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.
  - 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 8 questions on 11 pages including the title page. Please check to make sure all pages are included. 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>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points:</td>
<td>4</td>
<td>9</td>
<td>9</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>50</td>
</tr>
<tr>
<td>Score:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. (4 points) Convert each of the following numbers from binary to decimal or from decimal to binary, as specified. **Put your answer to each question on the line provided.**

   (a) 110110 from binary to decimal
       Answer: ________
       Solution: 54

   (b) 28 from decimal to binary
       Answer: ________
       Solution: 11100

   (c) 52 from decimal to binary
       Answer: ________
       Solution: 110100

   (d) 10101 from binary to decimal
       Answer: ________
       Solution: 21
2. (9 points) Each question below contains 2 different compile-time and/or run-time errors. Find them in each question, and circle what’s causing the error. Pick one of the errors for each question, put a * to the left of the line the error is on, and indicate whether the error is a compile-time or run-time error.

(a) Type of error: ________________

```java
int[] numbers = new int {3, 2, 1, 0, 2};
for(int i = 0; i < numbers.length; i++){
    numbers[i-1] = numbers[i];
}
```

Solution:

new int should be new int[], which is a **compile-time** error.

numbers[i-1] will cause a **run-time error** when it accesses numbers[-1]

(b) Type of error: ________________

```java
String greeting = "hello!", greeting2 = 'hi';
char[] mashup = new char[greeting.length()]
for(int i == 0; i < 5; i++){
    if(i%2 > 0){
        mashup[i] = greeting.charAt(i);
    } else {
        mashup[i] = greeting2.charAt(i/3);
    }
}
```

Solution:

greeting2 = 'hi' should be greeting2 = "hi", which is a **compile-time** error.

int i == 0 in the initial action should be int i = 0, a **compile-time** error.
(c) Type of error: 
int x = 0, y = 1, z = -1;

int[] xyz = new int[3];
xyz[0] = x;
xyz[1] = y;
xyz[2] = z;

for(int element : xyz){
    System.out.println("Original element is " element);
}

for(i = 0; i < 3; ){
    if(x == 0){
        x++;
    }
    i = x;
}

Solution: The print statement is missing a + between the string and element, a compile-time error. The for loop’s initial action is missing a type, a compile-time error. It should be int i = 0.
3. (9 points) For each of the code fragments below, write in the whitespace below the code what output is produced when `bar()` is called. Take care to format the output correctly.

(a) public static double foo(int x){
    System.out.print("foo! ");
    double out = x/2 + 1;
    if(out < x*3){
        x++;
        out--;
    }

    return out;
}

public static void bar(){
    int x = 10;
    int val = 3;
    double y;

    val = (int) foo(x);
    y = foo(val);
    System.out.println("x has value " + x);
    System.out.println("val has value " + val);
    System.out.println("y has value " + (int) Math.floor(y));
}

Solution:
foo! foo! x has value 10
val has value 5
y has value 2
(b) public static void foo(int[] array, int num) {
    for (int index = 0; index < num; index++) {
        array[index] += 2*num;
    }
}

public static void bar() {
    int[][] values = {{0, 1}, {2, 3}};
    for (int index = 0; index < values.length; index++) {
        System.out.println("Bar "+ index + "!");
        foo(values[index], index+1);
    }
    System.out.println(values[0][0] + ", " + values[0][1] + 
                       "\n" + values[1][0] + ", " + values[1][1]);
}

Solution:

Bar 0!
Bar 1!
2, 1
6, 7
4. For each of the following questions, consider the series of arrays as steps in a sorting algorithm iteration. Identify on the line provided which algorithm is being performed, and write the array as it would appear after the next iteration in the whitespace provided.

