Name (as on OPUS): ______________________________________________________

UserID: ________________________  Section: _________  Seat: _______

• 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.
  – 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.

• ANSWERS: Write your answers neatly and clearly in the space indicated. What I cannot read, I will assume to be incorrect. If you need to continue your answer on another page, state this in the answer area and give a page number so we can find your answer.

• TIME: This exam has 6 parts 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>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points:</td>
<td>9</td>
<td>7</td>
<td>10</td>
<td>20</td>
<td>6</td>
<td>18</td>
<td>70</td>
</tr>
<tr>
<td>Score:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Scope. Consider the following program.

```java
public class Scope {
    public static int var = 10;

    public static void first(int a, int b) {
        int t = a;
        a = b;
        b = t;
        System.out.println(a + "", " + b ); /*1*/
    }

    public static void second(int b) {
        int a = 6;
        System.out.println(a); /*2*/
    }

    public static int third(int var) {
        System.out.println(var); /*3*/
        return var - 10;
    }

    public static void main(String[] args) {
        int a = 2;
        int b = 4;
        if (b < 10) {
            int var = -3;
            System.out.println(var); /*4*/
        }
        System.out.println(var); /*5*/
        System.out.println(a + "", " + b ); /*6*/
        first(a, b);
        System.out.println(a + "", " + b ); /*7*/
        second(a);
        System.out.println(a); /*8*/
        var = third(4);
        System.out.println(var); /*9*/
    }
}
```

Give the output of the code at each location. If a particular print statement would result in an error, you may simply write “error” as the output. *Hint: the order of questions is not the order of execution!*

(a) (1 point) What is displayed when the print statement at /*1*/ executes?

(a) ___________

(b) (1 point) What is displayed when the print statement at /*2*/ executes?

(b) ___________

(c) (1 point) What is displayed when the print statement at /*3*/ executes?

(c) ___________
(d) (1 point) What is displayed when the print statement at /*4*/ executes?

(e) (1 point) What is displayed when the print statement at /*5*/ executes?

(f) (1 point) What is displayed when the print statement at /*6*/ executes?

(g) (1 point) What is displayed when the print statement at /*7*/ executes?

(h) (1 point) What is displayed when the print statement at /*8*/ executes?

(i) (1 point) What is displayed when the print statement at /*9*/ executes?

2. Linux Commands. The following questions reference the following directory hierarchy. Assume that you can create any file/directory in any directory. If a command cannot be executed or would result in an error, you may simply write “error” as your answer.

(a) (2 points) You want to create a new directory named labs inside the rina directory. Give a command which will accomplish this task.

(b) (1 point) Which Linux command allows you to change the current (working) directory?

(c) (2 points) You are currently in the lucia directory. Give a relative path to the song.mp3 file.

(d) (2 points) You are in the home directory. Give an absolute path to the HW2.java file.
3. Expression Evaluation: evaluate each expression and write your answer in the space provided. **Clearly indicate the data type of your answers!** For example, 4.0 is different then 4 is different then "4.0". The first is a double, the second is an int, and the last is a String. If the expression cannot be evaluated or is not proper Java syntax, you may simply write “error” as your answer.

```java
String s1 = "CS170", s2 = "2016";
int i1 = 2, i2 = 7, i3 = 20;
double d1 = 0.5, d2 = 9.6;

(a) (1 point) i1 != 4 || 8
(b) (1 point) (int) d2 + d1
(c) (1 point) s1 + "\\" + s2
(d) (1 point) i3 % i2
(e) (1 point) s2 + i2
(f) (1 point) s1 + d1 * i1
(g) (1 point) s1 - i2
(h) (1 point) i2 / i1
(i) (1 point) i1 < i3 && i3 > i2
(j) (1 point) !(d2 < i3) || i1 <= d1
```

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.
4. Short answer

(a) Consider the following code snippet:

```java
int a = //some initial value;
int b = //some initial value;

int val = 0;
if (a >= 55) {
    if (b <= 18) {
        val = 1;
    } else {
        val = 2;
        val = 3;
    }
} else if (a <= 20) {
    val = 4;
} else {
    val = 5;
}
System.out.println(val);
```

