APCS **Cool Set #1**

Determine the output of the programs shown below.

1.

```
public class ClassExample1
{
    public static void main( String[] args )
    {
        int k = 9, tortilla = 1;
        while( k>0 )
        {
            if( k%2 == 0 ) tortilla *= k;
            k--;
        }
        System.out.print("tortilla = " + tortilla);
    }
}
```

2.

```
public class ClassExample2
{
    public static void main( String[] args )
    {
        int sum = 0, count = 0;
        int lowNum = 8, maxNum = 20;
        double avg;
        for(int i = lowNum; i < maxNum + 1; i++)
        {
            sum += i;
            count++;
        }
        avg = (double)sum / count;
        System.out.printf("\n\t" + "The average of integers " + lowNum +
                          " through " + maxNum + " is " + "%6.2f", avg);
    }
}
```

3.

```
public class SumOfEvenInts2Thru30
{
  public static void main( String[] args )
  {
    int x = 0;
    int sum = 0;
    do
    {
      if( x % 2 == 0 ) sum += x;
      x++;
    }while( x < 31 );
    System.out.print("The sum of even integers 2 through 30 is " + sum);
  }
}
```

4.

```
public class SimpleForLoop1
{
    public static void main( String args[] )
    {
        int sum = 0;
        for( int number = 2; number <= 10; number += 2 )
        {
          sum += number;
        }
        System.out.print( "sum is " + sum );
    }
}
```

5.

```
public class AverageOf1Thru100
{
  public static void main( String[] args )
  {
    int sum = 0;
    double avg;
    int maxNum = 100;
    for(int i = 1; i < maxNum + 1; i++) sum += i;
    avg = (double)sum / maxNum;
    System.out.print("The average of integers 1 through " + maxNum +
                     " is " + avg + " .");
  }
}
```

6.

```
import gpdraw.*;
public class ConcentricColoredCircles
{
    public static void main( String[] args )
    {
       Color RED = Color.red, BLUE = Color.blue, BLACK = Color.black,
           GRAY = Color.gray, YELLOW = Color.yellow, PINK = Color.pink,
           ORANGE = Color.orange, GREEN = Color.green, MAGENTA = Color.magenta,
           WHITE = Color.white;

       SketchPad canvas = new SketchPad(600, 600);
       DrawingTool pencil = new DrawingTool( canvas );

       for( int i=1; i<10; i++ )
       {
           switch( i )
           {
               case 1:
                   pencil.setColor(RED);
                   break;

               case 2:
                   pencil.setColor(BLUE);
                   break;

               case 3:
                   pencil.setColor(BLACK);
                   break;

               case 4:
                   pencil.setColor(GRAY);
                   break;

               case 5:
                   pencil.setColor(YELLOW);
                   break;

               case 6:
                   pencil.setColor(PINK);
                   break;

               case 7:
                   pencil.setColor(ORANGE);
                   break;

               case 8:
                   pencil.setColor(GREEN);
                   break;

               case 9:
                   pencil.setColor(MAGENTA);
                   break;

               default:
                   break;
           }
            pencil.setWidth(50/i);
```

```
            pencil.drawCircle(85+15*i);

        }
    }
}
```

PROJECTS

1. **Write a class `Number` to work with `DigitsInANumberDriver` program. A sample of the output is shown below. Please note, and this is very important, that the driver program prompts for one integer (i.e. the integer to be analyzed is not entered one digit at a time).**

```
public class DigitsInANumberDriver
{
    public static void main( String[] args )
    {
        Number num = new Number();
        //note that num is a Number object. Don't
        //think of num as an int.
        int n;
        System.out.print("Enter an integer: ");
        n = SavitchIn.readLineInt();
        System.out.print("\n" + n + " has " +
                          num.numberOfDigits(n) + " digits.");
        num.showDigits( n );
    }
}
```

SAMPLE OUTPUT:

```
Enter an integer: 5275691

5275691 has 7 digits.

The digits of 5275691 are:  5   2   7   5   6   9   1
```

2. **Write a program that calculates the factorial of an integer. You can write one class and do it all inside of that class or you can write a driver program and a separate class to go with it. This is all up to you.**

SAMPLE OUTPUT:

```
Enter a number . . . 8

8! = 40320
```

3. **Problem statement:** ***A company wants to transmit data over the telephone, but they are concerned that their phones may be tapped. All of their data are transmitted as four-digit integers. They have asked you to write a program that encrypts their data so that it can be transmitted more securely. Your program should read a four-digit integer and encrypt it as follows: Replace each digit by*** **(the sum of that digit plus 7) modulus 10.** ***Then, swap the first digit with the third, swap the second digit with the fourth and print the encrypted integer. Write a separate program that inputs an encrypted four-digit integer and decrypts it to form the original number.***

Driver:

```
public class EncryptDemo
{
  public static void main( String[] args )
  {
    int originalNumber;
    System.out.print("Enter a four-digit number to encrypt: ");
    originalNumber = SavitchIn.readLineInt();
    Encrypt pin = new Encrypt(originalNumber);
    System.out.print("\nThe encrypted number is " + pin.getCode() );
    int codedNumber;
    System.out.print("\nEnter a four-digit encrypted number: ");
    codedNumber = SavitchIn.readLineInt();
    Decrypt num = new Decrypt(codedNumber);
    System.out.print("\nThe original number is " + num.getOriginal() );
  }
}
```

SAMPLE OUTPUT:

```
Enter a four-digit number to encrypt: 4893

The encrypted number is 6015
_____________________________________

Enter a four-digit encrypted number: 6015

The original number is 4893
```

4. **Write a program that reads in the size of the side of a square and then prints a hollow square of that size out of asterisks and blanks. Your program should work for squares of all side sizes between 1 and 20. For example, if your program reads a size of 5, it should print: (notice how the "square" looks more like a rectangle; that's OK)**

SAMPLE OUTPUT:

```
Enter the number of stars per side: 5

*****
*   *
*   *
*   *
*****
```

5. A right triangle can have sides whose lengths are all integers. A set of three integer values for the lengths of the sides of a right triangle is called a Pythagorean Triple. The lengths of the three sides must satisfy the relationship that the sum of the squares of the legs is equal to the square of the hypotenuse. Write class Triples to work with the driver shown below

```
public class PythagoreanTriples
{
    public static void main( String args[] )
    {
        System.out.print("You want to find all triples less than . . . ");
        int n = SavitchIn.readLineInt();
        Triples pythagoras = new Triples( n );
        pythagoras.findTriples();
    }
}
```

SAMPLE OUTPUT:

```
You want to find all triples less than . . . 50

side1    side2    hypotenuse
3        4          5
5        12         13
6        8          10
7        24         25
8        15         17
9        12         15
9        40         41
10       24         26
12       16         20
12       35         37
14       48         50
15       20         25
15       36         39
16       30         34
18       24         30
20       21         29
21       28         35
24       32         40
27       36         45
30       40         50
```