Assignment 14 –Data Structures

Questions: 7, 8
Submission Deadline: May 3, 2025
Assignment Weight: 2%

Question 1 (32 points – 8 for each)

The array `PARTITION` contains the following values:

```
PARTITION:
Index:   0   1   2   3   4
Value:   A   B   A   B   A
```

You are required to answer the following questions:

a. What does the array `rpA` look like after applying the **PARTITION** algorithm?

b. What is the purpose of the **QUICKSORT** algorithm in general?

c. What is the role of the **PARTITION** procedure within the **QUICKSORT** algorithm?

d. What is the time complexity of the **QUICKSORT** algorithm in the best-case, worst-case, and average-case scenarios?

Question 2 (36 points – A:16p , B:8p, C:12p)

You are given three sorted arrays in ascending order, each of size *n*, which contain only natural numbers (positive integers) — no duplicates within each individual array.

- Suggest an efficient algorithm for finding a number that appears in all three arrays (i.e., a common element).
- What is the time complexity of your suggested algorithm?
- Is it possible to achieve better complexity?

Please explain your answer.

Question 3 (10 points)

Regarding the array:
`A = ⟨3, 7, 8, 1, 8, 3, 9, 1, 8, 7, 8⟩`

a. Demonstrate the operation of the **COUNTING-SORT** algorithm on the array `A`.

– Show the contents of the array `C` at the beginning of executing line 6 in the subroutine (as appears on page 161).

– Show the contents of the array `C` at the beginning of executing line 9.

– Show the contents of the array `B` at the end of executing the subroutine.

Note: In line 9 of the **COUNTING-SORT** subroutine (as appears on page 140 of the textbook), the `for` loop is defined as follows:

```
for j ← 1 to length[A]
```

Show that the algorithm still functions correctly.

b. Is the algorithm still stable after this change? Justify your answer.
