• **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 13 pages including the title page. Please check to make sure all pages are included. You will have 75 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>10</td>
<td>10</td>
<td>11</td>
<td>15</td>
<td>5</td>
<td>5</td>
<td>9</td>
<td>10</td>
<td>75</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. (10 points) Definitions. Define 5 of the 6 terms below. You do not need to give a formal definition just a good description. You may include an example if it is helpful to your answer. Be brief, and clearly mark the term you do not want me to grade. Otherwise, I will discard the last term.

(a) local variable

Solution: A local variable is a variable that is only in scope (is visible or can be used) within the block of code that defined it. (for example, an if statement or method body). Once the function returns, the local variable is lost.

(b) keyword

Solution: A reserved word which is used by the compiler and has a special meaning in the language structure. Ex: if, int, class, etc

(c) argument

Solution: A value provided to a function when the function is called. This value is assigned to the corresponding parameter in the function

(d) compiler

Solution: A reserved word that is used by the compiler to parse the program; you cannot use keywords (such as if, class, and void) as variable or function names (identifiers).

(e) ASCII code (or encoding)

Solution: The internal mapping a computer makes between a symbol and a numerical value. Ex: ‘A’ maps to 65.

(f) string

Solution: A series of characters or symbols. In Java, denoted by use of quotation marks, Ex: "Hello World!"
2. (10 points) Evaluate each expression. Then give the result of the evaluation and the data type of the result. If the expression cannot be evaluated or is not proper Java syntax, you may simply write “error” for the value. The first row has been done for you.

String s1 = "Spring", s2 = "85";
char c1 = '5', c2 = 'q';
int i1 = 7, i2 = 4;
double d1 = 1.0, d2 = 0.75;
boolean b1 = true;

<table>
<thead>
<tr>
<th>Expression</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>4+1</td>
<td>5</td>
<td>int</td>
</tr>
<tr>
<td>i1 % i2</td>
<td>3</td>
<td>int</td>
</tr>
<tr>
<td>(i2+d1+2) / i1</td>
<td>1.0</td>
<td>double</td>
</tr>
<tr>
<td>!b1</td>
<td>false</td>
<td>boolean</td>
</tr>
<tr>
<td>b1 &amp;&amp; i2 != i1</td>
<td>true</td>
<td>boolean</td>
</tr>
<tr>
<td>&quot;c1&quot; + s1</td>
<td>“c1Spring”</td>
<td>String</td>
</tr>
<tr>
<td>c2 + (int)d2</td>
<td>113</td>
<td>int</td>
</tr>
<tr>
<td>s1.charAt(4) + i2 + s2</td>
<td>“11485”</td>
<td>String</td>
</tr>
<tr>
<td>(char)(i2 * 21)</td>
<td>'T'</td>
<td>char</td>
</tr>
<tr>
<td>i1-- != ++i2 + 1</td>
<td>true</td>
<td>boolean</td>
</tr>
<tr>
<td>c1 &lt; c2</td>
<td></td>
<td>i1 &lt; i2</td>
</tr>
</tbody>
</table>

Notes:

- The parenthesis operator has the highest precedence of all operators.
- The logical not operator and the casting operator have higher precedence than arithmetic operators, relational/comparison operators, and the other logical operators.
- Arithmetic operators have higher precedence than relational/comparison or logical operators.
- Relational/comparison operators have higher precedence than logical operators.
- Assignment operators have the lowest precedence of all operators.
3. (11 points) Consider the program below. What will be the output of the program when it is executed?

```java
public class Exam1 {
    public static int method1(int a, double b) {
        System.out.println("A " + a + " " + b);
        if (a > b) {
            System.out.println("B");
            b += 1.5;
            return (int)b;
        }
        if (b > a) {
            System.out.println("C");
            a -= 2;
            return a;
        }
        b += 12.5;
        return (int)(a+b);
    }
    public static void method2(int a) {
        if (a >= 20 && a <= 30) {
            a += 30;
            System.out.println("D " + a);
            return;
        } else {
            System.out.println("E " + a);
        }
        System.out.println("F");
    }
    public static String method3(String s) {
        System.out.println("Hello, " + s);
        if (s.length() > 5) {
            return s.substring(5);
        } else {
            return s.substring(2);
        }
    }
    public static void main(String[] args) {
        int a = 10;
        int b = 20;
        String s1 = method3("Hel" + a);
        method2(b);
        System.out.println("G " + s1);
        System.out.println("H " + a + " " + b);
        a = method1(b, a);
        System.out.println("I " + a + " " + b);
    }
}
```
**Solution:** Hello, Hel10
D 50
G 110
H 10 20
A 20 10.0
B
I 11 20
4. Short Answer.

(a) (2 points) In the following method, how many local variables are there? How many parameter variables are there?

```
public static double mystery(int a, String b, char c) {
    a = a + c;
    double d = a;
    boolean e = a <= c;
    b += c;
    return a;
}
```

**Solution:** 2 local variables (d and e)
3 parameter variables (a, b, and c)

(b) (3 points) Consider the following function:

```
public static int mysteryFunction(int a,int b) {
    if (a < b) {
        return a;
    }
    return b;
}
```

i. What does this function return if called with `mysteryFunction(5,10)`?

**Solution:** 5

ii. Write an appropriate comment for this method.

