COMSATS UNIVERSITY ISLAMABAD
DEPARTMENT OF COMPUTER SCIENCE
Mid Term Examination, Spring 2025

Class/Section: -BSCS – 5A/B, BCT, BDS-5, -5, BAI-5
Subject: -Design and Analysis of Algorithm
Instructor: -Dr. Malik Ahmed Kamran/Tanveer Ahmed /Ms.Hilal Jan

Marks:- 25
Time:- 90 minutes
Dated:- April 15, 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 *Enigma* algorithm with k=4 and A={5,9,2,6,4,1,3,7}

```
ALGORITHM Enigma(A[n], p, r, k)
1.  if ( p = r) then
2.  return A[p]
3.  else
4.      q ← Test(A, p, r)
5.     rank_x← q-p+1
6.     if( k = rank_x) then
7.       return x
8.     if(k< rank_x) then
9.     return Enigma(A, p, q-1, k)
10. else
11.    return Enigma(A, q+1, r, k-q)
```

```
ALGORITHM Test (A, p, r)
1.  x ← A[p]
2.  i ← p
3.  for s ← p + 1 to r
4.  if (A[s] < x) then
5.      i += 1
6.      swap(A[i], A[s])
7.  swap(A[p], A[i])
8.  return i
```

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

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

2.1 A robot can take steps of 1 meter or 2 meters. Let w_n denote the number of ways the robot can wal n meters. (Order is taken into account so that the 4 meter walk 1, 2, 1 is considered distinct from the meter walk 1, 1, 2). Design a **recursive algorithm** for computing w_n

2.2 Let $D_1, D_2 \ldots, D_n$ be n documents, and let D be a given document. Design a **Brute-Force** algorithm find the closest document D_i (where $1 \leq i \leq n$) to D using the Hamming distance as the similarity measur The Hamming distance between two strings of equal length is defined as the number of positions at whic their corresponding symbols are different.

CLO-4 Analyze best, average, and worst-case behaviors of an algorithm.

Question # 03: ---------- **CLO-4** ---------- **Marks = [10]**

3.1 Consider the following Algorithm

```
BinaryInsertionSort (int a[], int n)
{
  int ins, i, j;
  int tmp;

  for (i = 1; i < n; i++) {
    ins = BinarySearch (a, 0, i, a[i]);
    if (ins < i) {
      tmp = a[i];
      for (j = i - 1; j >= ins; j--)
        a[j + 1] = a[j];
      a[ins] = tmp;
    }
  }
}
```

```
BinarySearch (int a[], int low, int high, int key)
{
  int mid;

  if (low == high)
    return low;

  mid = low + ((high - low) / 2);

  if (key > a[mid])
    return BinarySearch (a, mid + 1, high, key);
  else if (key < a[mid])
    return BinarySearch (a, low, mid, key);

  return mid;
}
```

a) What is the measure of input's size in the given Binary Insertion Sort algorithm? **Marks = [0.5]**

b) What is the basic operation(s) (primitive operation(s)) in this algorithm? **Marks = [0.5]**

c) Can the number of times the basic operation is executed differ for inputs of the same size? Explain why or why not. **Marks = [0.5]**

d) Set up a summation formula to count how many times the basic operation is executed throughout the sorting process. Use the most suitable asymptotic notation among **O (Big-O), Ω (Omega), and Θ (Theta)** to classify the algorithm's efficiency for both the worst-case and best-case scenarios. **Marks = [02]**

e) Analyze how the runtime of this algorithm is affected when the input size is doubled in both the worst-case and best-case scenarios. **Marks = [1.5]**

3.2 Consider the following algorithm **Marks = [2+3]**

```
ALGORITHM Sort (A, f, l)
  1. if(f < l) then
  2. m ← n/3
  3. Sort(A, f, m)
  4. Sort (A, m+1, l)
  5. Combine (A, f, m, l)
```

a) Set up a **recurrence relation** for the given algorithm Sort( ). The **Combine** function merges the two sorted parts. Assuming it takes **O(n)** time.

b) Solve the recurrence relation that you derive in part (a) using **Recursion Tree** Method