

Design and Analysis of Algorithms Assignment - 6

Name: Dhanraj Kore

Div: TY B

Roll No: 60

Batch : B-3

Longest Increasing Sub-sequence

Approach 1: Using Recursion

CODE :

```
#include<stdio.h>
#include<stdlib.h>
int _lis( int arr[], int n, int *max_ref)
{
    if (n == 1)
        return 1;

    int res, max_ending_here = 1;

    for (int i = 1; i < n; i++)
    {
        res = _lis(arr, i, max_ref);

        if (arr[i-1] < arr[n-1] && res + 1 > max_ending_here)
            max_ending_here = res + 1;
    }

    if (*max_ref < max_ending_here)
        *max_ref = max_ending_here;

    return max_ending_here;
}
```

```

int lis(int arr[], int n)
{
    int max = 1;
    _lis( arr, n, &max );
    return max;
}

int main()
{
    int arr[] = { 10, 22, 9, 33, 21, 50, 41, 60 };

    int n = sizeof(arr)/sizeof(arr[0]);

    printf("\n\tLength of Longest Increasing Subsequence is %d\n\n", lis( arr, n ));

    return 0;
}

```

O/P:

The screenshot shows the Visual Studio Code editor with a C++ file named `lis_1.cpp`. The code implements a recursive function `_lis` to find the Longest Increasing Subsequence (LIS) of an array. The `main` function initializes an array `arr = { 10, 22, 9, 33, 21, 50, 41, 60 }` and calls `lis(arr, n)` to compute the result. The terminal output shows the execution of the program, which prints the length of the LIS as 5.

```

lis_1.cpp
8  for (int i = 1; i < n; i++)
9  {
10     res = _lis(arr, i, max_ref);
11     if (arr[i-1] < arr[n-1] && res + 1 > max_ending_here)
12         max_ending_here = res + 1;
13 }
14 if (*max_ref < max_ending_here)
15     *max_ref = max_ending_here;
16 return max_ending_here;
17 }
18
19 int lis(int arr[], int n)
20 {
21     int max = 1;
22     _lis( arr, n, &max );
23     return max;
24 }
25
26 int main()
27 {
28     int arr[] = { 10, 22, 9, 33, 21, 50, 41, 60 };
29     int n = sizeof(arr)/sizeof(arr[0]);
30     printf("\n\tLength of Longest Increasing Subsequence is %d\n\n", lis( arr, n ));
31     return 0;
32 }

```

```

PS C:\Users\ASUS\Documents\TY SEM 2\DAAD\LAB\LAB 6> g++ lis_1.cpp
PS C:\Users\ASUS\Documents\TY SEM 2\DAAD\LAB\LAB 6> .\a.exe

Length of Longest Increasing Subsequence is 5

PS C:\Users\ASUS\Documents\TY SEM 2\DAAD\LAB\LAB 6>

```

Time Complexity: Exponential

Space Complexity: $O(1)$

Approach 2: Using Dynamic Programming

CODE :

```
#include<bits/stdc++.h>

using namespace std;

void reverse(int* c,int cnt)
{
    int i=0,j=cnt-1;
    while(i<j)
    {
        swap(c[i],c[j]);
        i++;
        j--;
    }
}

void solve(int *a,int n)
{
    int *b = new int[n];

    for(int i=0;i<n;i++)
        b[i] = 1;

    for(int i=1;i<n;i++)
    {
        for(int j=i-1;j>=0;j--)
        {
            if(a[j]>=a[i])
                continue;

            int temp = b[j]+1;

            if(temp>b[i])
                b[i] = temp;
        }
    }
}
```

```

        //cout<<endl;

        int max = 0;

        for(int i=0;i<n;i++)

            if(b[i]>max)

                max = b[i];

        cout<<max<<endl;

        int temp = max,cnt=0,k=0;

        int *c = new int[n];

        for(int i=n-1;i>=0;i--)

            if(temp==b[i])

            {

                cnt++;

                c[k++] = a[i];

                temp--;

            }

        reverse(c,cnt);

        cout<<"\tLongest Increasing Subsequence : ";

        for(int i=0;i<cnt;i++)

            cout<<c[i]<<" ";

        cout<<endl;
    }

int main()
{
    int n;

    cout<<"Enter the total no of elements : ";

    cin>>n;

    int *a = new int[n];

    cout<<"Enter the elements : "<<endl;

    for(int i=0;i<n;i++)

        cin>>a[i];

    cout<<"\n\tLength of Longest Increasing Subsequence is : ";

    solve(a,n);

    delete []a;
}

```

O/P:

```
lis_2.cpp > main()
43     ...
44     c[k++] = a[i];
45     temp--;
46 }
47 reverse(c, cnt);
48 cout<<"\nLongest Increasing Subsequence : ";
49 for(int i=0; i<cnt; i++)
50     cout<<c[i]<<" ";
51 cout<<endl;
52 }
53 int main()
54 {
55     int n;
56     cout<<"Enter the total no of elements : ";
57     cin>>n;
58     int *a = new int[n];
59     cout<<"Enter the elements : "<<endl;
60     for(int i=0; i<n; i++)
61     {
62         cin>>a[i];
63     }
64     cout<<"\nLength of Longest Increasing Subsequence is : ";
65     solve(a, n);
66     delete []a;
67 }
```

```
PS C:\Users\ASUS\Documents\TY SEM 2\DAAD\LAB\LAB 6> g++ lis_2.cpp
PS C:\Users\ASUS\Documents\TY SEM 2\DAAD\LAB\LAB 6> .\a.exe
Enter the total no of elements : 8
Enter the elements :
10 22 9 33 21 50 41 60

Length of Longest Increasing Subsequence is : 5
Longest Increasing Subsequence : 10 22 33 41 60
PS C:\Users\ASUS\Documents\TY SEM 2\DAAD\LAB\LAB 6>
```

Time Complexity: $O(n^2)$

Space Complexity: $O(n)$