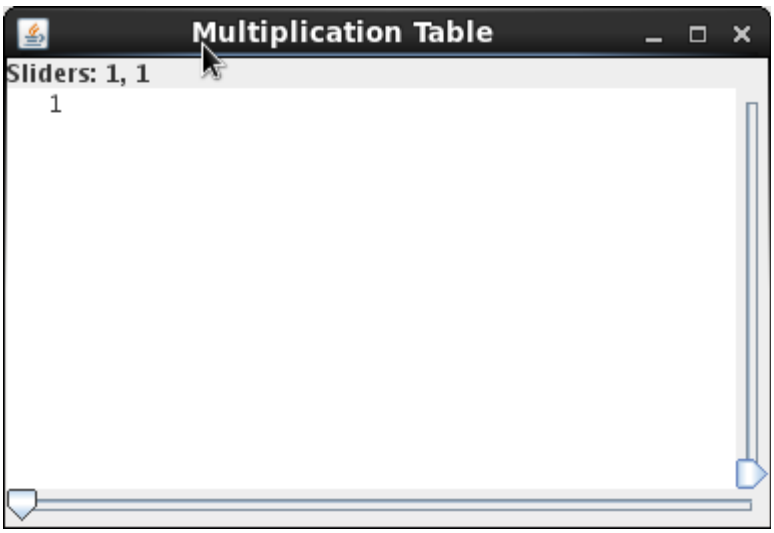

Illustration 3: Sliders when both are at 1

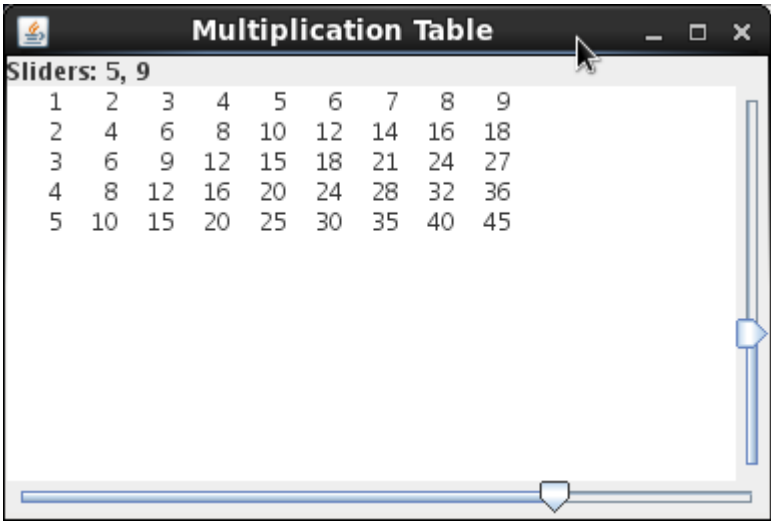

Illustration 4: When Sliders are at 7 (columns) and 5 (rows). Notice that the right slider controls number of rows, the bottom the number of columns

Hints:

The JTextArea in the above uses a fixed-width font. If output is a JTextArea the following will set the font to be fixed-width

```
output.setFont(new Font(Font.MONOSPACED, Font.PLAIN, 12));
```

To define a ChangeListener, you will need to import javax.swing.event.*.

BorderLayout() is probably the simplest layout to use to place sliders, jtextarea, and jlabels.

Look up the setEditable() method for JTextArea to insure that you cannot edit the multiplication table

You will turn in `MultTable.java`