1 Short Answer (10 Points Each)

1. What is the difference between a compiler and an interpreter? Also, discuss how Java accomplishes this task.

   Solution: A compiler will take a program written in a high-level language, translate it into machine language and then save the machine language program to a file that can be run on the computer. An interpreter does essentially the same thing except that it translates the high-level language to machine language one command at a time and does not save the machine language program to a file. Java uses a combination of the two. There is a compile stage that translates the Java code into byte-code that the interpreter (known as the JVM or Java Virtual Machine) runs.

2. Answer the following questions about numeric data types in Java.

   (a) What happens when you overload an int?
   Solution: The value cycles around to the minimum value of an int.

   (b) What happens when you overload a double?
   Solution: The value turns into Infinity.

   (c) What happens when you underload an int?
   Solution: The value cycles around to the maximum value of an int.

   (d) What happens when you underload a double?
   Solution: The value turns into 0.

   (e) What happens when you input a double when the Scanner is doing a nextInt?
   Solution: Run-time error, since it will not automatically convert a double to an int.

3. Complete the following program so that given the user input cutoff scores for each grade level and the input student score the program will print out the student's letter grade. A sample run is below.

```
Input A Cutoff: 92
Input B Cutoff: 78
Input C Cutoff: 65
Input D Cutoff: 53
Input Score: 68
Grade: C
```

   Solution:

```
1  import java.util.Scanner;
2
3  public class FinalSA001 {
4
5      public static void main(String[] args) {
6          Scanner keyboard = new Scanner(System.in);
7          System.out.print("Input A Cutoff: ");
8          double Acut = keyboard.nextDouble();
9          System.out.print("Input B Cutoff: ");
10         double Bcut = keyboard.nextDouble();
11         System.out.print("Input C Cutoff: ");
12         double Ccut = keyboard.nextDouble();
13         System.out.print("Input D Cutoff: ");
14         double Dcut = keyboard.nextDouble();
15         System.out.print("Input Score: ");
16         double Score = keyboard.nextDouble();
17
18         System.out.print("Grade: ");
19
20         if (Score > Acut)
21             System.out.println("A");
22         else if (Score > Bcut)
23             System.out.println("B");
24         else if (Score > Ccut)
25             System.out.println("C");
26         else if (Score > Dcut)
27             System.out.println("D");
28         else
29             System.out.println("F");
30     }
31 }
```

4. Write the following declarations.

   (a) A one-dimensional array of 103 doubles.
   Solution: double A[] = new double[103];

   (b) A two-dimensional array of integers with 27 rows and 15 columns.
   Solution: int A[][] = new int[27][15];

   (c) An arraylist of integers.
   Solution: ArrayList¡Integer¿ A = new ArrayList¡Integer¿();

5. Write a linear search method for an integer array that takes in an array and target value as parameters and returns the first position of the target in the array. If the target is not in the array then the method should return -1.

   Solution:

```
public static int LinearSearch(int[] A, int N) {
    for (int index = 0; index < A.length; index++) {
        if (A[index] == N)
            return index;
    }
    return -1;
}
```

6. Write a method that does either the bubble sort, insertion sort or selection sort for an array of integers. You must state which sort you are writing.

   Solution:

```
public static void BubbleSort(int A[]) {
    for (int i = A.length - 1; i > 0; i--) {
        for (int j = 0; j < i; j++) {
            if (A[j] > A[j + 1]) {
                int temp = A[j];
                A[j] = A[j + 1];
                A[j + 1] = temp;
            }
        }
    }
}

public static void InsertionSort(int[] A) {
    for (int itemsSorted = 1; itemsSorted < A.length; itemsSorted++) {
        int temp = A[itemsSorted];
        int loc = itemsSorted - 1;
        while (loc >= 0 && A[loc] > temp) {
            A[loc + 1] = A[loc];
            loc = loc - 1;
        }
        A[loc + 1] = temp;
    }
}

public static void SelectionSort(int[] A) {
    for (int lastPlace = A.length - 1; lastPlace > 0; lastPlace--) {
        int maxLoc = 0;
        for (int j = 1; j <= lastPlace; j++)
            if (A[j] > A[maxLoc])
                maxLoc = j;

        int temp = A[maxLoc];
        A[maxLoc] = A[lastPlace];
        A[lastPlace] = temp;
    }
}
```

