National University of Computer & Emerging Sciences, Karachi
Fall -2025 CS/CY-Department
Assignment- 1
1st September 2025

Course Code: CS2009	**Course Name: Design & Analysis of Algorithms**
Instructor Name:	
Date of Submission	**10th September 2025**

Instructions: **Max Marks: 100**

- Make your Assignments on Full Scale Papers, Typed Assignments will not be acceptable:
- 20% penalty for 1 day late
- 40% penalty for 2 days late Assignment 1
- Submission not allowed afterwards

(10 Marks)

1) Design an algorithm for 2 Dimensional Matrix Addition Compute its Time complexity using frequency count method, also trace the algorithm, for the array of size 3 x 4.

(10 Marks)

2) Explain the process of implementing Linear Search using arrays. Trace the algorithm to search an element 32 from the list of elements: 12, 27, 19, 32, 45.

3) What is the smallest value of n such that an algorithm whose running time is $100n^2$ runs faster than an algorithm whose running time is 2^n on the same machine? (10 Marks)

4) Find the time complexity of the following algorithm. Note: Must write increasing/decreasing pattern of algorithms. (20 Marks)

a)

```
Algorithm Fun(n)
        Sum=0;
        For(i=n2= i>=1 ; i/2)
                Sum=sum+I
                Printf("The Value of Sum is %d", sum)
```

b)

```
Algo fun(n)
        int i, j, k, p, q = 0
        for(i=1; i<n ; i++)
                P=0;
                For(j=n;j>1;j=j/2)
                        ++p;
                For(k=1;k<p;k=k*2)
                        ++q
        return q;
```

c)

```
while(m!=n)
        if(m>n)
                m=m-n
        else
                n=n-m
```

d)

```
algo fun(n)
        int i, j, k=0;
        for(i=n/2;i<=n;i++)
                for(j=2;j<=n; j=j*2)
                        k=k+n/2
        return k;
```

e)

```
k=1;
for(i=0; i<n; i++)
        for(j=0; j<n; j=j+k)
                printf("%d \t", j);
        k=k*2;
```

5) Prove the following (Big O, Omega, and Theta Notations) (50 Marks)

Definitions

Big O Notation: $f(n) \in O(g(n))$ if $\exists$ positive constants c and n_0 such that $f(n) \leq c \cdot g(n)$ for all $n \geq n_0$

Big Omega Notation: $f(n) \in \Omega(g(n))$ if $\exists$ positive constants c and n_0 such that $f(n) \geq c \cdot g(n)$ for all $n \geq n_0$

Big Theta Notation: $f(n) \in \Theta(g(n))$ if $f(n) \in O(g(n))$ AND $f(n) \in \Omega(g(n))$

a) Big O Proofs (Upper Bound) (20 Marks)

1. Prove: $5n^2 - 100n + 50 \in O(n^2)$
2. Prove: $n^2 + n\log n \in O(n^2)$
3. Prove: $n(\log n)^2 + n\log n \in O(n(\log n)^2)$
4. Prove: $n^4 + 50n^3 \notin O(n^3)$

b) Big Omega Proofs (Lower Bound) (15 Marks)

5. Prove: $4n^2 - 1000n + 25 \in \Omega(n^2)$
6. Prove: $n^2 + n\log n \in \Omega(n^2)$
7. Prove: $\log n \notin \Omega(n)$

c) Big Theta Proofs (Tight Bound) (15 Marks)

8. Prove: $10n^2 - 200n + 500 \in \Theta(n^2)$
9. Prove: $n^2 + n\log n \in \Theta(n^2)$
10. Prove: $n \log n + 50 \in \Theta(n \log n)$