National University of Computer and Emerging Sciences

Islamabad Campus

# Design and Analysis of Algorithms (CS2009)

**Course Instructor(s):**

Ms. Amina Siddique, Mr Arslan Aslam

# Sessional-I Exam

**Total Time (Hrs):** 1

**Total Marks:** 50

**Total Questions:** 6

**Date:** Sep 23, 2024

_____ _____ _____

**Roll No** **Course Section** **Student Signature**

**Do not write below this line.**

**Attempt all the questions.**

**[CLO 2. Analyze the time and space complexity of different algorithms by using standard asymptotic notations for recursive and non recursive algorithms.]**

**Q1:** Write time complexity of following code in terms of big-Oh (O). **[2 x 5 = 10 marks]**

| | |
|---|---|
| A | ```<br>if(n==1)<br>    return 1;<br>else<br>{<br>    int p=0;<br>    for(int i=1;p<=n;i++)<br>        { p=p+i; }<br>}<br>```<br>Ans: $\sqrt{n}$ |
| B | ```<br>for (int i = n*n; i >= 1; i = i / 5)<br>{<br>    for (int j = 0; j < i; j*=2)<br>    {<br>    cout << j;<br>    }<br>}<br>```<br>Ans: O(∞) |

C

```
int fun(int n)
{
   int sum = 0;
   int outer = 0, inner = 0;
   for (int i = 1; i < n; i=i*2)
   {
      outer++;
      cout << endl<<"Outer : " <<i<<" cycle no. " << outer << endl;
      inner = 0;
      for (int j = 1; j <= i; j *= 2)
      {
         inner++;
         cout<< inner << endl;
         sum++;
      }
   }
   cout << endl << sum << endl;
   return 0;
}
```

Ans: O( log n x log n ) or O (log i x log n)

D

```
int fun(int n)
{
        int count = 0;
        for (int i = 0; i < n; i++)
                for (int j = i; j > 0; j--)
                        count = count + 1;
        return count;
}
```

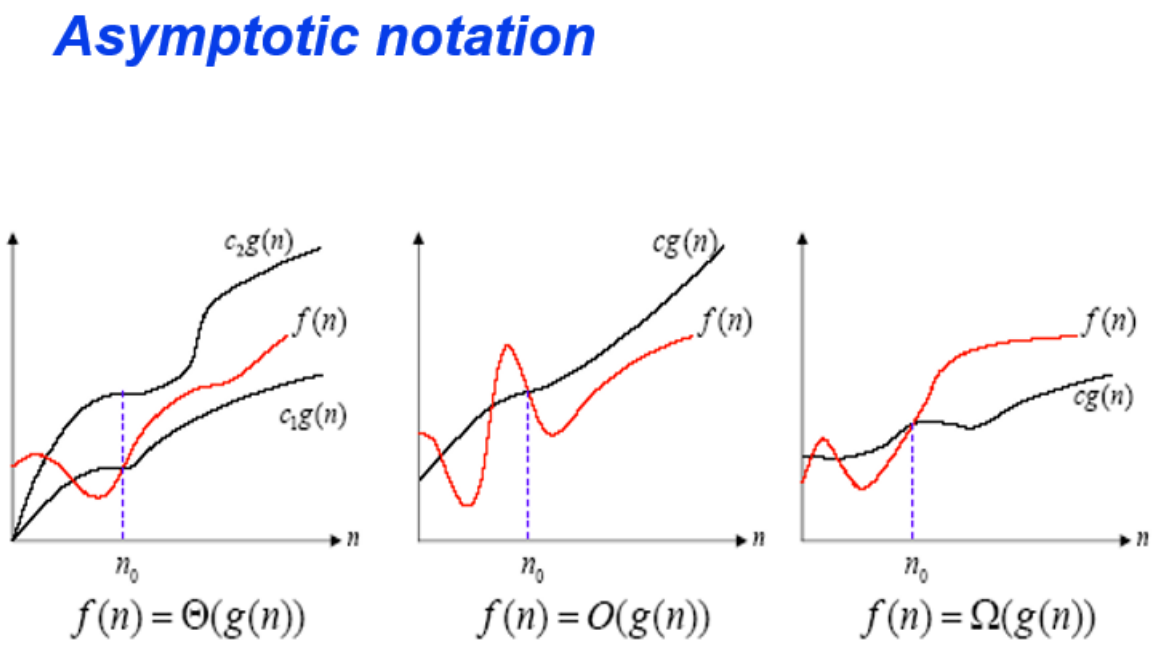
Ans:
O(n(n+1)/2
$O(n^{2)}$

E

```
int fun(int n)
{       int count= 0
        for (int j = 0; i < n*n ; i++)
                count++;
                if( j%2 == 0)
                        break;
        return count;
}
```

Ans: O(1)

**Q2. Explain mathematical function model of Big-Oh, Big Omega and Theta with the help of the graph.** **[3x2 = 6 marks]**




**Equations too.**

**[CLO 2. Analyze the time and space complexity of different algorithms by using standard asymptotic notations for recursive and non recursive algorithms.]**

**Q3: Insertion sort is an iterative sorting algorithm.** **[2+4 = 6 marks]**

```
void insertionSort(int arr[], int n)
{
    int i, key, j;
    for (i = 1; i < n; i++) {
        key = arr[i];
        j = i - 1;

        // Move elements of arr[0..i-1],
        // that are greater than key, to one
        // position ahead of their
        // current position
        while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}
```

a) Can we convert it into recursive function? If yes, mention base and recursive case?
Ans:Yes
Link: https://www.geeksforgeeks.org/recursive-insertion-sort/

b) Write recursive function of Insertion Sort?
Link: https://www.geeksforgeeks.org/recursive-insertion-sort/

