DAY 32

INTERVIEW BIT PROBLEMS :

1. SUBTRACT

Given a singly linked list, modify the value of first half nodes such that :
1st node's new value = the last node's value - first node's current value
2nd node's new value = the second last node's value - 2nd node's current value,
and so on …

NOTE :
If the length L of linked list is odd, then the first half implies at first floor(L/2) nodes. So, if L = 5, the first half refers to first 2 nodes.
If the length L of linked list is even, then the first half implies at first L/2 nodes. So, if L = 4, the first half refers to first 2 nodes.
Example :
Given linked list 1 -> 2 -> 3 -> 4 -> 5,
You should return **4 -> 2 -> 3 -> 4 -> 5**
as
for first node, 5 - 1 = 4
for second node, 4 - 2 = 2

CODE :

PYTHON

```
class Solution:
    # @param A : head node of linked list
    # @return the head node in the linked list
    def subtract(self, A):
        curr=A
        inp=[]
        c=0
        #storing the values of the linked list in input list
        while curr:
            inp.append(curr.val)
            curr=curr.next
            c+=1
        n=c//2
        #to get the last element in input list

        i=-1       j=0
        curr=A
        #changing the linked list till n/2 index in the linked list
        while j<n:
            curr.val=inp[i]-curr.val
            curr=curr.next
            j+=1
            i-=1
        return A
```

2. Coin Exchange Problem

Description

Problem Statement :-

Given an unlimited supply for few denominations [1,5,7,10] and an amount n, your code should output the number of ways the amount can be paid from the given denominations. The input should have the amount n of which you have to find the number of ways.
The output should print the value of **number of ways the amount can be paid**.

Sample Input
20
Sample Output
15

Sample Input
12
Sample Output
6

CODE :

JAVA

```
import java.util.Scanner;
import java.util.Arrays;
public class Source {
       public static int count(int[] d, int amount) {

         int[][] V = new int[amount + 1][d.length];

         /* base solution*/
         for(int i=0;i<amount+1;i++){
              V[i][0]=1;
         }
         for(int j=0;j<d.length;j++){
              V[0][j]=1;
         }
         /* rest of the matrix using recursion relation */
         for(int j=1;j<d.length;j++){
              for(int i=1;i<=amount;i++){
 /*check if the coin value is less than the amount needed*/
                   if (d[j] <= i ) {
                        V[i][j] = V[i][j-1]+ V[i - d[j]][j];
                   } else {
                        V[i][j] =V[i][j-1];
                   }
              }
```

```
        }
        return V[amount][d.length-1];
    }

    public static void main(String args[]) {
        int[] d = {1,5,7,10};
        Scanner scan = new Scanner(System.in);
        int amount = scan.nextInt();
        System.out.println(count(d, amount));
scan.close();
}
}
```