# Design and Analysis of Algorithms

ASSIGNMENT-21


**Group-2** **Section-C**

IIB2019004 Saloni Singla

IIB2019005 Sandeep Kumar

IIB2019006 Amanjeet Kumar

# Problem:

The longest Zig-Zag subsequence problem is to find length of the longest subsequence of given sequence such that all elements of this are alternating.

# 01
# Approach

Basic

# ALGORITHM-1

- During recursion, we will determine the state of the problem if,
  State is 0 then we want an element smaller than it as subsequence is already been started
  State is 1 then t we want an element greater than it as subsequence is already been started
  State is 2 then we can take any element as subsequence is yet to be started.

  Then in the Base case, when we have no further elements left. (n==0).

- After knowing the state, either discard the number or include the number in the current sequence if the previous element is smaller with state = 1 or previous element is greater with state = 0 and if state =2 we take all of the possibilities.

# Pseudo Code :

```
HELPER(arr,n,prev,state)
{
    if n = 0 then
       return 0
       if state < 0 then
    {
        a = b = INT_MIN
        a = HELPER(arr,n - 1, prev, state)
        if arr[n - 1] > prev then
            b = 1 + HELPER(arr,n - 1, arr[n - 1], 1)
        return max(a, b)
    }
```

```
    else if state>0
    {
        a = b = INT_MIN
        a = HELPER(arr,n - 1, prev, state)
        if arr[n - 1] < prev then
            b = 1 + solve(arr,n - 1, arr[n - 1], -1)
        return max(a, b)
    }
    else
    {
        a = b = c = INT_MIN
        a = HELPER(arr,n - 1, prev, state)
        b = 1 + solve(arr,n - 1, arr[n - 1], -1)
        c = 1 + solve(arr,n - 1, arr[n - 1], 1)
        return max(a, max(b, c))
    }
}
```

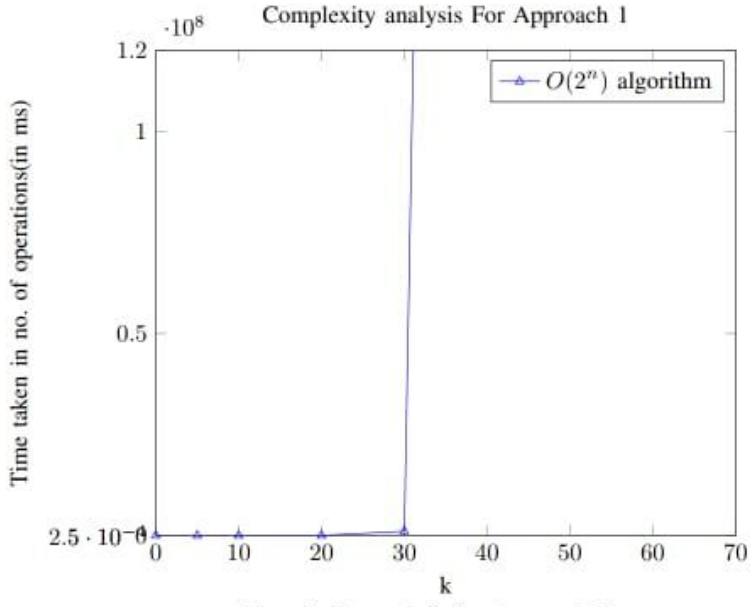
# Time Complexity and Space Complexity Analysis

**Time Complexity Analysis**

The time complexity will beO(2^n)because it checks out allthe possibilities by recursing.

**Space Complexity Analysis**

The space complexity isO(n),for storing the input array.



# 02

# Approach

Using DP

# ALGORITHM-2

- At start, we get to know about the state of the problem whether it is 0, 1 or 2.
- To store the values of each element, there is a two dimensional matrix of size n x 3 (dp[n][3]).
- 2D matrix will store each element’s state in best possible way.
- If the value of dp[i][state] is precalculated then directly pass on it for further calculation, else calculate dp[i][state] and store it for further use.
- Finally we return the lenght of the subsequence.

# Pseudo Code :

```
int HELPER(n,prev,state,dp[][],arr)
{
    if n=0 then
       return 0
    if dp[n][state] !=-1 then
       return dp[n][state]

    if state = 0
    {
        a=b=INT_MIN
        a = HELPER(n-1,prev,state,dp,arr)
        if arr[n-1]>prev then
            b = 1+ HELPER(n-1, arr[n-1] , 1,dp,arr)

        return dp[n][state] = max(a,b)
    }
```

```
    else if state = 1 then
    {
        a=b=INT_MIN
        a = HELPER(n-1, prev , state,dp,arr)
        if arr[n-1]<prev then
            b = 1+ HELPER(n-1, arr[n-1] , 0,dp,arr)

        return dp[n][state] = max(a,b)
    }
    else
    {
        a=b=c= INT_MIN;
        a = HELPER(n-1, prev , state,dp,arr)
        b = 1+HELPER(n-1, arr[n-1] , 0,dp,arr)
        c = 1+HELPER(n-1, arr[n-1] , 1,dp,arr)
        return dp[n][state] = max(a,max(b,c))
    }
}
```

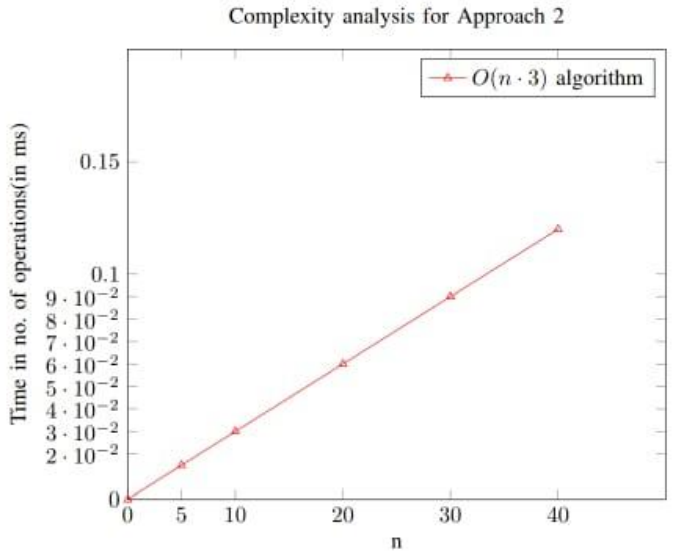
# Time Complexity and Space Complexity Analysis

**Time Complexity Analysis**

The time complexity will beO(n∗3)because we are in away doing memoization our approach 1 code and saving time by using values of the precalculated subproblems.

**Space Complexity Analysis**

The space complexity will beO(n)for input array andO(n∗3)for storing the values of each condition by dynamic programming.



# CONCLUSION

Above two methods have different time and space complexities and meet to fulfill the problem statement. The order in which they are good can be listed as:

I. Approach 2

II. Approach 1

Based on the time complexities and space complexities.

# REFERENCES

Utkarsh Trivedi, 'Longest Zig-Zag Subsequence',GeeksforGeeks, 2018. [Online]. [Accessed: 27-Mar-2021]