CS3230 Assignment 1

Zhuang Jianning
A0214561M

1. Determine, with proof, the minimum number k with the following property: (7 points)

   There exists a comparison-based algorithm that can decide using at most k comparisons whether any given 10-element array of integers contains only equal numbers.

comparisons are $x<y$, $x>y$, $x==y$

Given an array A of n integers, let EQUAL denote the computational problem of determining if the array A only contains equal numbers.

Claim 1. There exist a comparison based algorithm for EQUAL that uses at most n-1 comparisons in the comparison model for any input A

Proof. Define an algorithm RUNTHRU(A) with the following pseudocode:

```
RUNTHRU (A)
  n = A.length
  for i = 2 to n
      if !(A[i] == A[1])
          return false

  return true
```

In the worst case when all the elements are equal such that the algorithm does not break within the for loop, RUNTHRU has to make n-1 comparisons. Hence the upper bound in Claim 1 is tight

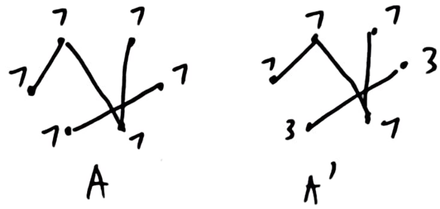
Claim 2. For any comparison-based algorithm correctly solving EQUAL on any input A, n-1 comparisons is the minimum upper bound.

Proof. Suppose M is an algorithm that correctly determines EQUAL on any input using < n-1 comparisons. Construct a graph G on input A with n nodes, where there is an edge between nodes i and j iff M compares A[i] and A[j]. Since G has < n-1 edges, it is disconnected.

There exists a partition of the nodes into C_1 and C_2 such that for any $i \in C_1$, $j \in C_2$, there is no edge between i and j. Hence $A[i]$ and $A[j]$ are not compared by M.

Suppose A has all elements equal while another array A' has all elements in C_1 equal and all elements in C_2 different from elements in C_1 but equal to each other in C_2.

M cannot distinguish between A and A', hence must err on either A or A' which is a contradiction. Hence, any comparison based algorithm that correctly solves EQUAL on any input must have an upper bound of at least n-1 comparisons.



Based on claim 1 and 2, n-1 is the minimum upper bound.

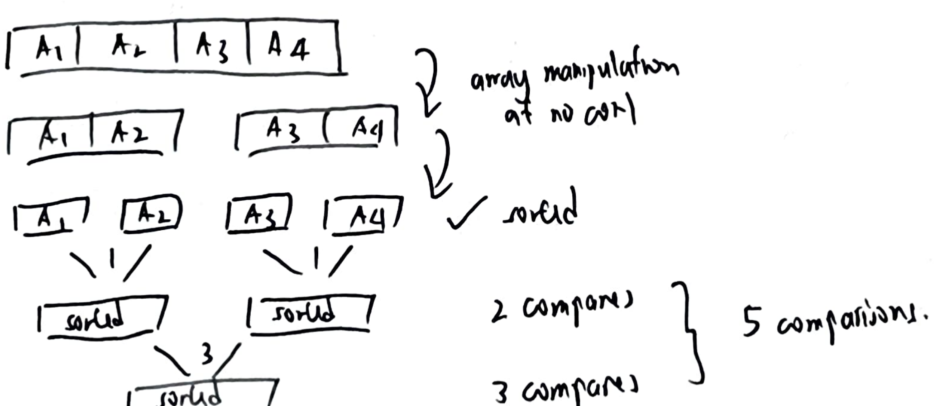
Minimum value of $k = 10-1 = 9$

2. Prove that there exists a comparison-based sorting algorithm that can sort any 4-element array of distinct integers using at most 5 comparisons (1 point)

Given an input array A of n distinct numbers, where n is a power of 2, MERGESORT can sort the array in $n\lg n - n + 1$ comparisons.

when $n = 4$, $4 \cdot \log_2 4 - 4 + 1 = 8 - 4 + 1 = 5$

MERGESORT divides A into 2 equal halves, recursively sorts the 2 halves and merges them into 1 sorted array. Given 2 sorted arrays of length n/2 each, the number of comparisons to merge them is $\leq n-1$.



3. Given 16 gold coins, with 2 fakes that weigh less than the real coins. Prove you can divide the 16 coins into 2 subsets of 8 coins, with each subset containing exactly 1 fake coin, using at most 3 weighings.

1st weighing. Split 16 coins into 2 equal subsets of 8 coins each

If they weigh the same, each subset must contain exactly 1 fake ✓

✓ OR 4 4 8
real real ?

Else, for the heavier subset, split into 2 subset of 4 coins. These 8 coins are all real

2nd weighing. Split the lighter subset of 8 coins into 2 subsets of 4 coins. If they weigh the same, they must each contain 1 fake each. Add each subset to the 4 real coins we put aside. ✓

✓ OR 6 6 4
real real ?

Else, for the heavier subset, split into 2 subsets of 2 coins. These coins are all real.

3rd weighing Split the lighter subset of 4 coins into 2 subsets of 2 coins. If they weigh the same, each must contain 1 fake. Add each subset to the 6 real coins we set aside. ✓

8 = 8
1 real 1 real
1 fake 1 fake

Else, the lighter subset must contain both fake coins. Split the remaining coins into 1 real & 1 fake and add to the 6 coins ✓

8 < 8
2 fake 2 real