An array is a data structure consisting of a collection of elements (values or variables), each identified by at least one array index or key. The simplest type of data structure is a linear array, also called one-dimensional array.

An array is stored such that the position of each element can be computed from its index tuple by a mathematical formula.

For example, an array of 10 32-bit (4 bytes), with indices 0 through 9, may be stored as 10 words at memory addresses 2000, 2004, 2008, ... 2036, so that the element with index i has address 2000 + (i × 4).

The memory address of the first element of an array is called first address, foundation address, or base address.

Tables are often implemented in the form of arrays, especially lookup tables; the word table is sometimes used as a synonym of array.

Arrays are among the oldest and most important data structures, and are used by almost every program. They are also used to implement many other data structures, such as lists and strings. They effectively exploit the addressing logic of computers. In most modern computers and many external storage devices, the memory is a one-dimensional array of words, whose indices are their addresses. Processors, especially vector processors, are often optimized for array operations.

The array is a data type of programming languages that consists of a collection of values or variables that can be selected by one or more indices computed at run-time.

Array types are often implemented by array structures; however, in some languages they may be implemented by hash tables, linked lists, search trees, or other data structures.

There are three ways in which the elements of an array can be indexed:

- 0 (zero-based indexing): The first element of the array is indexed by subscript of 0.
- 1 (one-based indexing): The first element of the array is indexed by subscript of 1.
- n (n-based indexing): The base index of an array can be freely chosen. Usually programming languages allowing n-based indexing also allow negative index values and other scalar data types like enumerations, or characters may be used as an array index.

DIFFERENCE

1D

1. 1D Array contains single row and multiple columns.
2. 1D Array is a simple collection of elements.
3. Ex:-

2d

1. 2D Array contains multiple row and multiple columns.
2. 2D Array is a collection of 1D Array .
3. Ex:-

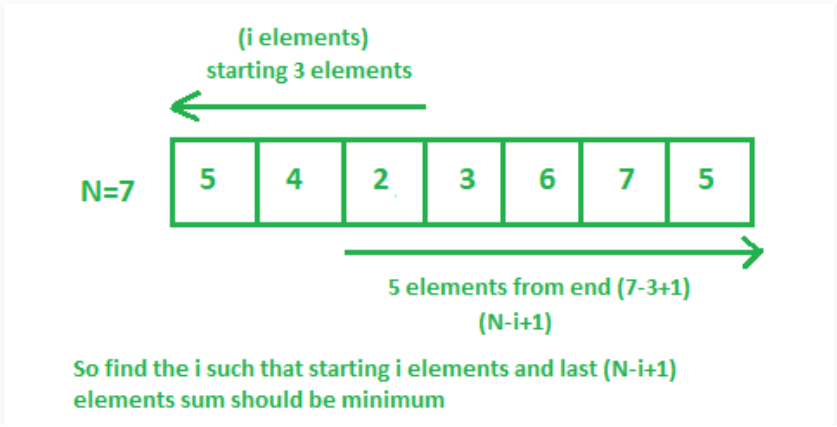
1	2	3	4
A	1	2	3
B	4	5	6
C	7	8	9

N-BASED INDEXING

Given an Array of integers. The task is to find the index i in the array at which the value of **prefixSum(i) + suffixSum(i)** is minimum.

Note:

- PrefixSum(i) = The sum of first i numbers of the array.
- SuffixSum(i) = the sum of last N − i + 1 numbers of the array.
- 1-based indexing is considered for the array. That is index of the first element in array is 1.



Example of N-Based Indexing:

```java
// Java program to find the index with
// minimum sum of prefix and suffix
// sums in an Array

import java.io.*;

class GFG {

static int indexMinSum(int arr[], int n)
{
    // Initialization of the min value
    int min = arr[0];
    int index = 0;

    // Find minimum element in the array
    for (int i = 1; i < n; i++) {
        if (arr[i] < min) {

            // store the index of the
            // current minimum element
            min = arr[i];
            index = i;
        }
    }

    // return the index of min element
    // 1-based index
    return index + 1;
}

// Driver Code
    public static void main (String[] args) {
    int arr[] = { 6, 8, 2, 3, 5 };
    int n =arr.length;
    System.out.println( indexMinSum(arr, n));
    }
}
```

Output:

```
3
```