Lab 6

1. Show that any comparison-based algorithm to sort 4 elements requires at least 5 comparisons in the worst case.

2. Devise an algorithm that arranges the elements of a length-n integer array according to the following scheme:
   position 0: the smallest integer
   position 1: the largest integer
   position 2: the second smallest integer
   position 3: the second largest integer etc.
   For example, this algorithm would arrange the input array {1, 2, 17, -4, -6, 8} as follows: {-6, 17, -4, 8, 1, 2}. (Notice that –6 is the smallest, 17 the largest, -4 second smallest, 8 second largest, etc.) Then answer the following:

   A. What is the asymptotic running time of your algorithm?
   B. Prove that it is *impossible* to obtain a comparison-based algorithm to sort an integer array, in the way described above, that performs better than $\Theta(n \log n)$.

3. Carry out the steps of RadixSort to sort the following {80, 27, 72, 1, 27, 8, 64, 34, 16} – Hint: use 9 for your radix.

4. Describe an $O(n)$ algorithm that does the following: Given an input array of *n* integers lying in the range 0 .. 3n - 1, the algorithm outputs the first integer that occurs in the array only once. (You may assume that each input array contains at least one number that has no duplicates in the array.) Explain why your algorithm has an $O(n)$ running time.
   Example: If the input array is [1, 2, 4, 9, 3, 2, 1, 4, 5], then the return value is 9 since 9 is the first integer that occurs in the array only once.