Name (print): ________________________________

- **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 notes, closed books, and no calculator.
  - Turn all mobile devices off and put them away now. You cannot have them on your desk.
  - You cannot use the Internet!
  - 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 multiple 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, an F grade in the course, and/or other disciplinary action that may be applied by the Emory Honor Council.

- **TIME:** This exam has 4 problem. 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>22</td>
<td>18</td>
<td>25</td>
<td>35</td>
<td>100</td>
</tr>
<tr>
<td>Score:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. (22 points) Write down the output of the following program according to the order of execution. The first row has been completed for you. Note, that the line number that should be included in the answer is the number specified on the left of code.

```java
public class Problem1 {

    public static long a = 1;

    public static int foo() {
        1. System.out.println(a);
        return 2;
    }

    public static int foo(int a) {
        2. System.out.println(a);
        a += 10;
        return 3;
    }

    public static int foo(long a) {
        3. System.out.println(a);
        return 4;
    }

    public static void main(String[] args) {
        {
            int a = 10;
            4. System.out.println(a);
            int b = foo(a);
            5. System.out.println(b);
            6. System.out.println(a);
            7. System.out.println(foo());
        }

        {
            8. System.out.println(foo(a));
        }

        9. System.out.println(foo(foo(3L)));
    }
}
```
2. (18 points) 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>Line number</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>Code</td>
<td>Error?</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
</tr>
<tr>
<td><code>String a = &quot;100&quot;; for (int i = 0; i &lt; a.length(); ++i) System.out.print(a[i]);</code></td>
<td>Yes</td>
</tr>
<tr>
<td><code>int[] a = new int[]; a[0] = 10;</code></td>
<td>Yes</td>
</tr>
<tr>
<td><code>public static int foo(int a) { System.out.println(a); }</code></td>
<td>Yes</td>
</tr>
<tr>
<td><code>public static int foo(int[] a) { return a[a.length]; }</code></td>
<td>Yes</td>
</tr>
<tr>
<td><code>int a = 10;</code></td>
<td>No</td>
</tr>
<tr>
<td><code>String a = &quot;10&quot;;</code></td>
<td></td>
</tr>
<tr>
<td><code>double[][] a = {1, 2};</code></td>
<td>Yes</td>
</tr>
</tbody>
</table>
3. (25 points) Complete the following Java program. The program calculates the sum of elements strictly above the main diagonal (elements on the main diagonal are highlighted) in each row of a square 2 dimensional array. For example:

\[
\begin{array}{ccc}
2 & 4 & 1 & 5 \\
3 & 1 & 2 & 6 \\
2 & 4 & 1 & 5 \\
5 & 4 & 3 & 2 \\
\end{array}
\]

The result is an array of \(\{10, 8, 5\}\), which is calculated by adding \(4 + 1 + 5\) on the first row, \(2 + 6\) on the second row and \(5\) on the third row. Note, that last row doesn’t have elements above the main diagonal.
public class Problem3 {
    public static void main(String[] args) {

        // 1. initialize matrix with elements from the example above
        int[] matrix =
                {
                    { 2, 4, 1, 5 },
                    { 3, 1, 2, 6 },
                    { 2, 4, 1, 5 },
                    { 5, 4, 3, 2 }
                };

        int[] res = getSums(matrix);

        // 2. Write code to print the matrix of results
        for (int i = 0; i < res.length; ++i)
            System.out.print(res[i] + " ");

    }

    // 3. specify the type of parameter
    public static int[] getSums( int[][] mat) {

        // 4. create an array of appropriate size
        int[] res = new int[mat.length - 1];

        // 5. finish the loop declaration
        for (int row = 0; row < mat.length - 1; ++row ) {
            // 6. Finish the loop
            for (int col = row + 1; col < mat[row].length; ++col ) {
                res[row] += mat[row][col];
            }
            return res; // 7. What should go here?
        }
    }
}

4. (35 points) Write a program that reads a positive integer from the user and prints back all its divisors. Your program should have a function called getDivisors that takes a number and returns an array containing all its divisors. Note: the number of divisors is
not known beforehand, so use the resizing technique we learnt in class. You can declare
resize method that takes an array and a new size and returns a new array of the given
size with elements copied from the first array.

```java
import java.util.Scanner;

public class Divisors {

    public static int[] resize(int[] array, int newSize) {
        int[] res = new int[newSize];

        // let's have a loop up to the minimum
        // of the new and old size so we can resize down as well.
        for (int i = 0; i < Math.min(array.length, newSize); ++i) {
            res[i] = array[i];
        }
        return res;
    }

    public static int[] getDivisors(int n) {
        int divCount = 0;
        int[] divisors = new int[10];

        for (int div = 1; div <= n; ++div) {
            if (n % div == 0) {
                divisors[divCount] = div;
                ++divCount;
                if (divCount == divisors.length) {
                    divisors = resize(divisors, divCount * 2);
                }
            }
        }

        // Now divisors array can be larger than
        // the actual number of divisors, let's resize it
divisors = resize(divisors, divCount);

        return divisors;
    }

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Please enter a number: ");
    }

}
int n = input.nextInt();

int[] divisors = getDivisors(n);

System.out.print("Divisors of " + n + " are:");
for (int i = 0; i < divisors.length; ++i)
    System.out.print(" " + divisors[i]);
System.out.println();

}