Ace your JavaScript Interview. Get my ebook. 100 solved Javascript, 20 solved React, & 2 frontend system design questions (1160+ copies sold). Get a Free preview.

Advertisements

Program to check if a subarray with 0 sum exits or not

Posted on January 21, 2019 | by Prashant Yadav

Posted in Algorithms, Arrays | Tagged Easy

Practically prepare for your JavaScript interview

JavaScript Revision
JavaScript-Concept Based Problems
Data Structures
Algorithms
Machine Coding
Web Fundamentals

Advertisements

An algorithm to check if a subarray with 0 sum exits or not.

We will implement a simple algorithm in javascript to check if an array has a subarray with 0 sum or not. Everything will be written in ES6.

Example

```
Input:
[3, 4, -7, 3, 1, 3, 1, -4, -2, -2]

Output:
true
[3, 4, -7]
[4, -7, 3]
[-7, 3, 1, 3]
[3, 1, -4]
[3, 1, 3, 1, -4, -2, -2]
[3, 4, -7, 3, 1, 3, 1, -4, -2, -2]
```

A naive solution O(n ^ 2).

Implementation

- We are going to traverse and sum all the subarrays of the given array and check if they are equal to 0.
- If they are equal to 0 then return `true` else return `false`.

```
let subWithZero = (arr) => {
  //loop through the array
  for(let i = 0; i < arr.length; i++){
    let sum = arr[i];
    //Check if initial item is zero then return true
    if(sum === 0){
      return true;
    }

    for(let j = i; j < arr.length; j++){
      //If there is any subarray with zero then return true
      sum += arr[j];
      if(sum === 0){
        return true;
      }
    }
  }

  //Else return false
  return false;
}
```

```
Input:
console.log(sumWithZero([3, 4, -7, 3, 1, 3, 1, -4, -2, -2]));
console.log(sumWithZero([3, 5]));

Output:
true
false
```

Time complexity: O(n ^ 2).

Space complexity: O(1).

Time and Space complexity

- We traversing twice with inner loop, so Time complexity is O(n^2).
- We are using constant space, so Space complexity is O(1).

Using Set.

Implementation

- We will be using Set to store the sum and check if there is sum with 0 or not.
- If there is sum with 0 then return `true` else return `false`.

```
let subWithZero = (arr) => {
 //create a new set
 let set = new Set();

 //add 0 to handle the case when first  element in array is 0
 set.add(0);

  //To calculate the sum
  let sum = 0;

  //loop through the array
  for(let i = 0; i < arr.length; i++){

    //calculate the sum of the subarrays
    sum += arr[i];

    //if sum is already there, then subarray with 0 is found
    if(set.has(sum)){
       return true;
    }

    //Add sum to the set
    set.add(sum);
  }

  //Return false by default
  return false;
}
```

```
Input:
console.log(sumWithZero([3, 4, -7, 3, 1, 3, 1, -4, -2, -2]));
console.log(sumWithZero([3, 5]));

Output:
true
false
```

Time complexity: O(n).

Space complexity: O(n).

Time and Space complexity

- We are iterating through the given array only once, so Time complexity is O(n).
- We are using Set to store the subarrays, so Space complexity is O(n).

Prepare for your **JavaScript Interview** practically on each Interview rounds and grab that job.

BEGIN LEARNING

Recommended Posts:

Minimum characters to delete to make string anagram

Program to check if two stacks are equal

4 sum problem

Find digital root of a given number

Find the largest sum of contiguous subarray

Count number of sub string recursively

Learn how to reverse words in a string

Program to print the pyramid pattern

Check if two string are anagram of each other

How to create key value array in javascript

Prev   Next

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Start typing...

Name*

Name

Email*

Email

POST COMMENT

Advertisements

About Us Contact Us Privacy Policy Advertise

Handcrafted with ❤somewhere in **Mumbai**

© 2023 LearnersBucket | Prashant Yadav