APCS **Cool – Set 3**

1. **Write recursive method `inWords` used to translate an integer into words as shown by the output below. These methods go inside class `RecursionProblem1`.**

```
/**
 * Dominguez
 */
public class RecursionProblem1
{
    public static void main( String args[] )
    {
        do
        {
            System.out.print( "\n" + "Enter an integer..." );
            inWords( SavitchIn.readLineInt() );
            System.out.print( "\n" + "again? y/n..." );
        }while( SavitchIn.readLineNonwhiteChar() == 'y' );
    }

    //Precondition: numero >= 0
    public static void inWords( int numero )
    { ? }

     //method used by inWords
     private static String digitWord( int digit )
     {
        String result = null;
        switch( digit )
        {
            case 0: result = "zero"; break;
            case 1: result = "one";  break;
            case 2: result = "two";  break;
            case 3: result = "three"; break;
            case 4: result = "four"; break;
            case 5: result = "five"; break;
            case 6: result = "six"; break;
            case 7: result = "seven"; break;
            case 8: result = "eight"; break;
            case 9: result = "nine"; break;
            default: System.out.println( "Fatal Error" ); break;
        }
        return result;
    }
}
```

SAMPLE OUTPUT

```
Enter an integer...203490
  two zero three four nine zero
again? y/n...y

Enter an integer...30400
  three zero four zero zero
again? y/n...n
```

2. **Write recursive program to count the instances of a certain digit in a positive integer.**

SAMPLE OUTPUT

```
Enter a positive integer...55515505

Enter a digit to count...5

55515505 contains 6 5's.
```

3. **Complete recursive method `clean` in class `CleanString` to produce the output shown.**

```
public class CleanString
{
    public static void main( String args[] )
    {
      String name = "6--&*^==  SPa(())^^!!rtaNs --%%$#w ...iN";
      //String name = "&&&*^5$Th__e   KapTA..in$@!";
      System.out.println("\n" + "original string ==> " +
                         "\"" + name + "\".");
      System.out.print( "\n" + "new string ==> " + "\"" +
                        clean( name.toLowerCase() ) + "\"." );
    }

    public static String clean( String n )
    { ? }
}
```

OUTPUT

```
original string ==> "&&&*^5$Th__e   KapTAin$@!".

new string ==> "thekaptain".
-------------------------------------------------------------------------------------------------------------------------------
original string ==> "6--&*^==  SPa(())^^!!rtaNs --%%$#w ...iN".

new string ==> "spartanswin".
```

4. **Write a recursive palindrome program. In this program you will use the method `clean` from the `CleanString` program.**

```
public class PalindromeRecursive
{
  public static void main( String args[] )
  {
    //String ping = "^  &@C..at,))T*  ac! ";
    String pong = "10!A...m,,AN(()))&*^$, a   p  l a--n, a canal:++45 Panama";
    String m = pong;
    System.out.println("\n" + "The original string is \"" + m + "\".");
    m = clean( m.toLowerCase() );
    System.out.println("\n" + "...the new cleaned-up string is ..." +
                       "\"" + m + "\".");
    if( aPalindrome(m) ) System.out.print( "\n" + "\"" + m + "\"" +
                                           " is a palindrome." );
    else System.out.print( "\n" + "\"" + m + "\"" + " is not a palindrome." );
  }

  public static String clean( String n )
  { ? }

  public static boolean aPalindrome( String n )
  { ? }
}
```

SAMPLE OUTPUT

The original string is: "10!A...m,,AN(()))&*^$, a p l a--n, a canal:++45 Panama".

...the new cleaned-up string is ..."amanaplanacanalpanama".

"amanaplanacanalpanama" is a palindrome.

The original string is: " ^ &@C..at,))T* ac! ".

...the new cleaned-up string is ..."cattac".

"cattac" is a palindrome.

The original string is: " $%^i... ---LiKE!!__jAv??a ".

...the new cleaned-up string is ..."ilikejava".

"ilikejava" is not a palindrome.

5. **Write a recursive program that prompts you for your first and last names and determines how many of each letter your name contains. Look at the output shown below.**

```
Enter a first name...Mathew
Enter a last name...Bissonnette

Hello Mathew Bissonnette! My, what an awesome name!

Your name contains:

1 a
1 b
3 e's
1 h
1 i
1 m
2 n's
1 o
2 s's
3 t's
1 w
```

6. Write class CircleSquarePattern that implements the recursive method drawPattern to create the cool (this is why it is in the Cool Set) color pattern shown below.

```
/**
 * Dominguez
 */
public class CircleSquarePatternDriver
{
    public static void main( String args[] )
    {
        CircleSquarePattern spurs = new CircleSquarePattern( 500 );
    }
}
```

