

[You may use a single side of an 8.5×11 in sheet of paper for reference.] Show all of your work clearly in the space provided. Be sure to **read each problem carefully**. Note that the exam is double sided. Look over the entire exam before starting. Be sure to pace yourself.

1. (15 points) True/False (T or F)

- _____ A minus sign in a UML class diagram indicates `private`.
- _____ A `static` method is underlined in a UML class diagram.
- _____ If a class attribute is annotated with “`final`” in a UML class diagram it is constant.
- _____ Each object has its own copy of an instance variable.
- _____ The Java keyword `this` can only be used inside of an instance method.

- _____ The Java keyword `this` can be used to call another constructor from the same class.
- _____ A class method can access instance variables.
- _____ An instance method can access class variables.
- _____ Instance methods can access private attributes of other objects from the same class.
- _____ If the CSC1110 coding standard is followed, a variable, `LENGTH_OF_EXAM`, must be initialized where it is declared.

- _____ Multiple debugger break points can be set simultaneously.
- _____ Since private methods are not visible to other classes, they should be avoided.
- _____ The following creates an array of three `int` values: `new int[3]`.
- _____ It is possible to create an array of `ArrayLists`.
- _____ Short-circuit evaluation can appear in `if`, `while`, and `for` statements.

2. (10 points) Indicate which line in the code below the following concept is represented. If the concept is not represented in the code, place an **X** in the blank instead of a line number.

_____ Default Constructor	1 <code>public class ExamI {</code>
_____ Class variable	<code>public static void main(String[] args) {</code>
_____ Instance method	3 <code>Student ed = new Student("Ed");</code>
_____ Class constant	<code>System.out.println("Hi " + ed.getname());</code>
_____ Helper method	5 <code>}</code>
_____ Visibility modifier	<code>}</code>
_____ Local variable	7 <code>public class Student {</code>
_____ Short-circuit evaluation	9 <code>private String name;</code>
_____ Pre-increment	<code>private final int id;</code>
_____ Array	11 <code>private static int numStudents = 0;</code>
	13 <code>public Student(String name) {</code>
	<code>this.name = name;</code>
	15 <code>id = ++numStudents;</code>
	<code>}</code>
	17 <code>public String getName() {</code>
	19 <code>return name;</code>
	<code>}</code>
	21 <code>public void setName(String name) {</code>
	23 <code>this.name = name;</code>
	<code>}</code>
	25 <code>}</code>

3. (5 points) Draw the memory diagram showing everything in memory at line 4 of main().

4. (5 points) Draw the memory diagram showing everything in memory as seen in the `getName()` method after being called on line 4 of `main()`.

5. (5 points) Indicate what will be displayed by the following code.

```
int num = 3;
System.out.println(num++);
for (int i = num; i <= num; i++) {
    int j = i + 3;
}
System.out.println(++num);
char a = 'a';
char b = 'b';
System.out.println(a - b);
b++;
System.out.println(b + " year");
```

6. (10 points) Implement a method, `sumLength()` that accepts an array list of strings and returns the sum of the length of all of the strings in the list.

7. (10 points) Implement a method, `grow()`, that accepts an array of `Scanner` objects and returns an array that is twice the size where every other reference in the array is set to `null`.

8. (30 points) Implement the **Student** class used in the following program so that it produces the output shown below. You should include everything in the **Student.java** file except for import statements and comments.

```
public static void main(String[] args) {  
    Student[] students = new Student[4];  
    students[0] = new Student();  
    System.out.println("Student 1: " + students[0].toString());  
    students[1] = new Student("Jon Dough");  
    System.out.println("Student 2: " + students[1].getLastName());  
    students[2] = new Student("Ed", "Edwards");  
    students[3] = students[1];  
    System.out.println("Student 3: " + students[2].toString());  
    students[0].setFirstName("Jane");  
    students[0].setLastName("Do'h");  
    System.out.println("Student 1: " + students[0]);  
    System.out.println("Student 4: " + students[3]);  
}
```

```
Student 1: First Last  
Student 2: Dough  
Student 3: Ed Edwards  
Student 1: Jane Do'h  
Student 4: Jon Dough
```



8. continued...