Chandubhai S. Patel Institute of Technology-Changa
U & P U. Patel Department of Computer Engineering

Subject Name: Design and Analysis of Algorithms Semester: 5th
Subject Code: CE342 Academic year: June -November 2019

Practical List

Analysis of Program should contain following sub heading(s).

1. Impact of Input Size on the Performance of Program. Make Table and Draw graph of Input Size Vs Running Time/Total No of Instructions. Take at least Five Input of Different Size.
2. Impact of Input Quality on the Performance of Program. Make Table and Draw graph of Best Case, Worst Case and Average Case Input Quality Vs Running Time/ Total No. of Instructions.
3. Rate of Growth of Program. Make Table and Draw Graph of Input Size Vs Instruction(s) Running Maximum No of Time in the Program.
4. Conclusion from the above graph or Data Table
5. For all Test cases, add column for output, calculate the answer and write the answer in the output column and verify with the output of the program.

<table>
<tr><th>Exp. No.</th><th colspan="2">Name of Experiment</th><th>Hours</th></tr>
<tr><td>1.</td><td colspan="2">Implement and analyze algorithms given below.</td><td>04</td></tr>
<tr><td></td><td>1.1</td><td>Factorial (Iterative and Recursive)</td><td></td></tr>
<tr><td></td><td>1.2</td><td>Euclidean algorithm</td><td></td></tr>
<tr><td></td><td>1.3</td><td>Matrix Addition and Matrix Multiplication(Iterative)</td><td></td></tr>
<tr><td></td><td>1.4</td><td>Recursive Linear Search and Binary Search</td><td></td></tr>
<tr><td></td><td>1.5</td><td>Find a subset of a given set S = {s1,s2,.....,sn} of n positive integers whose sum is equal to a given positive integer d. For example, if S= {1, 2, 5, 6, 8} and d = 9 there are two solutions {1,2,6} and {1,8}.A suitable message is to be displayed if the given problem instance doesn't have a solution.</td><td></td></tr>
<tr><td colspan="4"></td></tr>
<tr><td>2.</td><td colspan="2">Implement and analyze algorithms given below (Compare them)</td><td>02</td></tr>
<tr><td></td><td>2.1</td><td>Bubble Sort</td><td></td></tr>
<tr><td></td><td>2.2</td><td>Selection Sort</td><td></td></tr>
<tr><td></td><td>2.3</td><td>Insertion Sort</td><td></td></tr>
<tr><td colspan="4"></td></tr>
<tr><td>3.</td><td colspan="2">Divide and Conquer Strategy</td><td>04</td></tr>
<tr><td></td><td>3.1</td><td>Implement and perform analysis of worst case of Merge Sort and Quick sort. Compare both algorithms.</td><td></td></tr>
<tr><td></td><td>3.2</td><td>Implement the program to find X^Y using divide and conquer strategy and print the total number of multiplications required to find X^Y. Test the program for following test cases:
<table>
<tr><th>Test Case</th><th>X</th><th>Y</th></tr>
<tr><td>1</td><td>2</td><td>6</td></tr>
<tr><td>2</td><td>7</td><td>25</td></tr>
<tr><td>3</td><td>5</td><td>34</td></tr>
</table></td><td></td></tr>
<tr><td colspan="4"></td></tr>
</table>

4.	**Greedy Approach**		**04**
	4.1	A cashier at any mall needs to give change of an amount to customers many times in a day. Cashier has multiple number of coins available with different denominations which is described by a set C. Implement the program for a cashier to find the minimum number of coins required to find a change of a particular amount A. Output should be the total number of coins required of given denominations. Check the program for following test cases: **Test Case** | **Coin denominations C** | **Amount A** 1 | ₹1, ₹2, ₹3 | ₹ 5 2 | ₹18, ₹17, ₹5, ₹1 | ₹ 22 3 | ₹100, ₹25, ₹10, ₹5, ₹1 | ₹ 289 Is the output of Test case 2 is optimal? Write your observation.	
	4.2	Let S be a collection of objects with profit-weight values. Implement the fractional knapsack problem for S assuming we have a sack that can hold objects with total weight W. Check the program for following test cases: **Test Case** | **S** | **profit-weight values** | **W** 1 | {A,B,C} | Profit:(1,2,5) Weight: (2,3,4) | 5 2 | {A,B,C,D,E,F,G} | Profit:(10,5,15,7,6,18,3) Weight: (2,3,5,7,1,4,1) | 15 3 | {A,B,C,D,E,F,G} | A:(12,4),B:(10,6), C:(8,5),D:(11,7), E:(14,3),F:(7,1), G:(9,6) | 18	
	4.3	Suppose you want to schedule N activities in a Seminar Hall. Start time and Finish time of activities are given by pair of (si,fi) for ith activity. Implement the program to maximize the utilization of Seminar Hall. (Maximum activities should be selected.) **Test Case** | **Number of activities (N)** | **(si,fi)** 1 | 9 | (1,2), 1,3),(1,4),(2,5),(3,7), (4,9), (5,6), (6,8), (7,9) 2 | 11 | (1,4),(3,5),(0,6),(3,8),(5,7), (5,9), (6,10), (8,12),(8,11) (12,14), (2,13)	
5.	**Dynamic Programming**		**06**
	5.1	Implement a program which has BNMCOEF() function that takes two parameters n and k and returns the value of Binomial Coefficient C(n, k). Compare the dynamic programming implementation with recursive	

