1. Apply your knowledge of Java’s precedence relations to determine to answer the following questions:

(a) What is printed by the following statement:
   ```java
   System.out.println( 2 * 4 / 4 + 2 );
   ```
   (a) ____________

(b) Add parentheses to the expression
   2 * 4 / 4 + 2
   so that it computes 6.

2. For this question and its parts, use the following definitions:

   ```java
   public static int op( int x, int y ) {
       return x + y;
   }
   
   public static double op( double x, double y ) {
       return x * y;
   }
   
   public static double op( int x, double y ) {
       return 2 * ( x + y );
   }
   ```

   (a) What is the result of evaluating:
   ```java
   System.out.println( op( op( 1, 2 ), 2 ) );
   ```
   (a) ____________

   (b) What is the result of evaluating:
   ```java
   System.out.println( op( op( 1, 2 ), 2.0 ) );
   ```
   (b) ____________
2 Questions about Classes

1. Recall that any attempt to divide a number by 0 throws an ArithmeticException. The class ArithmeticException is a subclass of Java’s RuntimeException class.

With this in mind, assume that the statement badMethodCall() in the code fragment below always generates various ArithmeticExceptions, but in no particular order, but at least one of these is a divide by zero exception.

```java
try {
    badMethodCall(); // generates several Arithmetic Exceptions,
    // but in no particular order.
} catch( <exception 1> ) {
    return;
} catch( <exception 2> ) {
    return;
}
```

Assume that the programmer writes:

```java
try {
    badMethodCall();
} catch( RuntimeException re ) {
    System.err.println("Caught a runtime exception.");
    return;
} catch( ArithmeticError ae ) {
    System.err.println("Caught an Arithmetic exception.");
    return;
}
```

Which of the following statements is true?

A. Sometimes the program prints "Caught an Arithmetic exception", but sometimes the program prints "Caught a runtime exception."
B. The program always prints "Caught a runtime exception."
C. The program always prints "Caught an Arithmetic exception."
D. We do not have enough information to choose from any of these possibilities.
2. Given the following partial definitions:

```java
public class Person implements Comparable<Person> { ... }

public class Student extends Person {
    private String studentID;
    ...
}

public class Teacher extends Person {
    private String facultyID;
    ...
}
```

(a) Write the `equals` method on the class `Person`:

```java
// Implementation
```

(b) 

(c) Override the `equals` method on the `Teacher` class so that two `Teachers` are equal if they are `equal` according to the class `Person` and have equal `teacherIDs`. 

```java
// Implementation
```
3 Using Arrays and ArrayLists . . .

1. In the space below, write the rightPartition method that takes an array of Comparable objects and a Comparable object called pivot. The rightPartition method then returns a new ArrayList that contains all of the objects from the original ArrayList that are greater the pivot.

```java
public static ArrayList<Comparable> rightPartition(
    Comparable[] array, Comparable pivot ) { // begin here
```
2. Write the `eitherOr` method that takes two `ArrayLists` of distinct integers, meaning that no integer appears more than once in either list, and returns a new `ArrayList` of integers that appear in one or the other. For example:

```java
eitherOr( [1,2,3], [2,3,4] ) => [1,2,3,4]
eitherOr( [], [1,2,3] ) => [1,2,3]
eitherOr( [3,2,1], [1,2,3] ) => [1,2 3]
eetc.
```

Note: for this question, the order of elements in any of these `ArrayLists` is unimportant. You may assume that each parameter is an `ArrayList` that contains no duplicate integers.

You may only use the `ArrayList` operators discussed in class in your implementation.

```java
public static eitherOr(
    ArrayList<Integer> list1, ArrayList<Integer> list2 ) {
```