7. Write a method that takes four doubles as input parameters and returns the largest one.

   Solution:

```
1  public static double max(double a, double b, double c, double d) {
2      double m = a;
3
4      if (b > m)
5          m = b;
6      if (c > m)
7          m = c;
8      if (d > m)
9          m = d;
10
11     return m;
12 }
```

8. Write a method that will simulate the rolling of 5 die at a time and return the number of rolls needed for all the die to have the same value and each of the 6 values to be achieved in this manner. That is, the method should keep rolling 5 die at a time until there is a roll of 5 ones, a role of 5 twos, a role of 5 threes, a role of 5 fours, a role of 5 fives, a role of 5 sixes, not necessarily in order or consecutive.

   Solution:

```
1  public class FinalSA004 {
2
3      public static long rolls() {
4          long rollsneeded = 0;
5          boolean done = false;
6          boolean values[] = new boolean[6];
7
8          for (int i = 0; i < 6; i++) {
9              values[i] = false;
10         }
11
12         while (!done) {
13             rollsneeded++;
14             int r1 = (int) (Math.random() * 6 + 1);
15             int r2 = (int) (Math.random() * 6 + 1);
16             int r3 = (int) (Math.random() * 6 + 1);
17             int r4 = (int) (Math.random() * 6 + 1);
18             int r5 = (int) (Math.random() * 6 + 1);
19
20             if (r1 == r2 && r1 == r3 && r1 == r4 && r1 == r5)
21                 values[r1 - 1] = true;
22
23             done = true;
24             for (int i = 0; i < 6; i++) {
25                 if (!values[i])
26                     done = false;
27             }
28         }
29
30         return rollsneeded;
31     }
32
33     public static void main(String[] args) {
34         System.out.println("Rolls Needed = " + rolls());
35     }
36 }
```

9. Complete the following program by writing a method that takes in two integers as the first two entries in the output sequence and an integer for the number of terms in the sequence. The method is to be called `fib` and is to output an array list of integers. The array list is to contain the first two parameters as the first two entries and then each entry after that is the sum of the previous two entries. That is, entry three is the sum of entries one and two, entry four is the sum of entries two and three, and so on. Two runs of the program are below.

```
Input a: 1
Input b: 1
Input Number of Terms: 10
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
```

and

```
Input a: 3
Input b: 7
Input Number of Terms: 15
[3, 7, 10, 17, 27, 44, 71, 115, 186, 301, 487, 788, 1275, 2063, 3338]
```

Solution:

```
1  import java.util.ArrayList;
2  import java.util.Scanner;
3
4  public class FinalSA003 {
5
6      public static ArrayList<Integer> fib(int a, int b, int t){
7          ArrayList<Integer> A = new ArrayList<Integer>();
8          A.add(a);
9          A.add(b);
10
11         for (int i = 2; i < t; i++){
12             A.add(A.get(i-2) + A.get(i-1));
13         }
14
15         return A;
16     }
17
18     public static void main(String[] args) {
19         Scanner keyboard = new Scanner(System.in);
20         System.out.print("Input a: ");
21         int a = keyboard.nextInt();
22         System.out.print("Input b: ");
23         int b = keyboard.nextInt();
24         System.out.print("Input Number of Terms: ");
25         int t = keyboard.nextInt();
26
27         System.out.println(fib(a, b, t));
28     }
29 }
```