**Solution:** It returns the smaller of two input parameter numbers. In other words, it finds the minimum. It was not enough to restate the code!

(c) (2 points) Give the output of the following code fragment:

```
int a = 3;
int b = 10;
double pi = 3.14159;

if (b / a > pi)
    System.out.println("a");
else if ((double)b / a > pi)
    System.out.println("b");
else
    System.out.println("c");
```

**Solution:** b
(d) (2 points) What does the following method print if it is invoked with the call foo(-1, 8, -3)?

```java
public static void foo(int a, int b, int c) {
    if (b > c) {
        if (c > 0) {
            System.out.println("A");
        } else if (b > 0) {
            System.out.println("B");
            b = c - b;
        } else if (a < 0) {
            System.out.println("C");
        } else {
            System.out.println("D");
        } if (b > c) {
            System.out.println("E");
        } else {
            System.out.println("F");
        }
    } else {
        System.out.println("G");
    }
}
```

**Solution:** I ended up giving everyone full credit on this problem due to a significant typo. However, it’s been corrected in the version above, and you should understand how the above code gives the answer:

B
F

(e) (4 points) Give the output of the following code fragment:

```java
int a = 2;
int b = 10;
int c = 20;
int d = 7;

a += c;
b %= d;
System.out.println("a: " + a);
System.out.println("b: " + b);
c = ++a + b + d--;
System.out.println("c: " + c);
System.out.println("d: " + d);
```
Solution: a: 22
b: 3
c: 33
d: 6
(f) (2 points) Write a single Java statement which would cause the output:

She said,
"Hello" to me.

to appear in the terminal window.

Solution: System.out.println("She said\n"Hello\" to me");
5. (5 points) Prof. Summet has written the following code to convert numerical scores to letter grade for CS170. However, she has 5 syntax errors. Identify the errors by circling them and rewrite the line of code to be syntactically correct. Hint for 1 tricky error: what will the program do if the user enters -5?

```java
import java.util.Scanner;

public class Exam1{
    public static String grade(int score) {
        if (score >= 90)
            return "A";
        if (score >= 80)
            return "B";
        if (score >= 70)
            return "C";
        if (score >= 60)
            return "D";
        if (score => 0)
            return "F";
    }

    public static void main (String[] args){
        Scanner in = new Scanner(System.in);
        System.out.print("Enter a numerical grade: ");
        int score = in.nextInt();
        String g = Grade(score);
        System.out.println(score + " is a " + g);
    }
}
```

**Solution:**

- line 19: missing ;
- line 20: function name is grade not Grade
- line 21: missing " around String
- line 13: operator written incorrectly; should be >=
- lines 5-14: missing a return statement. It’s not enough just to add an else since the else will only pair with the last if statement. You need to either 1) add just a return statement like return "Error"; as the last statement of the method or rewrite the entire method to use else if and the last condition simply use an else.

Note that the space between main and the opening ( isn’t an error since Java is a format-free language.
6. (5 points) Write a method named isNegative that takes in an integer parameter and returns a boolean value. If the parameter is negative, the function should return the boolean value true. If the parameter is zero or positive, it should return the boolean value false.
Examples:
isNegative(-8) returns true
isNegative(5) returns false

Solution:

```java
public static boolean isNegative(int num) {
    if (num >= 0) {
        return false;
    } else {
        return true;
    }
}
```

Scoring:
1 point for a correct header
2 points for testing if the number is positive/negative correctly
1 point for returning false if the number is positive or zero
1 point for returning True if the number is negative
7. (9 points) Write a method named `uniqueProduct` which takes 3 integers as input. The function should return the product of the 3 inputs. However, if one of the input values is the same as another of the values, it should return the unique (not duplicated) number. If all three numbers are the same, the function should return -1. Examples:

```
uniqueProduct(2, 3, 4) returns 24
uniqueProduct(2, 2, 3) returns 3
uniqueProduct(1, 1, 1) returns -1
```

Solution:

```java
public static int uniqueProduct(int a, int b, int c) {
    if (a == b && b == c) {
        return -1;
    } else {
        if (a == b)
            return c;
        else if (b == c)
            return a;
        else if (a == c)
            return b;
        else
            return a * b * c;
    }
}
```

Scoring:
+2 header
+2 returns -1 if all are same
+4 handles cases where only 2 numbers are the same
+1 handles case where all are different
8. (10 points) Write a method named `strangeCat` which takes two Strings as input. This function should perform a “strange concatenation” as described below:

If the strings are different lengths, omit initial characters from the longer string so it is the same length as the shorter string. So “Hello” and “Hi” yield “loHi” (both of length 2, omits the initial 3 characters from “Hello”). If the strings are the same length, return a String made by repeating the first String. Examples:

```
strangeCat("Hello", "Hi") returns "loHi"
strangeCat("Java", "Hello") returns "Javaello"
strangeCat("Ho", "Hi") returns "HoHo"
```

**Solution:**

```java
public static String strangeCat(String a, String b) {
    if (a.length() > b.length()) {
        return a.substring(a.length()-b.length()) + b;
    } else if (a.length() < b.length()){
        return a + b.substring(b.length()-a.length());
    } else {
        return a+a;
    }
}
```

**Scoring:**

+2 header
+2 handles if b longer than a
+2 handles if a longer than b
+2 handles if strings are same length
+2 omits initial characters correctly (uses substring correctly)