If any of the following are not possible, you may just write “impossible” or “none” as your answer.

i. (2 points) Give an initial value for \(a\) and \(b\) which would cause “0” to be printed.
\[ a: \quad b: \]

ii. (2 points) Give an initial value for \(a\) and \(b\) which would cause “1” to be printed.
\[ a: \quad b: \]

iii. (2 points) Give an initial value for \(a\) and \(b\) which would cause “2” to be printed.
\[ a: \quad b: \]

iv. (2 points) Give an initial value for \(a\) and \(b\) which would cause “3” to be printed.
\[ a: \quad b: \]

v. (2 points) Give an initial value for \(a\) and \(b\) which would cause “4” to be printed.
\[ a: \quad b: \]

vi. (2 points) Give an initial value for \(a\) and \(b\) which would cause “5” to be printed.
\[ a: \quad b: \]

(b) (2 points) Give the maximum and minimum values which could be assigned to the variable \(num\) based on the following statement:

```java
int num = (int)(Math.random() * 29 + 4);
```

Minimum value: 
Maximum value: 

5
(c) (3 points) Consider the following method:

```java
public static void mystery(int c, int b, int a) {
    System.out.println("a is "+a+", b is "+b+" and c is "+c);
}
```

What is the output given the following variables and method call?

```java
int a = 2;
int b = 4;
int c = 8;
mystery(b, c, a)
```

(d) (3 points) Use the same method `mystery` as in the previous part (part c). Using explicit values (not variables!), write a method call which would result in the output “a is 13, b is 0, and c is 7”

```java
int a = 13;
int b = 0;
int c = 7;
mystery(b, c, a)
```

5. (6 points) Errors. You are writing a method which prints a cheering message followed by the name of the person to be cheered. Based on an integer input parameter, `cheerType`, you want to print one of three different messages: If the parameter is 1, the method should print “All the best”. If the parameter is 2, the method should print “You rock”. If the parameter is 3, the method should print “Congratulations”. If the parameter is any other value, the method should not print anything.

However, the code contains 3 errors. Circle the errors, fix them by writing a correction next to the problematic line, and indicate whether each error was a syntax error, a runtime error, or a logic error.

```java
public static void cheer(String name, int cheerType) {
    if (cheerType = 1) {
        System.out.println("All the best " + name);
    } else if (cheerType == 2) {
        System.out.println("You rock " + "name");
    } else {
        System.out.println("Congratulations " + name);
    }
}
```
6. Code Writing. For each of the following, write (mostly) syntactically correct Java code to solve the given problem.

(a) (4 points) Write a method named mean that takes two integers x and y as parameters. The method returns the arithmetic mean (ie, average) of x and y as a double, preserving the appropriate decimal part of the result.

Examples:
mean(5, 2) returns 3.5
mean(-2, 2) returns 0.0
(b) (7 points) Write a method named `inchesToFeet` that takes an integer representing a number of inches as parameter. First, the method PRINTS a message in the form:

\[ X" = Y' Z" \]

Where X is the original input (in inches), and Y and Z are the equivalent amount of feet and leftover inches, both as integers. Then, the method RETURNS the equivalent amount of feet including their decimal part. Hint: 1 foot = 12 inches

Examples:
- `inchesToFeet(75)` will print 75" = 6' 3" and return 6.25
- `inchesToFeet(60)` will print 60" = 5' 0" and return 5.0
(c) (7 points) In an enchanted kingdom there are 4 types of creatures: dragons, fairies, unicorns, and ogres which conform to the following rules:

- A dragon has 1 horn, can fly, and weighs more than 200 kg.
- A fairy has no horns, can fly, and weighs less than 10 kg.
- A unicorn has 1 horn, cannot fly, and weighs more than 75 kg.
- An ogre can have any number of horns, cannot fly, and weighs between 20 and 75 kg (inclusive).

Write a method named `creature(int horns, boolean canfly, double weight)` that takes an `int` (representing the number of horns), a `boolean` (representing whether or not the creature can fly), and a `double` (representing the creature’s weight), and returns a `String` representing the name of the type of creature, or “unknown” if the method cannot determine the type of creature based on its arguments.