CSE3004 21BCE9905

Design and Anaysis of Algorithms

1. Implementation of binary Search using divide and conquer Methodlogy.

Ans:-

```
import java.util.Scanner;
Public class Main {
 Public Static void main(String [] args){
  Scanner sc = new Scanner(system.in);
  int nums[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
 System.out.println("Enter the target Element ");
  int target = sc.nextInt();
  int result = binarySearch(nums, target);
  if ( result == -1){
   System.out.println("Element not found");
  } else {
  System.out.println("Index of the target element is "+result);}
 }
 Public Static int binarySearch(int nums[], int target){

   int Start = 0;
   int end = nums.length-1;
   int Steps = 0;
  while (Start <= end){
   Steps ++;
```

```
    int mid = (Start + end)/2;
    if (nums[mid] == target) {
     System.out.println("Number of Steps taken to search is"+
                                               Steps);
    } else if (nums[mid] > target){
       end = mid -1; } else {
        Start = mid+1;
    }
   System.out.println("Number of Steps taken to Search is" + Steps);
   return -1;
 }
}
```

2. Write a program to find optimal ordering of matrix multiplication (Note: Use Dynamic Programming method)

Ans:-

```
 import java.util.*;
 import java.io.*;
Public Class MatrixChainMultiplication {
 Public Static void main(String[] args){
   int arr[] = new int[] {1, 2, 3, 4};
   int size = arr.length;
  System.out.println("Minimum number of multiplications is "+
               MatrixChainOrder(arr, size));
 }
 Static int MatrixChainOrder(int p[], int n) {
    int m[][] = new int[n][n];
```

```
        int i,j,k,L,q;
    for(i=1;  i<n;  i++)
           m[i][i] =0;

    for(L=2; L<n;  L++)
    {
       for(i=1;  i<n-L+1;  i++)
       {
          j= i+L-1;
          if (j== n)
             Containue;
          m[i][j] = Integer.MAX_VALUE;

          for(k=i;  k<=j-1;  k++)
          {
             q = m [i][k] + m[k+1][j] +
                p[i-1] x p[k]* p[j];
                if(q<m[i][j]){
                   m[i][j]=q;
                }
          }
       }
    }
    return m[1][n-1];
  }
```

output :-

Minimum number of multilications is 18

3. Write a program that implements backtracking algorithm to Solve the problem i.e. Place eight non-attacking Queens on the board.

```
import Java.Util.Arrays;                    // 21BCE9905

class Helloworld{
   Static final int N=8;
  Public static void main(String[] args){
    int[][] board = new int[N][N];
    if(!SolveQueens(board,0)){
      System.out.println("NosuchSolution found")
    }
  Static boolean isSafe(int[][] Board, Int row, int col){

   for (int x=0; x<col; x++)
       if (board[row][x]==1)
           return false;

   for (int x=row; y=col; x>=0&& y>=0; x--,y--)

   if (board[x][y]==1)
        return false;
   for (int x = row; y=col; x<N&& y>=0; x++, y--)
      if (board[x][y]==1)
         return fans;
   return true;
  }

  Static boolean SolveNQueens(int[][] board, int col){
    if (col==N){
      for (int[] row: board)
         System.out.println(Arrays.toString(row));
     System.out.println();
     return true;
    }
  r
```

```
        for (int i=0; i<N; i++) {
            if (isSafe(board, i, col)) {
                board[i][col] = 1;
                if (solveNQueens (board, col+1))
                    return true;
                board[i][col] = 0;
            }
        }
        return false;
    }
}
```

Output:-

```
(1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0 0 0, 0]
[0, 1, 0, 0, 0, 0 0 0]
[0, 0, 0, 1, 0, 0, 0 0]
[0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0]
```