

[May use one 8.5 × 11 inch sheet of paper for notes.] Show all of your work clearly in the space provided or on the additional page at the end of the exam. If the additional page is used, clearly identify to which exam question it is related. Be sure to **read each problem carefully**. Note that the exam is double sided.

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

- \_\_\_\_\_ If `import java.lang.String` is not present before the class declaration, all occurrences of `String` must be replaced with `java.lang.String`.
- \_\_\_\_\_ If a method is overloaded, it means that there are two methods with exactly the same name that return different types (e.g., one returns `String` while the other returns `double`).
- \_\_\_\_\_ Local variables declared in a constructor are only accessible within that constructor.
- \_\_\_\_\_ A `private` attribute is accessible to all methods defined within the class.
- \_\_\_\_\_ It is illegal to place `this` on the left side of the assignment operator. E.g.,  

```
    this = that;
```
- \_\_\_\_\_ A constructor may call another constructor.
- \_\_\_\_\_ A class may have two attributes with exactly the same name as long as they are from different types and one of them is declared with `this.` in front of it. E.g.,  

```
    private double number;
    private int this.number;
```

2. (10 points) Give an example of a method implementation that makes use of the keyword: `this` to disambiguate between a local variable and an attribute.

```
public class Question2 {
    private double number;
```

```
}
```

3. (10 points) Identify the error in the following class and explain why it is an error:

```
public class Exam2 {

    private int questionNumber = 3;

    public static void main() {
        System.out.printf("The number of this question is :%d", questionNumber);
    }
}
```



4. (20 points) Recall the `Complex` class developed in lecture. The class contained two `private` attributes: `real` and `imag` (both `doubles`). Implement the `toString()` method for the class such that it produces output consistent with the following examples:

- `3.2` when the real component is 3.2 and the imaginary component is 0.0.
- `3.2 + i5.7` when the real component is 3.2 and the imaginary component is 5.7.
- `3.2 - i5.7` when the real component is 3.2 and the imaginary component is -5.7.
- `i5.7` when the real component is 0.0 and the imaginary component is 5.7.
- `-i5.7` when the real component is 0.0 and the imaginary component is -5.7.



5. (10 points) Show what will be displayed by the following code:

```
System.out.printf("%2s_%8s\n", "nice", "exam");  
System.out.printf("%08d_%8.2f\n", 8381, 3.577);
```

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4

6. (15 points) Create a UML class diagram for a `Fraction` class that represents a fraction with a numerator and denominator and, if implemented, would allow the following code to run:

```
public static void main(String[] args) {  
    Fraction f1 = new Fraction(1, 2);  
    Fraction f2 = new Fraction(2, 1);  
    Fraction f3 = new Fraction();  
    System.out.println("Math alert: " + f1 + "+" + f2 + "=" + f1.plus(f2));  
    System.out.println("Math alert: " + f1 + "-" + f3 + "=" + f1.minus(f3));  
}
```

producing the following output:

```
Math alert: 1/2 + 2 + = 5/2  
Math alert: 1/2 - 1 + = -1/2
```



7. (20 points) Partially implement the `Fraction` class from the previous problem. The class and the `main` method in the previous problem should compile without errors and all necessary constructors should be fully implemented, but any remaining methods do not need to be functional. For example, if you need a method called `cabbage` that returns a `double`, your implementation would look like this:

```
public double cabbage() {  
    return 0.0;  
}
```



Additional space — identify which problem your work is associated with.