2 Program Traces (25 Points Each)

1. Write the output of the following program for the given inputs.

```
1  import java.util.Scanner;
2
3  public class FinalTrace001 {
4
5      public static int Method3(int n1, int n2) {
6          System.out.println("In Method 3");
7          int retval = n1 * n2;
8
9          if (retval > 15)
10             retval--;
11         else
12             retval += 5;
13
14         System.out.println("Method 3 Return: " +
               retval);
15         return retval;
16     }
17
18     public static int Method2(int n1, int n2) {
19         System.out.println("In Method 2");
20         int retval = n1 / n2;
21         int retval2 = n1 % n2;
22
23         if (retval > retval2)
24             retval = retval2;
25
26         System.out.println("Method 2 Return: " +
               retval);
27         return retval;
28     }
29
30     public static int Method1(int n1, int n2, int
           n3) {
31         System.out.println("In Method 1");
32         if (n1 > n2)
33             return Method2(n3, n2);
34         else if (n3 > n2)
35             return Method3(n2, n1);
36         else
37             return Method3(n1, Method2(n2, n3));
38     }
39
40     public static void main(String[] args) {
41         Scanner keyboard = new Scanner(System.in);
42         System.out.print("Input Number 1: ");
43         int n1 = keyboard.nextInt();
44         System.out.print("Input Number 2: ");
45         int n2 = keyboard.nextInt();
46         System.out.print("Input Number 3: ");
47         int n3 = keyboard.nextInt();
48
49         System.out.println(Method1(n1, n2, n3));
50     }
51 }
```

(a) **Solution:**

```
Input Number 1: 10
Input Number 2: 15
Input Number 3: 6
In Method 1
In Method 2
Method 2 Return: 2
In Method 3
Method 3 Return: 19
19
```

(b) **Solution:**

```
Input Number 1: 3
Input Number 2: 5
Input Number 3: 7
In Method 1
In Method 3
Method 3 Return: 20
20
```

(c) **Solution:**

```
Input Number 1: 100
Input Number 2: 23
Input Number 3: 125
In Method 1
In Method 2
Method 2 Return: 5
5
```

2. Write the output of the following program for the given inputs.

```
1  import java.util.Scanner;
2
3  public class FinalTrace002 {
4
5      public static void PrintArray(int[] Arr) {
6          for (int i = 0; i < Arr.length; i++) {
7              System.out.print(Arr[i] + "  ");
8          }
9          System.out.println();
10     }
11
12     public static void mix(int[] Arr, int n) {
13         int t = 3;
14         int r = 4;
15         for (int i = 0; i < n / 2; i++) {
16             Arr[r % Arr.length] = Arr[t % Arr.
                   length];
17             t += n;
18             r += r;
19         }
20     }
21
22     public static void main(String[] args) {
23         Scanner keyboard = new Scanner(System.in);
24         System.out.print("Input Size: ");
25         int size = keyboard.nextInt();
26         System.out.print("Input n: ");
27         int n = keyboard.nextInt();
28         int A[] = new int[size];
29
30         for (int i = 0; i < A.length; i++) {
31             A[i] = i+1;
32         }
33
34         mix(A, n);
35         PrintArray(A);
36     }
37 }
```

(a) **Solution:**

```
Input Size: 10
Input n: 9
1  2  1  4  4  6  2  8  3  10
```

(b) **Solution:**

```
Input Size: 13
Input n: 11
1  2  3  13  4  6  11  8  2  10  11  12  2
```

3 Coding (25 Points Each)

1. Write the three methods `PopulateArray`, `PrintArray`, and `FindMaxMin` that will complete this program, the main is given below along with a run of the program. The program takes in the number of rows and columns of a two-dimensional array along with an entry maximum and minimum. It populates the array with random integers between the input minimum and maximum, inclusively. It then outputs the array along with an array that has two rows and the same number of columns where the first row contains the column minimums and the second row contains the column maximums.

```
Input rows: 4
Input cols: 5
Input Entry Minimum: 5
Input Entry Maximum: 150
  128   21   75   98   46
  106   59  112   18   60
   66   15  106  139   30
   90   22  140  112   87

   66   15   75   18   30
  128   59  140  139   87
```

```
public static void main(String[] args) {
    Scanner keyboard = new Scanner(System.in);
    System.out.print("Input rows: ");
    int rows = keyboard.nextInt();
    System.out.print("Input cols: ");
    int cols = keyboard.nextInt();
    System.out.print("Input Entry Minimum: ");
    int min = keyboard.nextInt();
    System.out.print("Input Entry Maximum: ");
    int max = keyboard.nextInt();

    int[][] A = new int[rows][cols];
    PopulateArray(A, min, max);
    PrintArray(A);
    System.out.println();
    PrintArray(FindMaxMin(A));
}
```

