1. Let us have two (unbounded) stacks at our disposal, with the operations as specified in the abstract data type for stacks. Use them to implement the operations enqueue and dequeue of the abstract data type for queues.

Do this in such a way that the operation enqueue is in $O(1)$, and the operation dequeue on average is in $O(1)$. (20 pts)

2. Sort the following sequence using quicksort:

   $22 \ 36 \ 6 \ 79 \ 26 \ 45 \ 75 \ 13 \ 31 \ 62 \ 27 \ 76 \ 33 \ 16 \ 62 \ 47$

For each (sub)list, take the first element as pivot. (20 pts)
3. (a) Construct an AVL tree by successively inserting the numbers

\[ 7 \quad 9 \quad 5 \quad 12 \quad 14 \quad 16 \quad 2 \quad 3 \]

starting from the empty tree. After each insertion, restructure the tree into an AVL tree if needed.

(b) Remove the root from the resulting AVL tree, and restructure the resulting binary search tree into an AVL tree again if needed. (20 pts)

4. Apply Dijkstra’s shortest path algorithm to the following undirected graph, with start node A.

```
A -- 13 -- B
  ^       |
  |       v
  C -- 4 -- D
  |       |
  |       v
  F -- 6 -- E
  |       |
  |       v
  G -- 20 -- E
```

(20 pts)

5. Explain why it is crucial for the worst-case time complexity of Kruskal’s algorithm that in the disjoint-set data structure, when two disjoint sets are joined, the larger set subsumes the smaller set in the union-find data structure. What would be the worst-case time complexity of Kruskal’s algorithm if the smaller set subsumed the larger set? (20 pts)