```
CS 6301.002.  Implementation of advanced data structures and algorithms
Spring 2016
Short Project 0h (Hashing)
Wed, Feb 24, 2016

Ver 1.0: Initial description (Feb 24, 1:00 PM).

Due: 1:00 PM, Thu, Mar 3.
```

Instructions for submitting project.

This is an OPTIONAL assignment. First solution will be graded out of 10. Each additional solution: 1 point.

a. Removing duplicates

Given an array of unsorted objects of some class (that implements hashCode and equals), move the distinct elements of the array to the front. The function behaves as follows. Let k be the number of distinct elements of A (k is not known). Find the k distinct elements of arr[], and move them to arr[0..k-1]. Return k. Use hashing to implement the algorithm in expected O(n) time. Signature: public static int findDistinct(T[] arr)

b. Find the most frequently appearing element

Write a function that takes as parameter an array of integers, and returns an integer that is most frequent in the array. Signature: public static int mostFrequent(int[] arr). Compare the performance of an O(nlogn) algorithm that sorts the array with Arrays.sort and then finds the element, with a solution using HashMap that runs in O(n) expected time.

c-f. Comparison of hashing algorithms

Compare performance of Java's hash tables (HashSet/HashMap in Java's library) with any of the advanced hashing algorithms from class: 2-Choice, Cuckoo, Hopscotch, Robin Hood. Each will be considered as a separate problem for grading purposes. Use an overflow table (implemented with Java's hash map/set), if needed.