TIME COMPLEXITY

1. Solve the following questions about time complexity.

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

```
int p = 0;
for (i = n / 2; i > 1; i /= 6) {
    for (j = 2; j <= n; j *= 4) {
        for (k = 0; k <= j; k *= 3) {
            p = p + n / 2;
        }
    }
}
```

b. 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)) {
            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;
        }
    }
}
```

c. Explain why the statement, "The running time of algorithm A is at least $O(n^2)$" is meaningless.

2. 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. $T(n) = 2T(n/4) + n^2$

E. $T(n) = 2T(n/4) + \sqrt{n}$

SORTING AND SEARCHING

3. 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 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.

4. While sorting a list of 10 integers in ascending order using Quick Sort, after the first partition (using the first element as a pivot), the list looks like this: [31, 19, 3, 17, 23, 37, 59, 61, 71, 43].

a. Which element was the pivot before partitioning?
b. Explain your answer in brief.

5. Modify the following binary search algorithm to find the first occurrence of the Key (T) in an ascendingly sorted input list (A).

```
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
```

6. 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 question 5 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.

7. Make an array with the digits of your id. The length of the array should be 8. (For example, if your id is 20101118) then your array should be [2,0,1,0,1,1,1,8]

a. Sort the array in descending order using merge sort. Write the merge function pseudocode and show the simulation.
b. Sort the array in ascending order using heapsort (USE MAX HEAP). Show the simulation only.

Divide and Conquer:

8. You are given an array that may contain both positive and negative integers. You want to find the sum of the contiguous subarray of numbers which has the maximum sum using Divide and Conquer approach.

For example, if the given array is [-1,**5**,**6**,**-2**,**4**,-7,6,-4], then the maximum subarray sum is 13. The subarray is [5,6,-2,4]

a. Show the simulation of finding the maximum sum subarray.
b. Write the pseudocode for finding the cross sum.

9. By using a 2 bit binary multiplication, explain how the Karatsuba algorithm works and bring the time complexity to $O(n^{1.58})$.