(a) (2 points) Sorting algorithm: ____________________
{3, 0, -2, 4, 7, 3}
{-2, 0, 3, 4, 7, 3}
{-2, 0, 3, 4, 7, 3}

**Solution**: Selection sort: {-2, 0, 3, 4, 7, 3}

(b) (2 points) Sorting algorithm: ____________________
{1, 4, 0, 3, -1}
{1, 4, 0, 3, -1}
{0, 1, 4, 3, -1}

**Solution**: Insertion sort: {0, 1, 3, 4, -1}

(c) (2 points) Sorting algorithm: ____________________
{6, 5, 3, 2, -1}
{-1, 5, 3, 2, 6}

**Solution**: Selection sort: {-1, 2, 3, 5, 6}
5. For the following arrays, list the index inspected (so, which index has its value compared to the key) at each step for the given searching algorithm. Separately, indicate and circle the index returned by the search. If the element is not found, assume the algorithm returns -1. If the algorithm is not applicable, write “impossible”.

(a) (1 point) \{-1, 0, 2, 6, 7, 9, 15\} binary search with key -3.

**Solution:** Indices: 3, 1, 0; return: -1

(b) (1 point) \{-1, 2, 4, 0, 6, 20, 23, 24\} linear search with key 30.

**Solution:** Indices: 0, 1, 2, 3, 4, 5, 6, 7; return: -1

(c) (1 point) \{-1, 3, 2, 6, 4\} linear search with key 2.

**Solution:** Indices: 0, 1, 2; return: 2

(d) (1 point) \{0, 2, 3, 1, 5, 6, 8\} binary search with key 1.

**Solution:** Impossible: array is not sorted.
6. (6 points) Write a method named `stringRep` that, given a string and an integer \( N \), returns a string made of \( N \) repetitions of the last \( N-1 \) characters of the string. You may assume that the string has at least one character, and \( N \) is between 1 and the length of the string, inclusive.

Example inputs → return values:

- ("Kiwi", 1) → ""
- ("Kiwi", 2) → "ii"
- ("Kiwi", 3) → "wiwiwi"

**Solution:**

```java
public static String stringRep(String word, int N){
    String out = "";
    String end = "";

    for(int i = N-1; i >= 1; i--){
        end += word.charAt(word.length() - i);
    }

    for(int i = 1; i <= N; i++){
        out += end;
    }

    return out;
}
```
7. (6 points) Write a method named `bottomSum` that takes an array of doubles and returns the sum of the smallest and second smallest values in the array. You may assume the array will have at least two elements in it.

Example inputs → return values:
{1.0, 4.0, 2.5, 3.2, 9.0, 0.2} → 1.2
{2.2, 3.3, 0.0, 1.1} → 1.1
{-5.3, -4.3} → -9.6

Solution:

```java
public static double bottomSum(double[] array){
    double curMin, secMin;
    if(array[0] < array[1]){ // Assume array[0] is smaller than array[1]
        curMin = array[0];
        secMin = array[1];
    } else { // Assume array[0] is larger than array[1]
        secMin = array[0];
        curMin = array[1];
    }

    for(int i = 2; i < array.length; i++){ // Iterate through the rest of the array
        if(array[i] < curMin){ // Check if the current element is smaller than curMin
            secMin = curMin; // Update secMin to the current minimum
            curMin = array[i]; // Update curMin to the current element
        } else if(array[i] < secMin){ // Check if the current element is smaller than secMin
            secMin = array[i]; // Update secMin to the current element
        }
    }

    return curMin + secMin; // Return the sum of the two smallest elements
}
```
8. (6 points) Given that a method with the method header

    public static int findMin(int[] values)

already exists, write an method overloading the findMin method. Your overloaded method should find and return the minimum element in an array of doubles. The array is not guaranteed to have any elements in it.

    Solution: Note: this is not possible for empty arrays! This was a mistake.

    public static double findMin(double[] values){
        if(values.length < 1){
            // unspecified.
        }
        double curMin = values[0];

        for(int i = 0; i < values.length; i++){
            if(values[i] < curMin){
                curMin = values[i];
            }
        }

        return curMin;
    }