HASHING DATA STRUCTURE

- Subarray with 0 sum

- array of positive and negative numbers.

  Find if there is a subarray (of size at-least one) with 0 sum

- What is SubArray ?

subArray of { 10, 20, 30 }

{ 10 }

{ 10, 20 }

{ 10, 20, 30 }

{ 20 }

{ 20, 30 }

{ 30 }

Hello World

HASHING DATA STRUCTURE

□ Subarray with 0 sum

□ array of positive and negative numbers.

Find if there is a subarray (of size at-least one) with 0 sum

□ Arr = { 4, -3, 2,1 }

Solution : { -3, 2, 1 }

Sum = 0

NAIVE SOLUTION : 1) pick on element

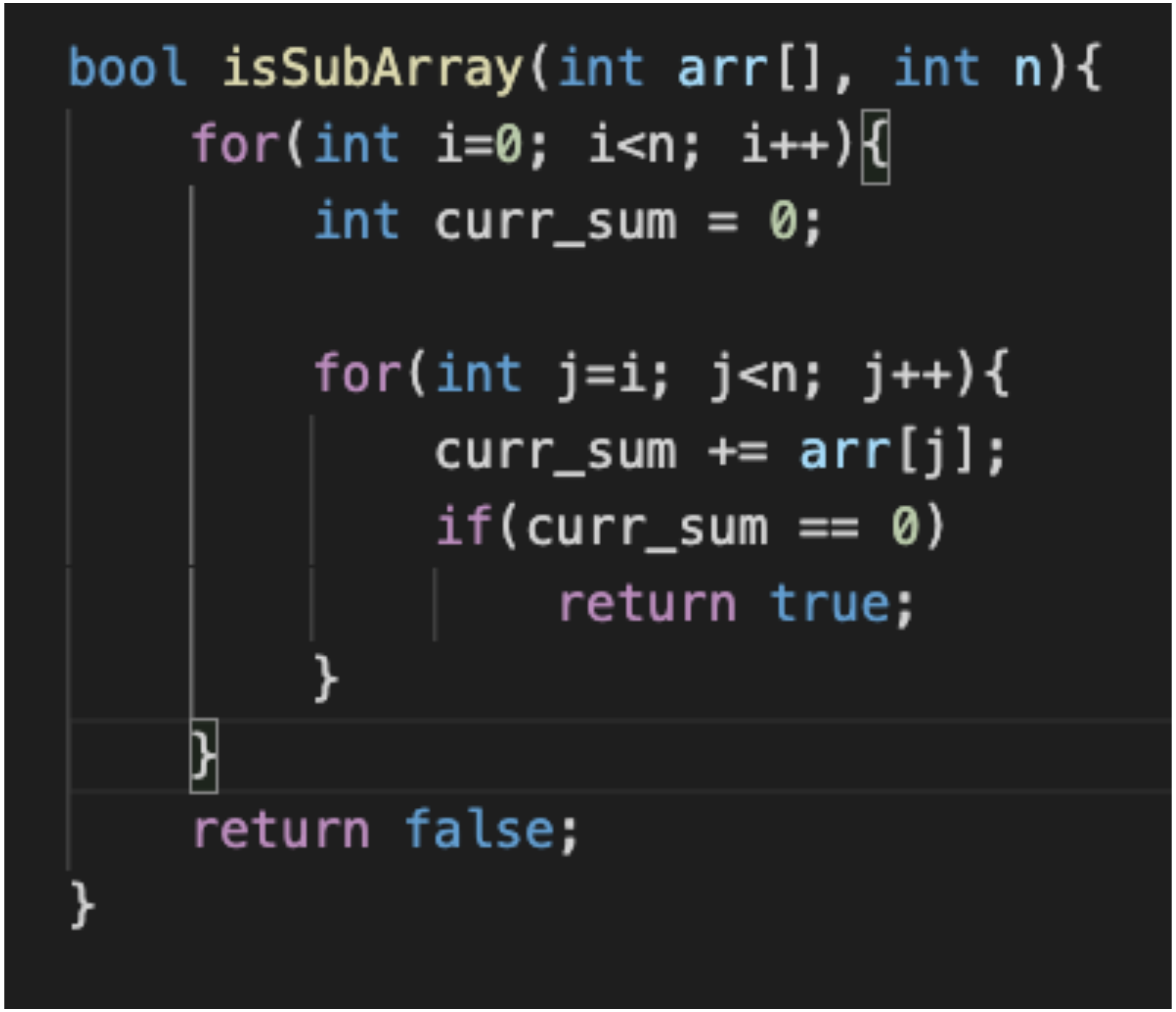
2) create a subarray from it

3) check sum of it

Hello World

HASHING DATA STRUCTURE

- **Arr = { 4, -3, 2,1 }**



```
bool isSubArray(int arr[], int n){
    for(int i=0; i<n; i++){
        int curr_sum = 0;

        for(int j=i; j<n; j++){
