**VISHWAKARMA GOVERNMENT ENGINEERING COLLEGE, CHANDKHEDA-382424**

**Information Technology Department**

|  |  |
| --- | --- |
| **Subject Name:** | 3150703-ANALYSIS AND DESIGN OF ALGORITHMS |
| **Branch & Semester:** | 5th - IT |
| **Enrollment No.** |  |
| **Name:** |  |
| **Academic Year:** | 2022-23 |

**Practical Index**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Aim of Practical** | **Date** | **Rubric wise Marks** | | | | **Sign** |
|  |  |  | **C1** | **C2** | **C3** | **Total** |  |
| **1** | Implement a function for each of following problems and count the number of steps executed/Time taken by each function on various inputs and write complexity of each function. Also draw a comparative chart. In each of the following function N will be passed by user.  1. To calculate sum of 1 to N numbers using loop.  2. To calculate sum of 1 to N numbers using equation.  3. To calculate sum of 1 to N numbers using recursion. |  |  |  |  |  |  |
| **2** | Write user defined functions for the following sorting methods and compare their performance by time measurement with random data and Sorted data.  1.      Selection Sort  2.      Bubble Sort  3.      Insertion Sort  4.      Merge Sort  5.      Quick Sort |  |  |  |  |  |  |
| **3** | Implement a function of sequential search and count the steps executed by function on various inputs for best case and worst case. Also write complexity in each case and draw a comparative chart. |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **4** | Compare the performance of linear search and binary search on various inputs. |  |  |  |  |  |  |
| **5** | Implement functions to find factorial of any number using iteration and recursive method. Compare the performance of two methods by counting number of steps executed/ Time taken by each function on various inputs. Also draw a comparative chart. |  |  |  |  |  |  |
| **6** | Implement a program for randomized version of quick sort and compare its performance with normal version of quick sort using steps count on random and sorted data. |  |  |  |  |  |  |
| **7** | Implement program to solve problem of making a change using dynamic programming. |  |  |  |  |  |  |
| **8** | Implement program of chain matrix multiplication using dynamic programming. |  |  |  |  |  |  |
| **9** | Implement program to solve LCS problem using dynamic programing. |  |  |  |  |  |  |
| **10** | Implement program to solve Knapsack problem using dynamic programming. |  |  |  |  |  |  |
| **11** | Implement Program for fractional/binary Knapsack using greedy design technique. |  |  |  |  |  |  |
| **12** | Implement program for Making Change using greedy design technique. |  |  |  |  |  |  |
| **13** | Implement program for Kruskal's algorithm. |  |  |  |  |  |  |
| **14** | Implement Rabin-Karp string matching algorithm. |  |  |  |  |  |  |

Rubrics For Evaluation :

C1 : Understanding of Problem

C2 : Implementation of Problem

C3 : Documentation and Regularity