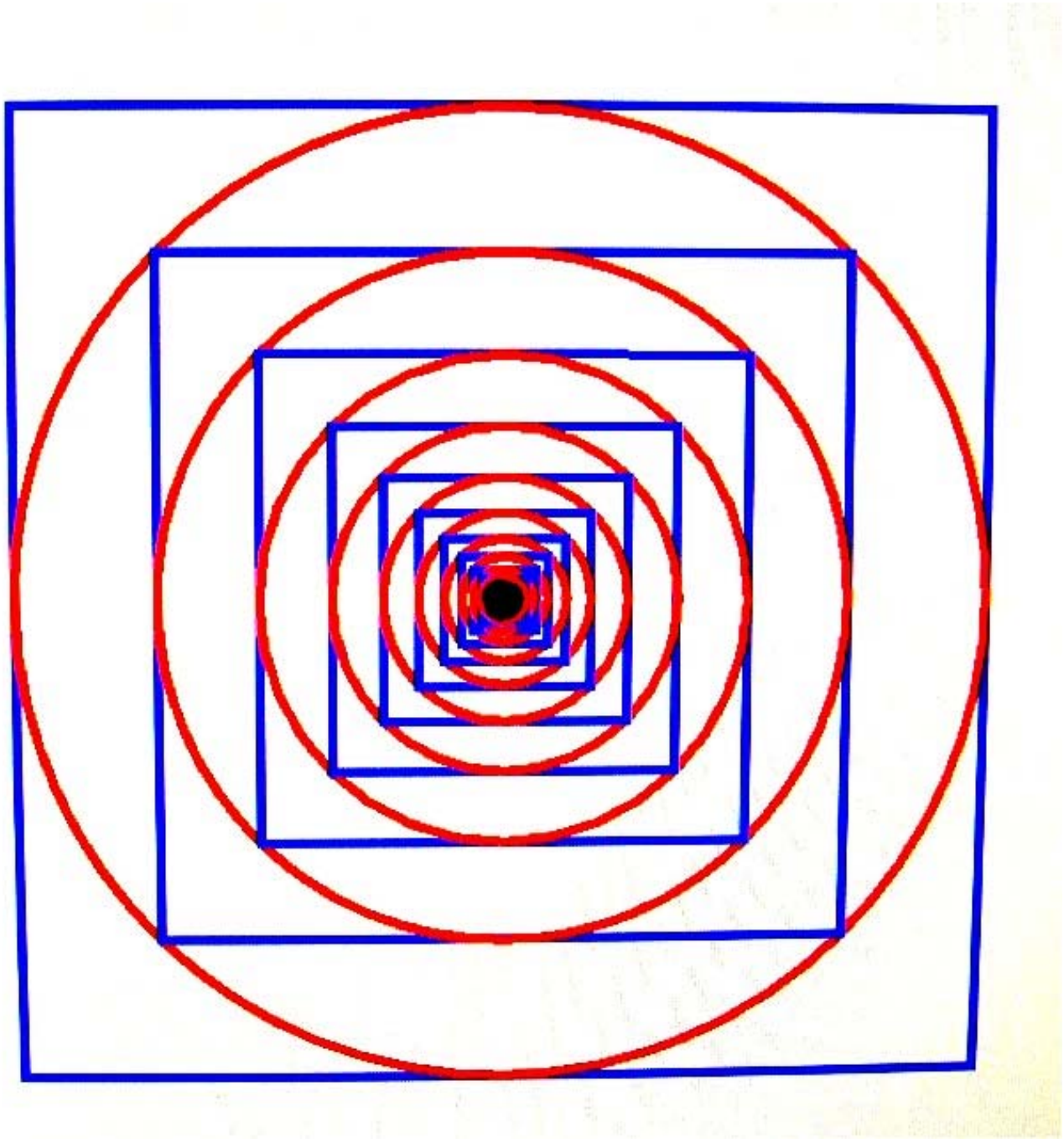
```
/**
 * Dominguez
 */
import java.awt.Color;
import gpdraw.*;
public class CircleSquarePattern
{
    private DrawingTool pen;
    private double side;

    public CircleSquarePattern( double length )
    {
        pen = new DrawingTool( new SketchPad(600, 600) );
        pen.setWidth(5);
        this.drawPattern( 10, length );
    }

  /**
   * @param  depth   level of recursion
   */
    public void drawPattern(int depth, double length)
    { ? }

}
```

OUTPUT



The next project is not a recursive program (at least I did not do it recursively)

7. Write a program to determine the time it takes a cylindrical water tank (of known height and diameter) to empty given a certain exit-nozzle diameter. Use feet-per-second for the units of velocity.

Print a table showing the elapsed time in seconds and the volume of fluid left in the tank.

Background:

If the water is flowing out of a tank where the air above the water is at atmospheric pressure as is the air outside the hole, and if the area of the surface of the water is much larger than the area of the hole, then, by Bernoulli's Law, the water will leave through the hole at a speed $velocity = \sqrt{2 \cdot g \cdot h}$, where ***g*** is the acceleration of gravity, ***h*** is the height of the surface of the water above the hole. Notice that this is the speed an object would get by falling through a height *h*. So the ***volume of water leaving per second***, ***Q***, will be

$$Q = Cross\text{-}sectional\ area * velocity$$

Also, we know that the volume of the fuel leaving the tank is

$$volume\ lost = velocity \bullet area\ of\ the\ nozzle \bullet time$$

Animated Demonstration of Bernoulli's Principle (very interesting for those of you engineer types):

http://home.earthlink.net/~mmc1919/venturi.html

OUTPUT

Enter the height of the tank in feet ==> 20
Enter the tank diameter in feet ==> 8
Enter the exit nozzle diameter in feet ==> .1666

Time(seconds)	Volume(cubic feet)
0.00	1005.31
200.00	854.99
400.00	716.83
600.00	590.84
800.00	477.01
1000.00	375.35
1200.00	285.85
1400.00	208.52
1600.00	143.36
1800.00	90.36
2000.00	49.54
2200.00	20.88
2400.00	4.41

The tank emptied after 2567.00 seconds, or 42.78 minutes.