Lab No. 8 – Generics
Date: 29/12/2007 – 2/1/2008

كل عام وأنتم بخير ، وتقبل الله منا ومنكم صالح الأعمال ☺

Exercise 1

A. Write the class **NumberChecker** that has only one generic method **isExist** which receives an array of numbers (array) and a single number (x) and check if (x) exists in the (array) or not.

boolean isExist(E [] array, E x)

B. Write the class **Checker** that inherits from **NumberChecker** and overloads the generic method **isExist** with a non-generic one that can check if a string exists in array of strings.

C. Using the class **Checker**, write a test class to verify that it can find an element in different kind of arrays (Integers, Doubles, Strings).

Exercise 2

A. Create a generic class **Pair<F, S>** and implement the following:
constructor: accept two parameters of type F and S
setFirst: set first parameter (of type F)
getFirst: get first parameter (of type F)
setSecond: set second parameter (of type S)
getSecond: get second parameter (of type S)

B. Create a generic class **Triplet<F, S, T>** that extends **Pair** class and implement the following:
constructor: accept three parameters of type F, S, and T
setThird: set third parameter (of type T)
getThird: get third parameter (of type T)

C. Using the class **Triplet**, write a test class to verify that you can store different arrangement of triplet data types.

Exercise 3

A. Implement the generic class MyList<E>

MyList<E>
- list: E [] - size: int
+ MyList() + MyList(int size) + add(E element): boolean + remove(int index): boolean + getElement(int index): E

B. Using the generic class MyList<E>, Implement the following classes