CSE221 – Home Assignment 1

BRAC UNIVERSITY | Spring 2025

Deadline: 13th-March-25 (Thursday) 11:59 PM | (*No Late Submission Accepted*)

Total Mark: 15

---

1. Solve the following questions about time complexity.

a. Calculate the time complexity of the following code snippet:

```
int p = 0;
for (i = n; i > 1; i -= 1) {
      for (j = 2; j <= n; j += 1) {
            p = p + n;
            if (p > (n ^ i)) // ^ sign means XOR in code
            {
                  break;
            }
      }
}
for (i = n / 2; i > 1; i /= 6) {
      for (j = 2; j <= i; j *= 4) {
            for (k = 0; k <= j; k *= 3) {
                  p = p + n / 2;
            }
      }
}
```

2. After working with the binary search algorithm, as a CSE221 student you want to explore its strength. You have several ideas in your mind. Try to modify the algorithm to implement your ideas.

a. What if you want to return the first index of the search key in case there are duplicates? For example: your search key is 13 and there are three 13s in the list.

b. Now, you also want to return the number of times the key appears in the list. Say, the search key is 54 and it occurs 4 times in indices 6,7,8,9. Then you should return (6,4).

3.

Consider the following list:

Index	0	1	2	3	4	5	6	7
Number	23	2	19	3	7	11	5	13

Will the algorithm given on the right be able to find the search value T=2 for the list above?
If yes, **show** the value of L, R, m in each step of the algorithm.
If no, **explain** why not.

```
function binary_search(A, n, T) is
    L := 0
    R := n - 1
    while L ≤ R do
        m := floor((L + R) / 2)
        if A[m] < T then
            L := m + 1
        else if A[m] > T then
            R := m - 1
        else:
            return m
    return unsuccessful
```

Here, ***A*** denotes the list, and ***n*** denotes its size.

4. For different tasks, we often need to search for things. And most of the time, we sort the dataset before performing search operations.
    a. To search an element in an unsorted array, we need O(N) time using linear search. Binary search works in O(logN) time but the array needs to be sorted beforehand which takes at least O(NlogN) time in general. Why would you then sort the array first and then perform a binary search instead of just performing a linear search?
    b. Say, you are working in a system where you need to sort a very big dataset. The memory available to you can barely accommodate the data. You have two options – merge sort & quick sort. Which one should you choose? Why?
    c. Construct an array where quick sort fails to work in O(NlogN) time.
5. Calculate the time complexity of the following recurrence relations.
   [Any method is acceptable as long as steps are shown]

   (a) $T(n) = 2T(n/2) + 1/n$
   (b) $T(n) = 2T(n/3) + n$
   (c) $T(n) = T(n/2) + T(n/5) + n$
   (d) $T(n) = 2T(n/4) + n^2$
6. You have been asked to sort the following array of integers in ascending order.

Index	0	1	2	3	4	5	6	7
Number	23	21	19	15	12	11	5	3

You decided to use Quick Sort using the first element as a pivot for the task. However, your teacher says "Your approach won't be efficient in this case".

(a) Why do you think your teacher says so?
(b) Write the recurrence relation of your approach and calculate the time complexity. You have to show the steps and proper mathematical logic