Solution:

```
import java.util.Scanner;

public class FinalProg001 {

    public static void PrintArray(int A[][]) {
        int dim1 = A.length;
        int dim2 = A[0].length;

        for (int i = 0; i < dim1; i++) {
            for (int j = 0; j < dim2; j++) {
                System.out.printf("%5d", A[i][j]);
            }
            System.out.println();
        }
    }

    public static void PopulateArray(int A[][], int min, int max) {
        int dim1 = A.length;
        int dim2 = A[0].length;

        for (int i = 0; i < dim1; i++) {
            for (int j = 0; j < dim2; j++) {
                A[i][j] = (int)(Math.random()*(max - min + 1)) + min;
            }
        }
    }

    public static int[][] FindMaxMin(int A[][]) {
        int rows = A.length;
        int cols = A[0].length;
        int[][] B = new int[2][cols];

        for (int i = 0; i < cols; i++) {
            int max = A[0][i];
            int min = A[0][i];

            for (int j = 0; j < rows; j++) {
                if (A[j][i] > max)
                    max = A[j][i];
                if (A[j][i] < min)
                    min = A[j][i];
            }
            B[0][i] = min;
            B[1][i] = max;
        }
        return B;
    }

    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        System.out.print("Input rows: ");
        int rows = keyboard.nextInt();
        System.out.print("Input cols: ");
        int cols = keyboard.nextInt();
        System.out.print("Input Entry Minimum: ");
        int min = keyboard.nextInt();
        System.out.print("Input Entry Maximum: ");
        int max = keyboard.nextInt();

        int[][] A = new int[rows][cols];
        PopulateArray(A, min, max);
        PrintArray(A);
        System.out.println();
        PrintArray(FindMaxMin(A));
    }
}
```

2. Create a `Triangle` object that stores the lengths of the three sides of the triangle, as doubles, and has the following methods. A constructor that loads in the three side lengths, if any length is negative it should be replaced with 0. There should be six accessor methods, three for getting the lengths of each side and three for setting the lengths of each side. As with the constructor, if an input length is negative it should be replaced with 0. The object should have four more methods. One for determining the perimeter length of the triangle, which is the sum of the side lengths. One for determining the area of a triangle, recall that this can be done with the formula,

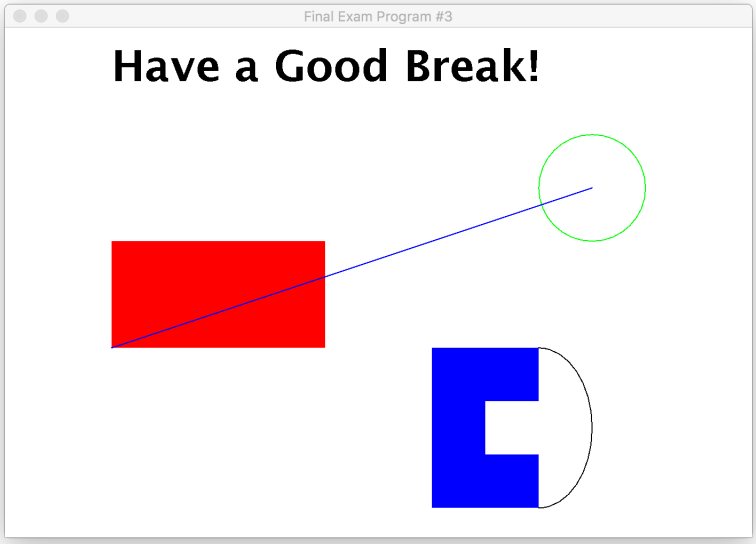
$$A = \sqrt{p(p-a)(p-b)(p-c)}$$

where $p = \frac{a+b+c}{2}$ is the semi-perimeter, that is, a, b, and c are the lengths of the sides. A method for determining if a triangle is a right triangle. Finally a method for determining if the side lengths will produce a triangle, that is, the lengths of the two shorter sided add up to be at least the length of the longest side. This object is not to take any input from the user nor do any output to the console. All input values are parameters and all output values are to be method returns.

