OOSD: Practice Problems 9

1. Write a method with the signature

   static Object [] mkArray(Collection c)

   that takes a Collection as its parameter and returns an array containing all of the Objects in the Collection. Do *not* use the method *toArray*.

2. Write a method with the signature

   static List getNums(String fileName)

   that, assuming fileName is the name of a textfile containing integers separated by whitespace, reads the textfile and returns an ArrayList containing Integer objects corresponding to each of the numbers in the file.

3. Write a Java method, called *mkNumbers*, that takes a List of Double objects as its parameter and creates a binary file, called *numbers*, that contains all of the numbers in the List as 64-bit binary values. No validation of the parameter is required.

4. Write a method with the signature

   static void storeNums(List nums, String fileName)

   that, assuming nums is a List containing mostly Integer objects, fills a binary file of **int**'s, whose name is given by the second parameter, with all the values of the Integer objects in the List. Since the List may have some other objects, just ignore those that are not Integer objects. Be certain to catch all checked exceptions in the method.

5. Write a mutator method that takes a List object as its parameter and removes the duplicate objects in the list object. This method will be a procedure.

6. Write a function that takes a List object as its parameter and returns the List with all duplicate objects removed. How does this method compare in efficiency with the method in problem 5?

7. Suppose that the variable *strings* refers to a List of distinct String objects. Write a code fragment that creates new List that contains those same String objects in alphabetical order. Do not write a sorting routine. Hint: Use a SortedSet.

8. A Map stores pairs of the form (*key*, *value*).

   Write a Java method that takes two arrays of Object as its parameters and creates and returns a Map whose *keys* are taken from the first array and whose *values* are taken from the corresponding positions of the second array.

   If the arrays are not of the same length, throw an IllegalArgumentException.
   If two keys are the same, throw an UnsupportedOperationException.
   Both of those exception classes are defined in *java.lang*

   Map mkMap(Object [] keys, Object [] values)

9. Redo problems 3 and 7 using generic containers.

10. Redo the problems MakeAL, MyLinkedList, SortDoms, UniqueWords, and Frequency using autoboxing and generic containers.