**[CLO 2. Analyze the time and space complexity of different algorithms by using standard asymptotic notations for recursive and non recursive algorithms.]**

**Q4:** Write time complexity of following recurrence equation using Master Theorem.**[5 x 2 = 10 marks]**

a) $T(n) = 4T\left(\frac{n}{2}\right) + n^2$

Ans: $O(n^2 \log(n))$

b) $T(n) = 2T\left(\frac{n}{2}\right) + n\sqrt{logn}$

Ans: $O(n(\log^{3/2} n))$

c) $T(n) = 5T\left(\frac{n}{3}\right) + n^{\frac{3}{2}}$

Ans: $O(n^{3/2})$

d) $T(n) = 2T\left(\frac{n}{4}\right) + \frac{n}{\log(n)}$

Ans: O(n)

e) $T(n) = 6T\left(\frac{n}{3}\right) + n^{\frac{5}{4}}$

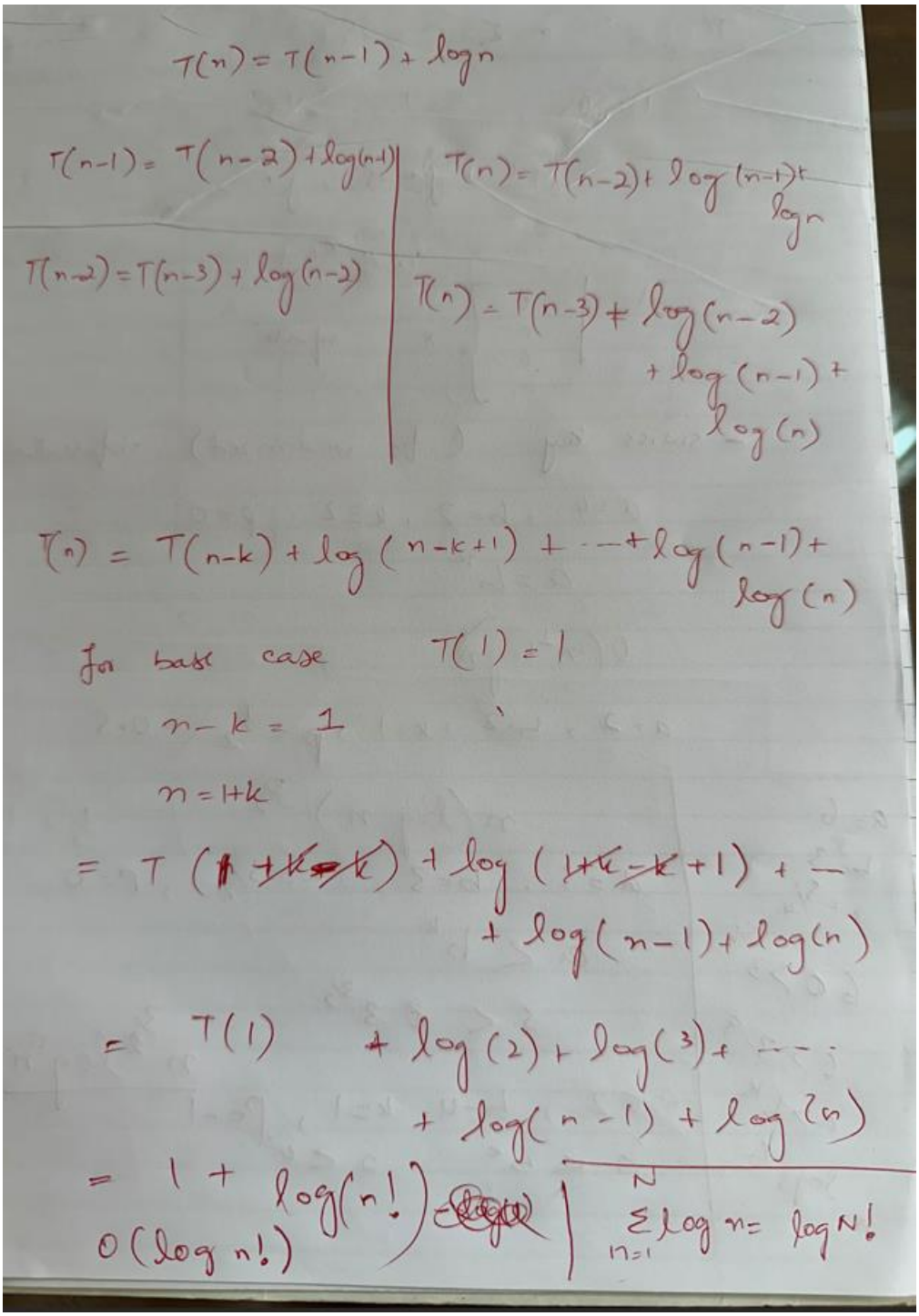
Ans: $O(n^{1.63})$

**[CLO 2. Analyze the time and space complexity of different algorithms by using standard asymptotic notations for recursive and non recursive algorithms.]**

**Q5:** Solve the following using iteration method (substitution method): **[8 marks]**

$$T(n) = T(n-1) + \log(n)\ , n > 0$$

$$T(1) = 1\ , n = 0$$



**[CLO 2. Analyze the time and space complexity of different algorithms by using standard asymptotic notations for recursive and non recursive algorithms.]**

**Q6:** Solve the following recurrence using recurrence tree method: **[10 marks]**

$$T(n) = T\left(\frac{n}{10}\right) + T\left(\frac{9n}{10}\right) + n \quad , \quad n > 1$$
$$T(1) = 1 \quad , \quad n = 1$$