The first quiz for the course will be on Wednesday, Feb 7 during your lab session. The following list provides more information about the quiz:

- The quiz will be a written quiz (no computer).
- Closed book, closed notes quiz.
- Answers must be neat and legible. We recommend that you use pencil and eraser.

The following exercises cover the material to be included in this quiz. Solutions to these exercises will not be provided, but you are welcome to discuss your solutions with the TA or instructor during office hours.

**Exercises**

1. What is encapsulation? How does it relate to abstraction?

2. What is the difference between procedural abstraction and data abstraction?

3. Define an enumerate type named `Day` that represents the days of the week. Using the enhanced for loop construct, write a code fragment that prints all the days of the week.

4. Using the `Day` enumerated type you defined above, define a static method called `randomDay` that returns a random day.

5. Is the following code fragment legal? Briefly explain.

   ```java
   ArrayList<Boolean> L1 = new ArrayList<Boolean>();
   ArrayList<Object> L2 = L1;
   ```

6. The `Student` class is defined as follows:

   ```java
   public class Student {
       private String name;
       private int id;

       public Student(String name, int id) {
           this.name = name;
           this.id = id;
       }

       public String toString() {
           return "Name: " + name + " Id: " + id;
       }
   }
   ```

   Rewrite the following code fragment using generics and the new for loop construct.

   ```java
   ArrayList L = new ArrayList();
   L.add(new Student("Mary", 10));
   L.add(new Student("John", 5));
   L.add(new Student("Kelly", 7));
   for (int i=0; i<L.size(); i++)
       System.out.println(L.get(i));
   ```
7. The WebSite class is defined as follows:

```java
public class WebSite {
    private String name;
    private int links;

    public WebSite(String name, int links) {
        this.name = name;
        this.links = links;
    }

    public String toString() {
        return "Name: " + name + " Links: " + links;
    }
}
```

a. Modify the class so it implements the Comparable interface, allowing you to compare WebSite objects based on their name.

b. Implement a comparator class that allow us to compare WebSite objects based on the number of links.

8. Implement a method named `compareStrArrays` that has the following prototype:

```java
public static boolean compareStrArrays(String[][] array1, String[][] array2);
```

The method will return true if the arrays have the same dimensions and the same String values. Keep in mind that each array can have rows with different lengths.

9. The PrinterJob class is defined as follows:

```java
public class PrinterJob {
    private int id;
    private int size;

    public PrinterJob(int id, int size) {
        this.id = id;
        this.size = size;
    }

    public int getId() { return id; }
    public int getSize() { return size; }

    public String toString() {
        return "Id: " + id + " Size: " + size;
    }
}
```

a. Add a new private field "jobType" as an enumerated type with the values Color and BW.

b. Modify the class so it implements the Comparable interface, allowing you to compare PrinterJob objects based on their id.

   c. Implement a comparator class named sizeComparator, that allow us to compare PrinterJob objects based on their size.
d. Use generics to implement a version of the ArrayList class that only accepts objects of class PrinterJob.

e. Implement an enhanced for loop that prints all PrinterJob objects in the customized ArrayList above.