            curr_sum += arr[j];
            if(curr_sum == 0)
                return true;
        }
    }
    return false;
}
```

Time Complexity : O (N ^2)

Hello World

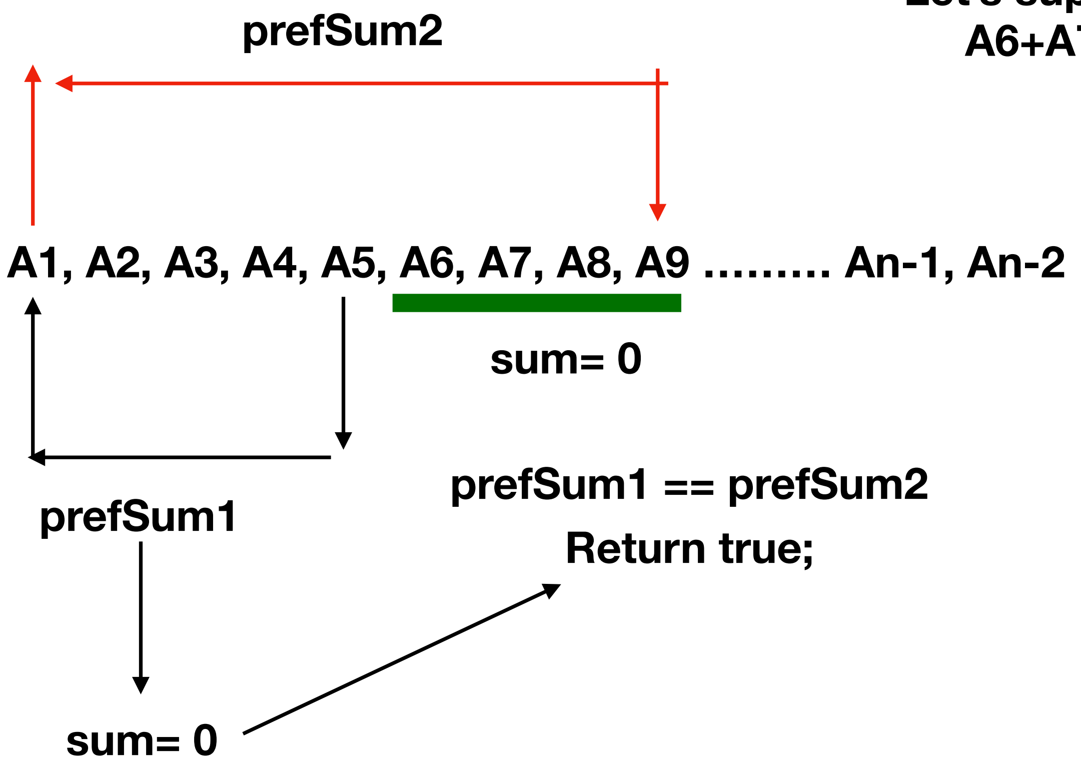
HASHING DATA STRUCTURE

- **Efficient and Best solution :**

We are using Hashing and Prefix sum

Understand the concept

Let's suppose Subarray
A6+A7+A8+A9 = 0



prefSum2

A1, A2, A3, A4, A5, A6, A7, A8, A9 ……… An-1, An-2

sum= 0

prefSum1

prefSum1 == prefSum2

Return true;

sum= 0

Hello World

HASHING DATA STRUCTURE

🟧 **Arr = { -3, 4, -3, -1, 1 }**

pref_Sum = 0

Hash = { }

I =0 pref_Sum = -3

Hash = { -3 }

I =1 pref_Sum = 1

Hash = { -3, -1 }

I =2 pref_Sum = -2

Hash = { -3, -1, -2 }

I =3 pref_Sum = -3

Already present in hash

Return true

Hello World