CSE221 Section 10 & 11
Fall 2025 [MUNR]
Assignment 1

Instructions:

- The assignment must be handwritten.
- Naming convention:**"StudentID_A1", e.g.: 13101001_A1**
- Submitted scanned PDF in this Google Form: https://forms.gle/UCFjcEF6tzED8PpA7
- Deadline: **11th November, 2025 (11.59PM)**

Please make sure to submit your assignment only through the Google Form. Submissions elsewhere or after the deadline will not be accepted.

Part 1: Iterative Time Complexity

Q1. Analyze the following code snippet and express the time complexity using the Big O-notation. You must mention the time complexity for each loop.

```
int b = 0;
for (int i = n; i > 0; i -= 2) {
    for (int j = n; j > 0; j--) {
        b += i * j;
    }
}

for (int k = 1; k < n; k *= 4) {
    for (int m = 0; m < n; m += 3) {
          if (m == 10) {
              break;
          }
          b += k + m;
    }
   }
   return b;
```

Q2.

(a) Find the time complexity of the following codes.

i)

```
x = 0
for (i = 1; x < n; i = i + 1){
    x = x + i
}
```

ii)

```
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;
        }
    }
}
```

(b) Suppose, there are n students in a school. Every year, the top k students are selected and awarded special prizes.
You are given a list of the students which is not sorted according to their results. **What is the time complexity of selecting the students to be awarded?** Assume that awards are given in a constant time.

(c) Prove (with proper explanation or a formula): $\sqrt[4]{10n^4 + 9n^3 + 8n}$ **= Θ(n).**

Part 2: Recursive Time Complexity

Q1. Calculate the time complexity of the following recurrence relations. [Any method is acceptable as long as steps are shown]

$T(n) = 2T(n/2) + 1/n$
$T(n) = 625T(n/5) + n$
$T(n) = 2T(n/6) + n^3$

Q2. Analyze the code snippet and answer the following questions:

```
int funcB (int n) {
    int m = 0;
    for (int k = 1; k <= n; k += 2) {
        for (int j = n; j >= 1; j--) {
            for (int i = 1; i <= n; i += 2) {
                if (i%2 != 0) {
                    break;
                } else {
                    m = k - j + i;
                }
            }
        }
    }
}
int funcA (int n) {
    if (n == 1) {
        return n;
    }
    else {
        x = funcA(n/2);
        y = funcA(n/4);
        z = funcA(n/6);
        w = funcB(n/2);
        return w;
    }
}
```

i) Express the time complexity of the function **'funcB'** using the Big O-notation.
ii) Write the recurrence relation of the function **'funcA'**. (You can use your answer from question (i) if required.)
iii) Solve the recurrence relation of no (ii) using any suitable method.

Q3. Consider the recurrence relation T(n) = T(√n) + O(n). Can we solve this using the Master Theorem? **Explain** whether the structure of this recurrence fits the requirements of the Master Theorem.

Part 3: Searching & Sorting Algorithms

Q1. Your friend gave you an integer (key) and asked you to find its integer square root without using built-in functions like sqrt() or exponentiation. For example, if the input is 25, the output should be 5 because 5^2=25. If the square root is not an integer, return the floor value of the actual square root. For example, for an input of 10, the output should be 3.

Your friend's initial approach is to use linear search (time complexity O(n)) as shown below.

```
def LinearSearchToFindSquareRoot(key):
     result = -1
     for i in range(1, key, 1):
         if i * i <= key:
             result = i
     return result
```

i) Present an $O(\sqrt{n})$ time algorithm with pseudocode/programmable code/step-by-step logical instructions.
ii) Present an $O(log_2 N)$ time algorithm with pseudocode/programmable code/step-by-step logical instructions.

Q2. You are working with a weather monitoring station that records the temperature every hour for 12 hours during the night shift. During the night, the temperature gradually drops to reach the coldest hour, and then begins to rise again. **Your task is to find the minimum temperature of the night, which corresponds to the coldest hour.**

The station tracks temperature every hour over a 12-hour period. A sample pattern of the temperature reading per hour is given here:

Hour	1	2	3	4	5	6	7	8	9	10	11	12
Temperature (°C)	28	24	20	16	12	10	9	11	15	19	23	27

Propose a modification to binary search to solve the problem. Present the solution with pseudocode/programmable code/step-by-step logical instructions.

