**Department of CSE, BMSCE**

**ADA Lab program list for Test-1**

**Aug-Dec 2019**

**1.** Using Quick Sort, Given an array of n numbers, the task is to answer the following queries:

Kth Smallest(start, end, k) : Find the Kth smallest number in the range from array index 'start' to 'end'.

Examples:

Input : arr[] = {3, 2, 5, 1, 8, 9|

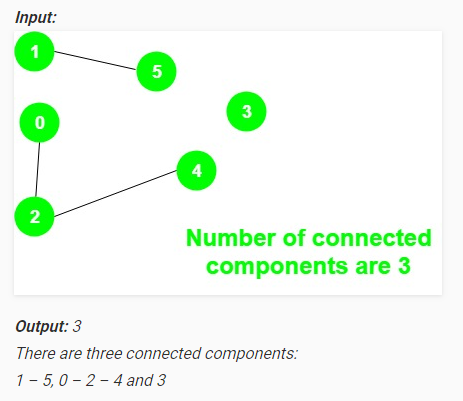
Query 1: start = 2, end = 5, k = 2

Query 2: start = 1, end = 6, k = 4

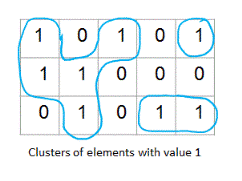
Output : 2

5

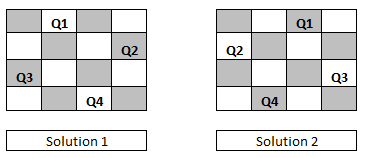
**2.** Using DFS/BFS, given an undirected graph, print all connected components line by line. For example consider the following graph.



**3.** Using DFS/BFS, Given a 2D matrix of 0s and 1s, find total number of clusters or islands formed by elements with value 1.  For example, in the below shown 2D matrix there are total three such clusters.



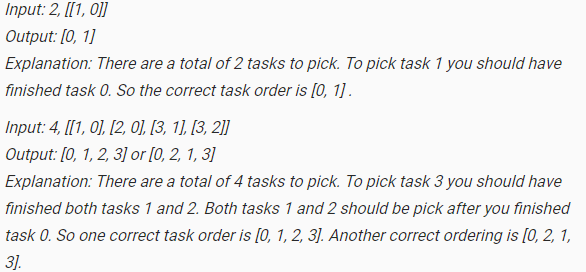
**4.** Using Backtracking, Solve N-Queens problem. The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. For example, following are the solutions for 4 Queen problem.



**5.** Using Topological Sorting, find the ordering of tasks from given dependencies. There are a total of n tasks you have to pick, labelled from 0 to n-1. Some tasks may have prerequisites tasks, for example to pick task 0 you have to first finish tasks 1, which is expressed as a pair: [0, 1]

Given the total number of tasks and a list of prerequisite pairs, return the ordering of tasks you should pick to finish all tasks.

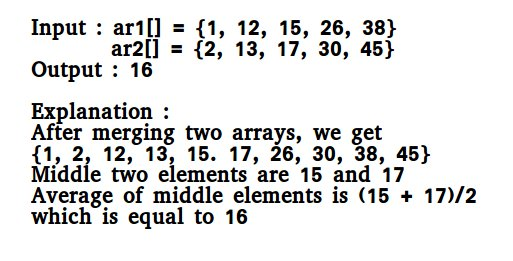
There may be multiple correct orders, you just need to return one of them. If it is impossible to finish all tasks, return an empty array.



**6.** Using Backtracking, solve Sum of Subsets problem. Subset sum problem is to find subset of elements that are selected from a given set whose sum adds up to a given number K. We are considering the set that contains non-negative values. It is assumed that the input set is unique (no duplicates are presented) and it is given in increasing order.

Example: n=3, S=6 and W1=2, W2=4, W3=6 ; Solutions : {2, 4} and {6}

**7.** Using Divide and Conquer technique, find the median of two sorted arrays of same size. The complexity of the program should be O(log(n)).



**8.** Using DFS/BFS, finding shortest distance between two cells in a matrix or grid. Given a matrix of N\*M order. Find the shortest distance from a source cell to a destination cell, traversing through limited cells only. Also you can move only up, down, left and right. If found output the distance else

-1. This problem is meant for single source and destination.

|  |  |
| --- | --- |
| Input : {'0', '\*', '0', 's'},  {'\*', '0', '\*', '\*'},  {'0', '\*', '\*', '\*'},  {'d', '\*', '\*', '\*'} | Input : {'0', '\*', '0', 's'},  {'\*', '0', '\*', '\*'},  {'0', '\*', '\*', '\*'},  {'d', '0', '0', '0'} |
| Output : 6 | Output : -1 |

The path can only be created out of a cell if its value is \* and at any given moment, we can only move one step in one of the four directions. The valid moves are:

Go Up: (x,y) -> (x-1, y) Go Left: (x,y) -> (x, y-1)

Go Down: (x,y) -> (x+1, y) Go Right: (x,y) -> (x, y+1)

**9.** Using divide and conquer technique

**a.** Search an element in a sorted and rotated array. Program running time should be O(log n)

Input : arr[] = {5, 6, 7, 8, 9, 10, 1, 2, 3};

key = 3

Output : Found at index 8

Input : arr[] = {5, 6, 7, 8, 9, 10, 1, 2, 3};

key = 30

Output : Not found

Input : arr[] = {30, 40, 50, 10, 20}

key = 10

Output : Found at index 3

**b.** Given a sorted array with repeating integers. You need to find the first occurrence , last occurrence and count of a given key in the array. Use Only Binary Search. Your program should run in O(log N) time. If element is not present print -1 -1 0.

Input Format

First line contains N , then N integers in next line , then key in the next line.

Output Format

3 space separated integers for lower bound , upper bound and count/frequency of that key.

Sample Input

6

1 2 2 2 3 4

2

Sample Output

1 3 3

Explanation

First occurrence of 2 is at 1. Second Occurrence of 2 is at 3. Count is 3.

**10.** Using backtracking, Solve Rat in Maze problem. A Maze is given as N\*N binary matrix of blocks where source block is the upper left most block i.e., maze[0][0] and destination block is lower rightmost block i.e., maze[N-1][N-1]. A rat starts from source and has to reach the destination. The rat can move only in two directions: forward and down.

In the maze matrix, 0 means the block is a dead end and 1 means the block can be used in the path from source to destination.

Following is binary matrix representation of the above maze.

{1, 0, 0, 0}

{1, 1, 0, 1}

{0, 1, 0, 0}

{1, 1, 1, 1}

Following is a maze with highlighted solution path.



Following is the solution matrix (output of program) for the above input matrix.

{1, 0, 0, 0}

{1, 1, 0, 0}

{0, 1, 0, 0}

{0, 1, 1, 1}

All entries in solution path are marked as 1.