EECS2030: LAB 8 Due: Nov 30th, 2021- 11:59 pm

The purpose of this lab is to practice working with Generics as well as practice comprehending Junit test cases.

1. Setup

Please download `Lab8.zip` that is attached to this description.

- Open `eclipse`.
- Click on *`File`* and select *`Import`*.
- Choose *`Existing Projects into Workspace`* and click *`Next`*.
- Click on *`Select Archive File`* and then *`Browse`*. Find `Lab8.zip` and click *`Finish`*.
- Please make sure that you do not already have a project called `EECS2030_Lab8`, otherwise, eclipse cannot import it for you.

You should see two files, one is called `Utility.java` and one `UtilityTester.java`.

2. Programming Task

For this lab, we are asking you to create a parameterized class called `Utility` and decide on its methods' signature.

Class Utility is a class that contains 4 methods. The implementation of the methods is given below. The only thing that you need to do is to decide on the correct method signature so that all the test cases are passed.

- `linearSearch` is a method that linearly searches for the given `item` (i.e. the input parameter) in `list`. The `list` is the instance variable of class `Utility`. You should decide on what is the correct method signature when the implementation looks like below:

```
int position = -1;
for (int i = 0; i <list.size(); i++) {
      if (item.compareTo(list.get(i)) == 0) {
            position = i;
            break;
      }
}
return position;
```

- `mergeList` is a method that gets a List of 'some type' as its input parameter and add all the items that exist in the instance variable to the end of the `list`. The code that you need to use as the implementation of this method is as follow:

```
for (T obj: this.list)
      list.add(obj);
```

- `containList` is a method that gets a list of 'some type'. If the instance variable of this class contains all the elements of the input list, it adds all the elements of the input to the instance variable and returns true. The body of this method looks like below:

```
if (this.list.containsAll(list)) {
      this.list = new ArrayList<T>();
      for (T obj: list)
            this.list.add(obj);
      return true;
}
else
      return false;
```

- `removeZero` is a method that does not work on the instance variable of this class. It gets a list and removes all the zeros from the list. Here is the implementation of the method:

```
for (int i = 0; i < myList.size(); i++) {
      if (MyInteger.isZero(myList.get(i))) {
            myList.remove(i);
            i--;
      }
}
```

In addition to the above methods, you should implement the constructors of the class in a way that does not fail the test cases.

A few other classes can be found in `Utility.java` that are used for testing the correctness of your code. These classes are called `Point`, `MyInteger`, `NaturalNumber`, `OddNumber` and `EvenNumber`.

For this lab, we do not provide any javaDoc on purpose, because the implementation of the methods are given. Besides, we don't want to give any hint on the signature of the methods.

3. Submit

You only submit one file that is called `Utility.java` via eClass by clicking on the lab link. Although you are encouraged to write the javaDoc, you do not need to submit the HTML files.