implementation of BNMCOEF(). (In output, entire table should be displayed.)

Test Case	n	k
1	5	2
2	11	6
3	12	5

5.2 Implement the program 4.2 using Dynamic Programing. Compare Greedy and Dynamic approach.

5.3 Given a chain < A1, A2,…,An> of n matrices, where for i=1,2,…,n matrix Ai with dimensions. Implement the program to fully parenthesize the product A1,A2,…,An in a way that minimizes the number of scalar multiplications. Also calculate the number of scalar multiplications for all possible combinations of matrices.

Test Case	n	Matrices with dimensions
1	3	A1: 3*5, A2: 5*6, A3: 6*4
2	6	A1: 30*35, A2: 35*15, A3: 15*5, A4: 5*10, A5: 10*20, A6: 20*25

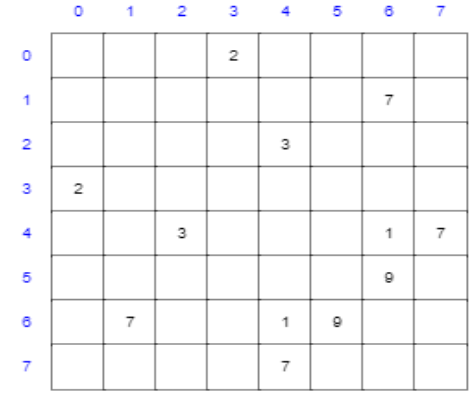
5.4 Implement a program to print the longest common subsequence for the following strings:

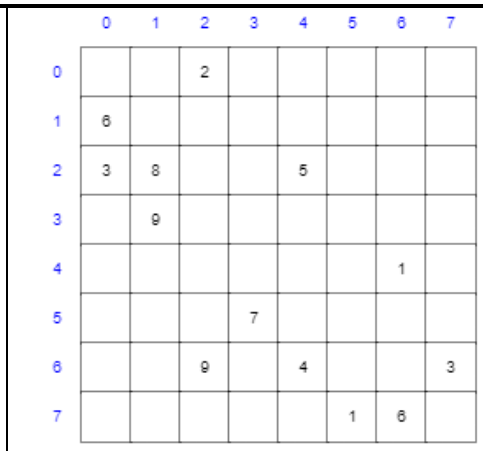
Test Case	String1	String2
1	ABCDAB	BDCABA
2	EXPONENTIAL	POLYNOMIAL
3	LOGARITHM	ALGORITHM

6. Graph **06**

6.1 Write a program to detect cycles in an directed graph.

6.2 From a given vertex in a weighted graph, implement a program to find shortest paths to other vertices using Dijkstra's algorithm.

Test Case	Adjacency Matrix of graph	Start Vertex
1		1

		2		3
	6.3	Find Minimum Cost spanning tree of a given undirected graph using Prim's algorithm.		
7.	**Backtracking**			**02**
	7.1	Implement a program to print all permutations of a given string.		

Test Case	String
1	ACT
2	NOTE

8.	**String Matching Algorithm**			**02**
	8.1	Suppose you are given a source string S[0 ..n − 1] of length n, consisting of symbols a and b. Suppose that you are given a pattern string P[0 ..m − 1] of length m < n, consisting of symbols a, b, and *, representing a pattern to be found in string S. The symbol * is a "wild card" symbol, which matches a single symbol, either a or b. The other symbols must match exactly. The problem is to output a sorted list M of valid "match positions", which are positions j in S such that pattern P matches the substring S [j..j + |P|− 1]. For example, if S = ababbab and P = ab*, then the output M should be [0, 2]. Implement a straightforward, naive algorithm to solve the problem.		
	8.2	Implement Rabin karp algorithm and test it on the following test cases:		

Test Case	String	Pattern
1	2359023141526739921	31415 q=13
2	ABAAABCDBBABCDDEBCABC	ABC q=101
