ITI 1121. Introduction to Computer Science II

Laboratory 5
Summer 2016

- Further understanding of interfaces
- Implementation of generics using collections of type vector and list

Part I

Interface

1 Combination

Let's revisit the class Combination from laboratory 1. Make all the necessary changes to the class **Combination**
(see below) so that it implements the interface **java.lang.Comparable.**

```
public interface Comparable<E> {
// Compares this object with the specified object for order.
// Returns a negative integer, zero, or a positive integer
// as this object is less than, equal to, or greater than the
// specified object.
public int compareTo( E other );
}
```

When two Combination objects are compared, the first pair of values is compared first, if the value of this object is smaller than that of the other then the method will return -1, if it is greater, then it will return 1, if those values are equal, then the second pair of values is considered,the value -1 or 1 will be returned if the second value of this object is smaller than that of the second, finally, if those values are also equal then the third pair of values is considered.

Comparable is part of a package, called lang, that is imported by default. Therefore, no need to import Comparable.
Write a test program for thoroughly testing your implementation.
File:

- Combination.java

Part II

Generics

The laboratory deals with the implementation of generics using collections of type vector and list.

Generics enable *types* (classes and interfaces) to be parameters when defining classes, interfaces and methods. Much like the more familiar *formal parameters* used in method declarations, type parameters provide a way for you to re-use the same code with different inputs. The difference is that the inputs to formal parameters are values, while the inputs to type parameters are types.

Vector internally is an **array-based** data structure (linked list is **node-based**) and grows dynamically when more elements are included. A vector comes with two properties – **size** and **capacity**. When a vector object is created, it comes with a default capacity that can store 10 elements.

Write a programme regarding implementation of vector generics using following instructions:

1. Create a class vectorGenerics and make necessary imports to use classes of collections interface. These classes are part of util package in java.

2. Declare a vector vect1 that stores only integer values and another vector vect2 that stores only String values.

3. Use following methods of collections to carry out operations on the declared vectors.

4. add: to add elements to the declared vector. Make sure the type of elements added comply with the type of elements of respective vector.

5. max and min : to calculate maximum and minimum value in both vectors.

6. addAll: declare another vector vect3 that stores integer values and add few elements to it. Then, add all elements of vect3 to vect1.

7. reverse: to reverse the values of vect3.

8. shuffle: to shuffle the values in vect1.

9. rotate: rotate the values in vect1 by three places.

10. fill: fill the values in vect1 by the integer 100.

11. clear: clear vect1.