INSTRUCTIONS:

- Keep your eyes on your own paper and do your best to prevent anyone else from seeing your work.
- Do NOT communicate with anyone other than the professor/proctor for ANY reason in ANY language in ANY manner.
- This exam is closed note, closed books, and no calculator.
- Turn all mobile devices off and put them away now. You cannot have them on your desk.
- Write neatly and clearly indicate your answers. What I cannot read, I will assume to be incorrect.
- Stop writing when told to do so at the end of the exam. I will take 5 points off your exam if I have to tell you multiples times.
- Academic misconduct will not be tolerated. Suspected academic misconduct will be immediately referred to the Emory Honor Council. Penalties for misconduct will be a zero on this exam, and F grade in the course, and/or other disciplinary action that may be applied by the Emory Honor Council.

TIME: This exam has 4 questions. Please check to make sure no page is missing. You will have 50 minutes to complete this exam.

I commit to uphold the ideals of honor and integrity by refusing to betray the trust bestowed upon me as a member of the Emory community. I have also read and understand the requirements and policies outlined above.

Signature:_______________________________

<table>
<thead>
<tr>
<th>Question</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points:</td>
<td>25</td>
<td>25</td>
<td>25</td>
<td>25</td>
<td>100</td>
</tr>
<tr>
<td>Score:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. (25 pts) Write down the output of the following program according to the order of execution.

```java
public class problem1 {
    public static int[] a = new int[]{1, 2, 3};

    public static void print(int[] a) {
        if (a == null) {
            print();
            return;
        }
        for (int i = 0; i < a.length; i++) {
            System.out.println(a[i]);
        }
    }

    public static void print() {
        for (int i = 0; i < a.length; i++) {
            System.out.println(a[i]);
        }
    }

    public static int[] a(int[] a) {
        int[] b = new int[a.length];
        for (int i = 0; i < a.length; i++) {
            b[i] = a[i] - 1;
        }
        return b;
    }

    public static void main(String[] args) {
        print(a);
        {
            int[] a = new int[]{7, 8, 9};
            print(a);
            print(null);
            problem1.a = a(a);
            print(a);
        }
        print(a);
    }
}
```
<table>
<thead>
<tr>
<th>Function</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td>2.</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>9</td>
</tr>
<tr>
<td>3.</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td>4.</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>9</td>
</tr>
<tr>
<td>5.</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
2. (25 pts) State whether the code is correct or has an error. Explain where and how the error occurs if any. The first row has been completed for you.

<table>
<thead>
<tr>
<th>Code</th>
<th>Error?</th>
<th>Why</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int[] a = new int[5];</code> &lt;br&gt; <code>for(int i = 0; i &lt;= a.length; i++) {</code> &lt;br&gt; <code>  a[i] = i;</code> &lt;br&gt; <code>}</code></td>
<td>Yes.</td>
<td>It should be i &lt; a.length in the for loop. Because the max index of a is a.length -1.</td>
</tr>
<tr>
<td><code>public class problem2 {</code> &lt;br&gt; <code>  public static int a = 1;</code> &lt;br&gt; <code>  public static int a() {</code> &lt;br&gt; <code>    return a;</code> &lt;br&gt; <code>  }</code> &lt;br&gt; <code>}</code></td>
<td>No.</td>
<td></td>
</tr>
<tr>
<td><code>public class problem2 {</code> &lt;br&gt; <code>  public static String a(String a) {</code> &lt;br&gt; <code>    return a;</code> &lt;br&gt; <code>  }</code> &lt;br&gt; <code>  public static String a(int a, int b) {</code> &lt;br&gt; <code>    return a(a);</code> &lt;br&gt; <code>  }</code> &lt;br&gt; <code>}</code></td>
<td>Yes.</td>
<td>There is no function a which takes one int as a parameter.</td>
</tr>
<tr>
<td><code>int[] a = {};</code></td>
<td>No.</td>
<td></td>
</tr>
<tr>
<td><code>int[] a = new int[10];</code> &lt;br&gt; <code>for(int i = 0; i &lt; a.length; i++) {</code> &lt;br&gt; <code>  a[i] = a[a.length - i];</code> &lt;br&gt; <code>}</code></td>
<td>Yes.</td>
<td>a[a.length – 0] would cause an error when i is 0.</td>
</tr>
<tr>
<td><code>int[][] a = new int[10][];</code> &lt;br&gt; <code>for(int i = 0; i &lt; a.length; i++) {</code> &lt;br&gt; <code>  a[i] = new int[a[i].length];</code> &lt;br&gt; <code>System.out.println(a[i].length);</code></td>
<td>Yes.</td>
<td>The entries in a are not initialized so that a[i].length will cause an error.</td>
</tr>
</tbody>
</table>
3. (25 pts) Nearest neighbor search.

Given a set of points and a query point (randomly generated), find the closest point in the set to the query point. We use a two-dimensional array (with size n-by-2) to store the set of n points. Each row in the array represents the two coordinates of a point. The number of points in the set is therefore the number of rows in the array.

```
public class problem3 {
    public static double[][] pointSet = {{4.0, 2.0}, {1.5, 3.0}, {4.2, 5.2}, {0.5, 4.9}, {5.0, 5.0}};
    public static double[] generateQueryPoint() {
        double[] point = new double[2];
        point[0] = Math.random() * 5.0;
        point[1] = Math.random() * 5.0;
        return point;
    }
    public static int nearestPoint(double[] point) {
        // compute the distance between query point and point 0
        double x0 = point[0] - pointSet[0][0];
        double y0 = point[1] - pointSet[0][1];
        double distance0 = Math.sqrt(x0 * x0 + y0 * y0);
        // assume distance0 is the shortest distance
        double shortestDistance = distance0;
        int index = 0;
        for (int i = 1; i < pointSet.length; i++) {
            double x = point[0] - pointSet[i][0];
            double y = point[1] - pointSet[i][1];
            double distance = Math.sqrt(x * x + y * y);
            // update shortest distance if necessary
            if (shortestDistance > distance) {
                shortestDistance = distance;
                index = i;
            }
        }
        return index;
    }
    public static void printResult(int i) {
        System.out.println("The nearest point is point "+ i + ": (" +
        pointSet[i][0] + ", " + pointSet[i][1] + ")");
    }
    public static void main(String[] args) {
        double[] point = generateQueryPoint();
        int i = nearestPoint(point);
        printResult(i);
    }
}
```
4. (25 pts) Fibonacci number.

Complete the following program to create and return a **Fibonacci sequence** with n numbers. You can assume n is always greater than 0.

A Fibonacci sequence is defined as following:

The first two numbers in the Fibonacci sequence are 0 and 1, and each subsequent number is the sum of the previous two. For example, when n is 6, the returned array should be {0, 1, 1, 2, 3, 5}

```java
public static int[] Fibonacci(int n) {
    int[] result = new int[n];
    for (int i = 0; i < 2 && i < result.length; i++) {
        result[i] = i;
    }
    for (int i = 2; i < result.length; i++) {
        result[i] = result[i - 1] + result[i - 2];
    }
    return result;
}
```