Name (as in OPUS) (print): _________________________________

Section: __________________________ Seat Assignment: ________________

- **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 9 pages including the title page. Please check to make sure all pages are included. You will have 70 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>6</td>
<td>3</td>
<td>6</td>
<td>15</td>
<td>5</td>
<td>7</td>
<td>18</td>
<td>10</td>
<td>70</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. For each print statement below, give the resulting value which is printed to the screen. **Your answer should indicate type:** that is, Strings should be enclosed in quotation marks ("), characters should be enclosed in single quotation marks ('), and numerical types should correctly indicate floating point or integer precision. Each statement should be evaluated independently. In other words, assume the statements begin with the initial variable values below. If the statement would result in an error, you may simply write “error” as your answer.

   **String** word = "CS170 abcd";
   **int** i = 4;
   **int** j = 20;
   **char** c = ‘d’;

   (a) (1 point) System.out.println(word.length());
   
   (b) (1 point) System.out.println(word.charAt(6));
   
   (c) (1 point) System.out.println(c + 5);
   
   (d) (1 point) System.out.println(i);
   
   (e) (1 point) System.out.println(j++ - i++);
   
   (f) (1 point) System.out.println(“abcd” == word.substring(6))

   The ASCII values ’A’-’Z’ are 65-90 and the values for ’a’-’z’ are 97-122.

2. Code Tracing

   (a) (1 point) You invoke a program from the command line with:
   
   java MyProgram 20 3.14 CS true
   
   Give the value and type of args[1] in main.

   (a) **3.14, String**

   (b) (2 points) What is returned by the following method if it is invoked with the call mystery(5)?

   public static int mystery(int n) {
     if(n <= 2) {
       return 4;
     } else {
       return mystery(n-2) + n;
     }
   }

   (b) **12**
3. Consider the following code:

```java
public static void swap1(int[] a) {
    int temp = a[0];
    int last = a.length - 1;
    a[0] = a[last];
    a[last] = temp;
}

public static void swap2(int num1, int num2) {
    int temp = num1;
    num1 = num2;
    num2 = temp;
}

public static void allZeros(int[] a, int num) {
    a = new int[num];
}

public static void main(String[] args) {
    int[] array = {10, 20, 30, 40, 50};
    //method call from following questions here
}
```

(a) (2 points) Draw the array `array` if the comment in `main` is replaced with the statement: `swap1(array);`

Solution: 

```
50  20  30  40  10
```

(b) (2 points) Draw the array `array` if the comment in `main` is replaced with the statement: `swap2(array[1], array[3]);`

Solution: 

```
10  20  30  40  50
```

(c) (2 points) Draw the array `array` if the comment in `main` is replaced with the statement: `allZeros(array, 5);`

Solution: 

```
10  20  30  40  50
```
4. For each of the following questions, give the output of the code. Write your answer in the box provided next to the code. If the code would result in an infinite loop, give the first 3 outputs and then write “infinite loop.” If the code would result in an error for any reason, you can simply write “error.”

(a) (3 points)
for(int i = 3; i < 5; i++) {
    int j = 10;
    while(j > 5) {
        System.out.print((j-i) + " ");
        j -= 2;
    }
    System.out.print("\n");
}

Solution: 7 5 3
6 4 2
Scoring: +1.5 pt each line of output

(b) (3 points)
int n = 9;
while (n > 0) {
    n -= 3;
    if (n % 4 == 2) {
        n++;
    }
    System.out.println(n);
}

Solution: 7
4
1
-2

(c) (3 points)
int[] a = {10, 20, 30, 40, 50, 60};
int[] b = {1, 2, 3, 4, 5};
for(int i = 1; i <= 3; i++) {
    System.out.println( a[b[i]]);
}

Solution: 30
40
50
Scoring: 1 for each correct number.

(d) (3 points)
int i = 1;
while(i < 5) {
    if (i % 2 == 0) {
        System.out.println("even "+ i);
        continue;
    } else {
        System.out.println("odd "+ i);
    }
    i++;
}

Solution: odd 1
even 2
even 2
infinite loop
Scoring: 1.5pt for output; 1.5 pt for infinite loop
(e) (3 points) Assume the following method is invoked with \texttt{easterBunny(24)}:

```java
public static void easterBunny(int n) {
    if (n > 0) {
        if (n % 4 == 0) {
            System.out.println("hippity " + n);
            easterBunny(n/3);
        }
        if (n % 3 == 0) {
            System.out.println("hop " + n);
        }
    }
}
```

**Solution:**
- hippity 24
- hippity 8
- hop 24

Scoring: +1 each line of output

5. (5 points) Match each of the following code snippets in the left hand column to its output in the right hand column by drawing a line and connecting them. Each output may not be used or may be used more than once.

