Problem 1 (15 marks). Show the priority queue after a delete-min on the priority queue below:

```
  5
 / \
7   8
|   |
12  14 15 17
```

Solution.

```
  5
 / \
6   8
|   |
7   10
12  14 15 17
```

Problem 2 (15 marks). Show the priority queue after inserting the number 4 into the priority queue below:

```
  3
 / \
5   6
|   |
7   10
12  14 15 17
```

Solution.

```
  3
 / \
5   4
|   |
7   6
12  14 15 17
```

Problem 3 (15 marks). Consider the following AVL-tree. Give the sequence of nodes visited when we use the tree to find the successor of 55.

```
  40
 /  \
15   60
 / \  / \  /  
9   29 45 52 56 100
 / \ / \ \ / \ / /  
3   12 18 30 41 80 85 108
 / \ / \ / \ /  
10 17 30 50 80 85 108
```

Consider the following AVL-tree. Give the sequence of nodes visited when we use the tree to find the successor of 55.
Solution. Nodes 40, 68, 45, 50, 60.

Problem 4 (15 marks). Indicate the imbalanced nodes in the following AVL-tree (there are five of them).

![AVL tree diagram]

Solution. Nodes 40, 68, 92, 96, 100.

Problem 5 (20 marks). Consider the following AVL-tree. Suppose that we insert the integer 97 into the tree. Show the resulting AVL-tree after fixing the imbalanced nodes (using the algorithm we discussed in the lecture).

![AVL tree diagram]

Solution.

Problem 6 (20 marks). Describe how to use an AVL-tree to implement a priority queue. Recall that a priority queue supports two operations on a set $S$ of integers:

- Insert($e$): add a new integer $e$ to $S$. 
• Deletemin: remove the smallest integer from $S$.

Your implementation should perform each operation in $O(\log n)$ time where $n = |S|$.

**Solution.** For insert($e$), simply insert $e$ into the AVL-tree. For deletemin, keep descending to the left child until reaching a node $u$ with no left child. Delete the element stored in $u$. 