You are expected to work on the problems before coming to the lab. Discussion sessions are not meant to be a lecture. TA will guide the discussion and correct your solutions if needed. We will not release "official" solutions. If you are better prepared for discussion, you will learn more. TAs will record names of the students who actively engage in discussion and report them to the instructor. The instructor factors-in participation in the final grade as bonus points.

1. Run Insertion-Sort on input array $A = \langle 5, 8, 4, 2, 3, 1 \rangle$. Show how the array looks before each iteration of the `for` loop in line 1; see page 18 for the pseudo-code.

2. Consider the searching problem: The input is an array $A[1 \cdots n]$ of n numbers and a value v. You are asked to find an index i such that $A[i] = v$. If there is no such index, return NIL. The following is a pseudocode of linear search, which scans through the sequence, looking for v. Using a loop invariant, prove that your algorithm is correct. Make sure that your loop invariant fulfills the three necessary properties. What is the asymptotic worst case running time?

```
1. For i = 1 to n
2.   If A[i] == v then return i
3. return NIL.
```

3. You are given an array $A[1 \cdots n]$ of n numbers, and would like to know if the array includes two equal numbers. Describe an algorithm that does this job for you. Your algorithm must be as efficient as possible.

4. We can express Insertion-Sort as a recursive procedure as follows. In order to sort $A[1 \cdots n]$, we recursively sort $A[1 \cdots n-1]$ and then insert $A[n]$ into the sorted array $A[1 \cdots n-1]$. Give a pseudocode of this recursive version of insertion sort. Write a recurrence for the running time.

```
Recursive-Insertion (i)
   If i == 1 return;
   Exercise.
```

5. Show the operation of Merge-Sort on the array $A = \langle 7, 4, 2, 8, 3, 1, 5, 6, 9 \rangle$ as shown in Fig. 2.4.

6. We are given three sorted arrays $A[1 \cdots x]$, $B[1 \cdots y]$, $C[1 \cdots z]$. Describe an algorithm that merges the given three sorted arrays into one (sorted array). What is the running time?

7. Merge-Sort is always faster than Insertion-Sort. True or false? Justify your answer.

8. Observe that if the sequence A is sorted, we can check the midpoint of the sequence against v (the element we're looking for) and eliminate half of the sequence from further consideration. The binary search algorithm repeats this procedure, halving the size of the remaining portion of the sequence each time. Write pseudocode, either iterative or recursive, for binary search. Argue that the worst-case running time of binary search is $O(\log n)$.

9. Can you use binary search to improve the running time of insertion sort to $O(n \log n)$?