Solution:

```
1  public class Triangle {
2      private double a;
3      private double b;
4      private double c;
5
6      // Constructor
7      public Triangle(double s1, double s2, double s3
           ) {
8          setSide1(s1);
9          setSide2(s2);
10         setSide3(s3);
11     }
12
13     // Accessor Methods
14     public double getSide1() {
15         return a;
16     }
17
18     public double getSide2() {
19         return b;
20     }
21
22     public double getSide3() {
23         return c;
24     }
25
26     public void setSide1(double s1) {
27         if (s1 > 0)
28             a = s1;
29         else
30             a = 0;
31     }
32
33     public void setSide2(double s2) {
34         if (s2 > 0)
35             b = s2;
36         else
37             b = 0;
38     }
39
40     public void setSide3(double s3) {
41         if (s3 > 0)
42             c = s3;
43         else
44             c = 0;
45     }
46
47     // Other Methods
48     public double Area() {
49         double p = (a + b + c) / 2;
50         return Math.sqrt(p * (p - a) * (p - b) * (p
               - c));
51     }
52
53     public double Perimeter() {
54         return a + b + c;
55     }
56
57     public boolean isRight() {
58         boolean righttri = false;
59         if (a * a + b * b == c * c)
60             righttri = true;
61
62         if (a * a + c * c == b * b)
63             righttri = true;
64
65         if (c * c + b * b == a * a)
66             righttri = true;
67
68         return righttri;
69     }
70
71     public boolean isTriangle() {
72         boolean tri = true;
73
74         double longleg = a;
75         if (b > longleg)
76             longleg = b;
77         if (c > longleg)
78             longleg = c;
79
80         if (a + b + c - longleg <= longleg)
81             tri = false;
82
83         return tri;
84     }
85 }
```

3. Write the paint method in the GraphicsJPanel class to produce the following image. The main and the shell for the GraphicsJPanel are below.



Solution:

```
1  import java.awt.*;
2  import javax.swing.*;
3
4  public class FinalProg003 extends JFrame {
5      private static FinalProg003 prog;
6      private GraphicsJPanel canvas;
7
8      public static void main(String[] args) {
9          prog = new FinalProg003(args);
10         prog.setTitle("Final Exam Program #3");
11
12         prog.setBounds(20, 20, 700, 500);
13         prog.setVisible(true);
14         prog.toFront();
15     }
16
17     public FinalProg003(String[] args) {
18         canvas = new GraphicsJPanel();
19
20         getContentPane().setLayout(new BorderLayout
               ());
21         getContentPane().add(canvas, BorderLayout.
               CENTER);
22     }
23 }
```

```
1  import java.awt.*;
2  import javax.swing.*;
3
4  public class GraphicsJPanel extends JPanel {
5
6      public GraphicsJPanel() {
7          setBackground(Color.white);
8      }
9
10     public void paint(Graphics g) {
11         super.paint(g);
12
13         g.setColor(Color.red);
14         g.fillRect(100, 200, 200, 100);
15
16         g.setColor(Color.black);
17         g.setFont(new Font(Font.SANS_SERIF, Font.
               BOLD, 40));
18         g.drawString("Have a Good Break!", 100, 50)
               ;
19
20         g.setColor(Color.green);
21         g.drawOval(500, 100, 100, 100);
22
23         g.setColor(Color.blue);
24         g.drawLine(100, 300, 550, 150);
25
26         g.setColor(Color.blue);
27         g.fillRect(400, 300, 100, 150);
28
29         g.setColor(Color.white);
30         g.fillRect(450, 350, 50, 50);
31
32         g.setColor(Color.black);
33         g.drawArc(450, 300, 100, 150, 270, 180);
34     }
35 }
```