



**Comsats University Islamabad**  
**Department of Computer Science**  
Mid Term Examination, Fall 2025

Class/Section: -BSCS – 5A  
Subject: -Design and Analysis of Algorithm  
Instructor: -Tanveer Ahmed  
/

Marks:- 25  
Time:- 90 minutes  
Dated:- October 29, 2025

- Students are required to display their ID cards.
- Attempt all questions. Return this question paper with your answer sheet.
- Use of Mobile Phones and Digital Diaries not allowed

(CLO-1: Demonstrate fundamental concepts of algorithms along with an algorithmic approach to a given problem.)

Question # 01: ----- CLO-1----- Marks = [5]

- 1.1. Demonstrate what the following algorithm computes by performing a dry run of the *mystery* algorithm with  $n = 99$ . What value is returned by the call *mystery* (99)

```
ALGORITHM mystery ( $n$ )  
// Input: A positive integer  $n$   
// Output:-----  
    if  $n > 100$  then  
        return  $n - 10$ ;  
    else  
        return (mystery (mystery ( $n + 11$ )));
```

(CLO-2: Design new algorithms for different computational problems)

Question # 02: ----- CLO-2----- Marks = [5+5]

2.1 Suppose you are given an array  $A[1..n]$  of sorted integers that has been circularly shifted  $k$  positions to the right. For example,  $[35, 42, 5, 15, 27, 29]$  is a sorted array that has been shifted  $k = 2$  positions. We can obviously find the largest element in  $O(n)$  time. Describe an  $O(\log n)$  time algorithm to find the maximum in  $A$ .

2.2 Consider the problem of counting, in a given text, the number of substrings that start with an  $A$  and end with a  $B$ . For example, there are four such substrings in  $CABAAXBYA$ . Design an algorithm for this problem using Brute-Force approach.

Question # 03:

CLO-4

3.1 Find the **time complexity** of the following code. Use most suitable asymptotic notation among  $O$ ,  $\Omega$ , and  $\Theta$  to specify the time efficiency class of the given code.

```
int Function (int n, int a[ ])
{
    int i, m, k=1;
    for(i = 0; i < 5; i++)
    {
        if(a[i] == n)
        {
            m = n;
            while(m)
            {
                if(m%3==2)k++;
                m /= 2;
            } //end while
        } // end if
    } end for
    return k;
} //end function
```

3.2 Consider the following recursive algorithm **Test**

- Derive Recurrence relation of the given algorithm
- Solve recurrence relation that you derive in part (a) using Recursion Tree Method.
- Analyze the worst-case behavior of the following recursive algorithm **Test**. Use appropriate asymptotic notation among  $O$ ,  $\Omega$ , and  $\Theta$  to specify the time efficiency class of the given algorithm.

```
Test(n)
{
    if(n = 0)
        return
    else
    {
        for (i = 1; i <= n; i++)
        {
            // Some O(1) task
        }
        Test(n-1)
        Test(n/2)
    }
}
```