**Course: Algorithm**
**Prof. Prem Nair**
**Student: Binh Van Tran**
**ID: 986648**
**Homework: Lab 7**

1. **Question 1 –** Practice Radix Sort {179, 721, 639, 549, 292, 427, 335, 435, 62} with radix is 9

| **Key % 9** | **(Key /9) % 9** | **Key / 9 /9** |
|---|---|---|
| 0: 639; 549 | 335; 179 | 62 |
| 1: 721 | | |
| 2: 335 | 427 | 179 |
| 3: 435 | 435 | 292 |
| 4: 292; 427 | | 335 |
| 5: | 292 | 427; 435 |
| 6: | 62 | 549 |
| 7: | 549 | 639 |
| 8: 179; 62 | 639; 721 | 721 |

2. **Question 2 -** Experimenting with lower bound
   As the theorem of lower bound in compared-based sorting algorithm we have
   For an input of *n* elements, the representative decision tree **T** has
   **T** has n! leaves
   Height of the tree: h >= *log(n!*)
   Number of comparisons: >= *log(n!)*

   In this question, we have n = 4, so the number of comparisons should be at least log(4!) = 3 + log3 = 4.5 is the lower bound of comparisons that the algorithm need to make.
   5 of comparisons is a proper value, therefore this doesn't violate the theoretical lower bound.
3. **Exploring new ideas**: Forward and backward sorted array (FBS array)
   - Step 1: Sort the array ascendingly
   - Step 2: Reverse the values at the odd indexes descendingly
   - Step 3: Put even index value & odd index value in right order

```
fbs(A, n)
    Input array A of n integers
    sortedArray ← quickSort(A, 0, n)
    j ← n % 2 == 0? n − 1 : n − 2
    for i ← 1 to n/2 do
        if A[i] < A[j] then
            swap(A, i, j)
        i ← i + 2
        j ← j + 2
```

```
for i ← 0 to n − 1 do
    for j ← 1 to n − 1 do
        if A[i] > A[j] then
            swap(A, i, j)
        j ← j + 2
    i ← i + 2
```

As we can see:
Step 1: QuickSort take *nlogn*
Step 2: take *n/2*
Step 3: take $n^2/4$
Step 1 + Step 2 + Step 3 = $nlogn + n/2 + n^2/4$
So, we can say time complexity is $O(n^2)$