Perfect Sum Problem

I/p: arr[] = {2, 3, 5, 6, 8, 10}

N = 6

sum = 10

o/p: 3

{2, 3, 5}

{10}

{2, 8}

Recursive solution

```
bool CountSubset( vector<int> arr, int n, int sum)
{
    // Base condition
    if (n == 0) return 0;
    if (sum == 0) return 1;

    // Logic
    if (arr[n-1] <= sum)
    {
        return CountSubset(arr, n-1, sum-arr[n-1]) +
               CountSubset(arr, n-1, sum);
    }
    else
        CountSubset(arr, n-1, sum);
}
```

Now, In previous code, Logic works fine ✓

↳ there is some issue with Base condition *

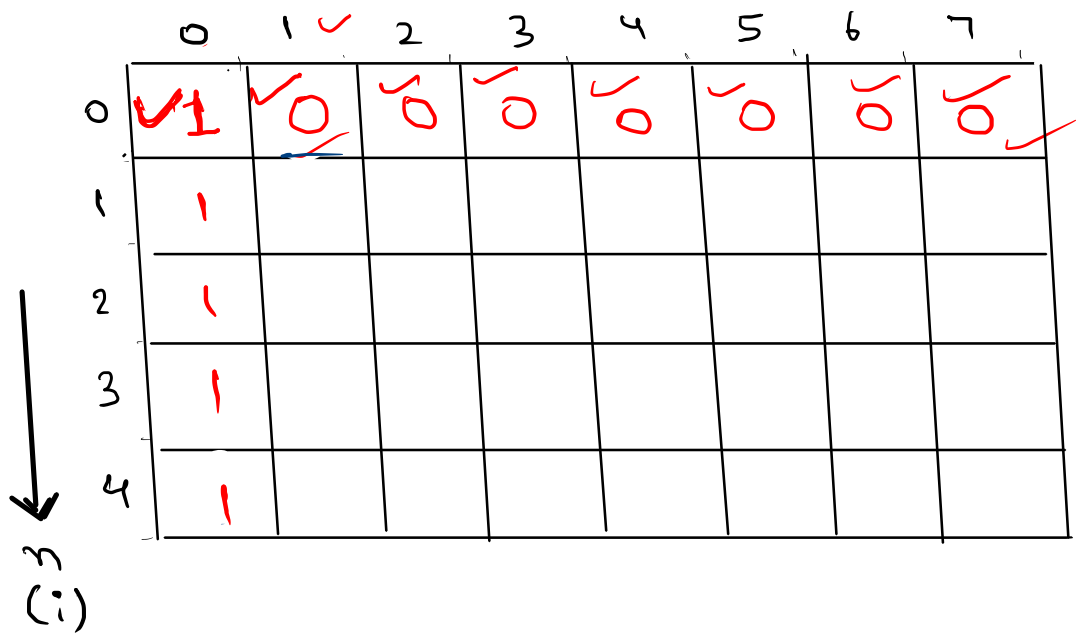
In, our previous Question
we Assumed that, we have only +ve Integers] **

↳ arr[] = {1, 3, 5, 2}

↳ But. In this question, ✓
we have non-negative Integers. ✓

↳ means we have
arr[] = { 3, 2, 0, 1, 5 }

In previous question, we did Initialisation like this, which is ✓ Completely RIGHT for +ve Integers



How, modify for non-negative Integers

↳ if (n==0) and Sum >= 1

↳ false → 0

↳ if (n==0) and Sum == 0

↳ 1

← arr[] = { }

Sum = 0

Sum = 5 ✗

1, 2, 3, ... 10

{ }

In previous question, we did Initialisation like this, which is Completely RIGHT for +ve Integers

n (i) \ sum	0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0
1	1							
2	1							
3	1							
4	1							

#2 if(n==1) → [Single Integer Array]

sum==0

↳ arr[] = {0} = 2

arr[] = {5} = 1

{Ø} → {}, {0}

{} 1,2,3.

sum!=0

↳ arr[] = {5} = 1

else → 0

sum=5

{5} 1,2,3

** if(n>=2)

arr[] = {0,0}

n=2

sum = 0

↳ {0}, {0}, {0,0}, {} = 4

In previous question, we did Initialisation like this, which is Completely RIGHT for +ve Integers

	0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0
1								
2								
3								
4								

n (i)

non -ve Integer

{0}

if(n==1) → single Integer Array

↓

sum==0

↳ arr[] = {0} = 2

arr[] = {5} = 1

sum != 0

sum=5

↳ arr[] = {5} = 1

else → 0

* if(n>=2)

arr[] = {0, 0}

sum = 0

↳ {0}, {0}, {0, 0}, {} = 4 → Can't predict

So Finally
Base condition.

```
if (n==0 and sum>0 ) return 0;
else if ( n==1)
{   if (sum==0)
        if( arr[0]==0 ) return 2;
        else return 1;
    else
        if ( arr[0]==sum) return 1;
        else return 0;
}
else if ( sum==0 and n==0 ) return 1;
```

0 0 0 0 0

1

But

Recursive
↓
memoization

20