AHMEDABAD UNIVERSITY
School of Engineering and Applied Science
2025-2026 Winter

CSE301 Design and Analysis of Algorithms
Mid-Term Examination
Monday February 23 2026 12:00-13:00

Name:	Roll no.	A	U	2						Marks /60

PART-A

INSTRUCTIONS

1. *Write your name [2.5 points] and ID [2.5 points] in the beginning of question paper provided to you.*
2. *You are allowed three A4 sized cheat sheets of handwritten or typed notes. During the exam you are not allowed to exchange or share your cheat sheets. No calculators or electronic gadgets allowed.*
3. *This PART A has 5 short answer questions. You are required to write your choice along with the justification in space provided in the question paper. You are allowed to use rough sheets for scratch work.*

1. Given a function that takes as input a base b and an exponent n *[6 points]*

```
function power(int b, int n)
begin
        result = 1
        if n == 0 return 1 // base case b^0 = 1
        if n == 1 return b // base case b^1 = b
        if n > 1
                result = power(b, floor(n/2)) * power(b, ceiling(n/2)) // recursively call power for
                                                                       // floor(n/2) and ceiling(n/2)
        else
                return(-1)
end
```

a. [3 points] Write down the recurrence corresponding to this program as a function of n.

b. [3 points] Solve the recurrence equation as a function of n.

2. You are given the following modification of the merge sort algorithm for sorting n integers. [6 points]

Merge sort recursively calls itself until the size of the list is greater than some integer k. The moment the size of a list is less than or equal to k it directly uses insertion sort to sort the list of size at most k. (i.e., instead of recursively solving the entire problem, the recursion stops moment the size of the list is less than or equal to k)

What is the run-time complexity of this modified merge sort as a function of n and k?

3. Given a binary search tree (BST) on n nodes, where each node has the following structure [10 points]

```
Struct Node
{
    Int     key // key value stored at the node
    Int     size_of_right_sub_tree // the number of nodes in its right sub-tree
}
```

a. [7.5 points] Write a recursive procedure "int find_no_items_greater_than(Node *ptr, int key)" that given a pointer to the root of the BST and a key value, returns the number of items in that BST with key values greater than "key". You can assume all items in the BST are distinct.

b. [2.5 points] What is its run-time complexity as a function of n?

4. Given the following functions of n, please answer the following questions: [18 points]

i. $log(n!)$
ii. n^2
iii. $1/n$
iv. $log(log(n))/log(n)$
v. 2^{100000}
vi. $n^{2/logn}$
vii. $n/log(n^2)$
viii. $log(log(n^2))$
ix. n^{logn}
x. $2^{log(n^{1/2})}$

a. [3 points] Which function grows faster than n^3?

b. [3.5 points] What are the constant functions in this list?

c. [2.5 points] What is the slowest growing function in the above list?

d. [2.5 points] Which is the fastest growing function in this list?

e. [2.5 points] Which function has the same growth rate as *nlogn*?

f. [4 points] What are the functions that grows slower than *n* but grows faster than *log(n)* (functions that are slowly growing when compared to n but faster than log(n))

5. Given a min-heap consisting of $n = 2^h - 1$ nodes, for some integer $h > 1$, please answer the following questions: [15 points]

a. [2 points] How many leaf nodes are there?

b. [3 points] How many leaf nodes you need to remove to make the number of leaf nodes less than or equal to the number of non-leaf nodes?

c. [4 points] If you need to find for some constant k, the worst-case run-time to find the k^{th} smallest item in the min-heap?

d. [6 points] If you must convert this min-heap into a max heap, how will you do this and what will be the worst-case run-time complexity as a function of n?