Quick Sort

TOTAL POINTS 4

1. What is the worst case running time of Quick Sort? **1 / 1 point**

- (•) $O(n^2)$
- ( ) $O(n \log n)$

✓ **Correct**
In the worst case, Quick Sort will always partition array of size n into parts of size 1 and $n-1$, and so it will make $O(n + (n-1) + (n-2) + \cdots + 2 + 1) = O(n^2)$ operations.

2. What is the running time of the Partition procedure? **1 / 1 point**

- ( ) $O(\log n)$
- ( ) $O(\frac{n}{\log n})$
- (•) $O(n)$

✓ **Correct**
Partition works in $O(n)$ time as it needs to compare every element to the pivot.

3. What is the amount of additional memory that regular Quick Sort uses (besides the array being sorted) in the worst case? **1 / 1 point**

- (•) $O(n)$
- ( ) $O(1)$
- ( ) $O(\log n)$

✓ **Correct**
In the worst case, the array is always divided into a part of size 1 and a part with all the other elements, and the recursion depth in this case will be $O(n)$. Recursion needs $O(1)$ additional memory for each call, so in the worst case Quick Sort will use $O(n)$ additional memory. However, by using tail recursion elimination we can make Quick Sort use no more than $O(\log n)$ additional memory. See the lecture with the final remarks about Quick Sort.

4. Which parts need to be sorted in the Quick Sort algorithm after applying the 3-way partition? **1 / 1 point**

- (•) Only the part with the elements less than the pivot and the part with the elements greater than the pivot.
- ( ) All three parts.
- ( ) Just the part with the elements equal to the pivot.
- ( ) Just the part with the elements greater than the pivot.
- ( ) Just the part with the elements less than the pivot.

✓ **Correct**
There is no need to sort the elements equal to the pivot, because they are already in the correct positions after 3-way partition.