| A. int n = 3; int i = 0; while (i <= n-1) { System.out.print(i); i++; } | I. 012 |
| B. int n = 3; int i = 0; while (i < n) { i++; System.out.print(i); } | II. 023 |
| C. int n = 3; int i = 1; while (i <= n) { System.out.print(i); i++; } | III. 123 |
| D. int n = 3; for (int i = n; i > 0; i--) { System.out.print(n-i); } | IV. 210 |
| E. int n = 4; for (int i = 0; i < n; i += 3) { int j = n - i; while (j < n) { System.out.print(j); j++; } } | V. 321 |
| F. int n = 4; for (int i = 0; i < n; i += 3) { int j = n - i; while (j < n) { System.out.print(j); j++; } } | VI. 0123 |
| G. int n = 4; for (int i = 0; i < n; i += 3) { int j = n - i; while (j < n) { System.out.print(j); j++; } } | VII. 0234 |
| H. int n = 4; for (int i = 0; i < n; i += 3) { int j = n - i; while (j < n) { System.out.print(j); j++; } } | VIII. 1234 |
| I. int n = 3; int i = 0; while (i <= n-1) { System.out.print(i); i++; } | IX. 3210 |
**Solution:** (A) answer: 012 (B) answer: 123 (C) answer: 123 (D) answer: 012 (E) answer: 123
6. The method `sumOdds` is supposed to calculate and return the sum of all the odd numbers between `a` and `b` (extremes included), with `a >= 0` and `a <= b`. However it contains a mistake that makes it behave incorrectly in some cases.

```java
public static int sumOdds(int a, int b) {
    if (a < 0) {
        a = 0;
    }

    int sum = 0;
    while (a <= b) {
        if (a % 2 == 1) {
            sum += a;
            a += 2;
        } else {
            a += 2;
        }
    }

    return sum;
}
```

(a) (2 points) Complete the sample method call below with values for which the method returns the **correct** result. Then give the value returned in this case.

```java
sumOdds(any odd value, any value greater/equal to a);
```

value returned: **answers vary**

**Solution:**

(b) (3 points) Complete the method call below with values for which the method returns an **incorrect** result. Then give the result that **should be** returned (if the method functioned as specified), and the result that is actually returned by the method as written.

```java
sumOdds(any even value, any value greater/equal to a);
```

value that should be returned: **answers vary**

value returned: **answers vary**

(c) (2 points) Fix the mistake so the method will always return the correct result if `a >= 0` and `a <= b`. You do not need to rewrite the entire method; simply use the line numbers and rewrite the incorrect line(s).

**Solution:** line 11 should be modified to `a += 1`
7. For each of the following parts, you will be writing a method named `asSum` which takes an integer, `n`, and returns a `String` representing the value of `n` as the sum of `n` positive units. If `n` is negative or zero, the method returns the empty string. Examples:

- `asSum(3)` returns "1+1+1"
- `asSum(1)` returns "1"
- `asSum(-3)` returns "" (empty String)

(a) (5 points) Write `asSum` using a `for` loop

Solution:

```java
public static String asSum(int n) {
    String result = "";
    if (n > 0) {
        result += "1";
    }
    for (int i = 1; i < n; i++) {
        result += "+1";
    }
    return result;
}
```

(b) (5 points) Write `asSum` using a `while` loop

Solution:

```java
public static String asSum(int n) {
    String result = "";
    if (n > 0) {
        result += "1";
    }
    int i = 1;
    while (i < n) {
        result += "+1";
        i++;
    }
    return result;
}
```

(c) (8 points) Write `asSum` as a recursive method

Solution:

```java
public static String asSum(int n) {
    if (n <= 0) {
        return "";
    } else if (n == 1) {
        return "1";
    } else {
        return "1+" + asSum(n - 1);
    }
}
```
8. (10 points) Write a method named `sumArrays` that takes two arrays of integers as parameters, `x` and `y`. The method returns a new array where each element is the sum of the corresponding elements from `x` and `y`. If `x` and `y` have different lengths, the missing elements in the shorter array are treated as zeros. Your code should be able to handle an empty array as a parameter.

Examples:

- `sumArrays({1, 2, 3}, {5, 1, 2})` returns `{6, 3, 5}`
- `sumArrays({1, 2, 3}, {5})` returns `{6, 2, 3}`
- `sumArrays({1}, {5, 1, 2})` returns `{6, 1, 2}`
- `sumArrays({}, {})` returns `{}`

Solution:

```java
public static int[] sumArrays(int[] x, int[] y) {
    int[] result;
    if (x.length >= y.length) {
        result = new int[x.length];
    } else {
        result = new int[y.length];
    }

    for (int i = 0; i < result.length; i++) {
        int a = 0;
        if (i < x.length) {
            a = x[i];
        }
        int b = 0;
        if (i < y.length) {
            b = y[i];
        }
        result[i] = a + b;
    }
    return result;
}
```

Another solution with a different approach:

```java
public static int[] sumArrays(int[] x, int[] y) {
    int[] result = new int[Math.max(x.length, y.length)];
    for(int i = 0; i < x.length; i++) {
        result[i] += x[i];
    }
    for(int i = 0; i < y.length; i++) {
        result[i] += y[i];
    }
    return result;
}
```