Q3. Consider a farm with n cows of two colors, say, white cows and black cows. The cows are arranged in their barn in such a way that all white cows are kept first, followed by all black cows. Each of the cows is assigned a serial number. **You are to find the serial number of the first black cow**.

(a) Describe how to convert the given information into an array of integers.

(b) Given the array of integers from a, **present a method with a time complexity less than O(n) to solve the mentioned problem in the paragraph**. Use pseudocode/ executable code/ step-by-step logical instructions to show your method.

(c) Suppose we implement Quicksort using a strategy where we choose the pivot as the median of the first, last, and middle elements of a subarray. To be specific, the pivot is selected as follows:

$$\text{pivot} = \text{median}(A[\text{st}], A[\text{end}], A[\text{mid}]) \quad \text{where} \quad \text{mid} = \left\lfloor \frac{\text{st} + \text{end}}{2} \right\rfloor$$

For example, consider the array A = [6, 2, 9, 1, 7, 3, 8].
In the initial Quicksort call, start = 0 gives A[0] = 6, end = 6 gives A[6] = 8, and mid index = 0+62 = 3 gives A[3] = 1. Among these three values—1, 6, and 8—the sorted order is (1, 6, 8), so the median, and thus the chosen pivot, is 6.

What is the worst-case input for this pivot selection strategy in Quicksort? Generate such a worst-case scenario with a 7-element array containing the integers 1 through 7, each used exactly once. Briefly explain why it leads to worst-case behavior.

Now consider a factory where n humanoid robots work. You are to sort the robots based on some criteria. Answer Q2(d) – (e) based on this.

(d) Each robot can form connections with some of the other robots. That means each robot can form at most (n-1) connections. Which algorithm will be the fastest to sort the robots based on the number of connections they formed? **Write the name and time complexity of the algorithm.**

(e) Suppose, in the factory, there exists a system that can find the tallest robot in O(1) time. That means you are given a function, find_tallest(A), which returns the tallest robot from a list called A, in constant time.

Considering this information, design an algorithm to sort the robots in descending order of their heights. **The time complexity of your algorithm must not exceed O(n).** Present your algorithm using pseudocode/ executable code/ step-by-step logical instructions.

Part 4: Divide & Conquer

Q1. A football team's analytics department hired you and gave you a task of reviewing match performance data from the past season. Each match's net goal difference (goals scored minus goals conceded) is recorded in a list. Now you were asked by the department to identify the team's best consecutive performance streak, where the net goal difference was maximized over a series of matches.

net_goal_difference: [1, 4, 3, -5 , 5, 6, 1, -4]

To prove your skills, you used an approach where the problem is divided into subproblems to find the solution. Now tell us more about it by answering the following questions.

i) For the given input above, determine the number of times the Cross-Sum exceeds both the Left-Sum and the Right-Sum. Exclude base cases (subarrays of size 1) from consideration. Show your work properly.
Cross-Sum: sum of elements calculated by combining sum from mid to left and mid to right of the current subarray.
Left-Sum: maximum sum in the left half of the subarray.
Right-Sum: maximum sum in the right half of the subarray.

ii) The department wants you to find the maximum consecutive matches with a positive goal difference. Now present your idea to solve the problem considering you have to use the same approach used earlier with pseudocode/programmable code/step-by-step logical instructions.

Q2. Inspired by the Karatsuba algorithm, a curious CSE student Benjamin decided to modify the algorithm in his own way. For a n digit number, instead of using subproblems of size n/2, Benjamin used subproblems of size n/3. He believes with this modification, he can get a faster algorithm than Karatsuba's. So, for finding product of two n digit numbers A and B, **Benjamin splitted A into 3 subproblems (A_1, A_2, A_3) each with n/3 digits and B into 3 subproblems (B_1, B_2, B_3) each with n/3 digits.** Benjamin wants to write a divide and conquer algorithm for finding the product of A and B from these smaller subproblems. (Assume n is a power of 3)

i) Write A in terms of A_1, A_2, A_3 and n. Write B in terms of B_1, B_2, B_3 and n.
ii) Calculate the product AB from your answers in (A).
iii) Help Benjamin write the pseudocode/step-by-step logical instructions of the divide and conquer algorithm.