CS263: Design and Analysis of Algorithm Lab

Name: Hritik Kumar
Roll no.: 202051088

Problem 1

John wants to put everything in its proper order. He follows his rule that all numbers must be arranged in ascending sequence. Unfortunately, this isn't always the case. A "violation" circumstance, according to him, is one in which a lower number appears after a greater number in the collection, violating the ascending order. Determine the total number of such violations given a set of integers.

CODE:

```
#include<bits/stdc++.h>
using namespace std;

int variation(int arr[], int n,int i){
    if(i==(n-1)){
        return 0;
    }
    int count=0;
    for(int j=i+1; j<n; j++)
        if(arr[j]<arr[i])
            count++;

    return (count + variation(arr,n,i+1));
}

int main(){
    int n;
    cin>>n;
    int arr[n];
    for(int i=0; i<n; i++)
        cin>>arr[i];

    cout<<variation(arr,n,0);
    return 0;
}
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS C:\Users\Lenovo\Desktop\c++files\Assignment\Assignment 3> g++ Violation.cpp -o Violation.exe
PS C:\Users\Lenovo\Desktop\c++files\Assignment\Assignment 3> ./Violation.exe
10
59 16 82 9 72 64 54 95 93 85
14
PS C:\Users\Lenovo\Desktop\c++files\Assignment\Assignment 3>
```

Recursion Relation :

T(n) = time to solve problem of size n – Recursive Case

T(0) = time to solve problem of size 0 – Base Case T(n) = T(n-1) + n

Time Complexity :

(Using Iterative Substitution Method)

T(n) = T(n-1) + n

= T(n-2) + 2n

= T(n-3) + 3n

.....

Kth Step : T(n) = T(n-k) + nk

If we take k =n then,

T(n) = T(0) + n2

= O(1) + O(n2)

⇨ Time Complexity : **$O(n^2)$**

Problem 2

You arrive at Disney Land, where you can participate in various activities and games. Each activity or game has different costs. You can choose only one activity. You've been given a set C of n coins in various values $(c_1, c_2, c_3, \ldots, c_n)$. Try to find the fewest number of coins to pay for your activity.

CODE:

```cpp
#include <bits/stdc++.h>
using namespace std;
typedef long long int ll;
```

```cpp
ll dp[101][1005];

vector<int>v;
ll solve(int i,int total)
{
    if(dp[i][total]!=-1)
    {
        return dp[i][total];
    }
    if(i==0)
    {
        if(total==0)
        {
            return 0;
        }
        else
        {
            return 1e9;
        }
    }
    if(v[i-1]<=total)
    {
        return dp[i][total]=min(1+solve(i,total-v[i-1]),solve(i-
1,total));
    }
    else
    {
        return dp[i][total]=solve(i-1,total);
    }
}
int main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    int n;
    cin>>n;
    v.resize(n);
    ll total;
    cin>>total;
    for(int i=0;i<n;i++)
    {
        cin>>v[i];
```

```
    }
    memset(dp,-1,sizeof(dp));

    (solve(n,total)>=1e9)?(cout<<-1):cout<<solve(n,total);
}
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS C:\Users\Lenovo\Desktop\c++files\Assignment\Assignment 3> g++ MinimumCoins.cpp -o MinimumCoins.exe
PS C:\Users\Lenovo\Desktop\c++files\Assignment\Assignment 3> ./MinimumCoins.exe
3
20 10 5
40

Minimum no of coins is: 2
PS C:\Users\Lenovo\Desktop\c++files\Assignment\Assignment 3> ./MinimumCoins.exe
3
20 10 5
25

Minimum no of coins is: 2
PS C:\Users\Lenovo\Desktop\c++files\Assignment\Assignment 3>
```

Recursion Relation:

T(n) = time to solve problem of size n – Recursive Case
T(0) = time to solve problem of size 0 – Base Case
T(n) = T(n-1) + O(1)

Time Complexity:

(Using Iterative Substitution Method)
T(n) = T(n-1) + 1
= T(n-2) + 2
= T(n-3) + 3
.....
Kth Step
T(n) = T(n-k) + k
If we take k =n then,

$T(n) = T(0) + n$

$= O(1) + O(n)$

⇨ Time Complexity : **O(n)**