ESO207 Assignment 2
Submit on Friday 16/06/2017 at the start of the class.

Please write clearly.

Question 1. [Marks].
Consider the following pseudocode which accepts input from array A containing numbers.

Initialize empty queue Q;
Initialize empty stack S;
for $i := 0$ *to* $n-1$ **do**
  $Enqueue(A[i], Q)$;
end
while $\neg Empty(A)$ **do**
  $x := Dequeue(Q)$;
  if $\neg Empty(S)$ *AND* $Top(S) > x$ **then**
    $y := Pop(S)$;
    $Enqueue(y, Q)$;
  end
  $Push(x, S)$;
end
for $i := 0$ *to* $n-1$ **do**
  $y := Pop(S)$;
  Output y;
end

(a) What does this algorithm compute?
(b) Construct the worst case instance of the input that requires maximum time. Determine the worst case time complexity.
(c) Prove the correctness of this algorithm by defining an invariant for the While-loop.
Hint: Let $S = a_1a_2a_3 \ldots a_n$ be a sorted sequence and $T = b_1b_2 \ldots b_k$ be a subsequence. The first hole in T is the first range of contiguous indices missing from T. For example let $S = 11, 12, 13, \ldots, 20$. If $T = 11, 12, 16, 18$, then the first hole is $3 \ldots 5$. If $T = 11, 12$, then the first hole is $3 \ldots 10$. If $T = 13, 14, 17$, then the first hole is $1, 2$. Similarly you can define second hole etc. This concept of the first hole may be useful in the design of an invariant.

Question 2. [Marks].
In order to implement the queue data structure using an array $A[0 : n-1]$, write the pseudocode for the following operations.
(a) Enqueue
(b) Dequeue
(c) Empty
(d) Full

Question 3. [Marks].
In the substitution based analysis of Quick Sort we used

$$T(n) \leq c_1 n + (2/n) \sum_{i=0}^{n-1} T(i) \qquad \ldots(1)$$

where the quick sort algorithm was given as

```
if S = ∅ then
│  Return ∅;
end
if S = {x} then
│  Return x;
end
Randomly select x from S;
S1 := {y ∈ S | y.key ≤ x.key};
S2 := {y ∈ S | y.key > x.key};
L1 := QuickSort(S1);
L2 := QuickSort(S2);
Return L1.L2;
```

Algorithm 1: $QuickSort(S)$

(a) Show that if multiple elements have same key, then inequality (1) does not hold.

(b) Suppose the QuickSort algorithm is modified as follows, then show that inequality is valid, hence the analysis is also valid.

```
if S = ∅ then
│  Return ∅;
end
if S = {x} then
│  Return x;
end
Randomly select x from S;
S1 := {y ∈ S | y.key < x.key};
S2 := {y ∈ S | y.key > x.key};
S3 := {y ∈ S | y.key = x.key};
L1 := QuickSort2(S1);
L2 := QuickSort2(S2);
L3 := Sequence(S3);
/* write S3 elements as a sequence                                  */
Return L1 · L3 · L2;
```

Algorithm 2: $QuickSort2(S)$

(c) Is the direct analysis (done in the class) valid for the first version of QuickSort? Justify your answer.