<Codemithra />™

Explore | Expand | Enrich

TEST TIME ON DATE AND TIME

URL: https://qrgo.page.link/Y1cEC

QR CODE:

<Codemithra /> ™

Explore | Expand | Enrich

ARRAYS

JAVA ARRAYS

- **Java array** is an object which contains elements of a similar data type.
- Additionally, The elements of an array are stored in a contiguous memory location.
- It is a data structure where we store similar elements.
- We can store only a fixed set of elements in a Java array.
- Array in Java is index-based, the first element of the array is stored at the 0th index, 2nd element is stored on 1st index and so on.

ARRAY REPRESENTATION

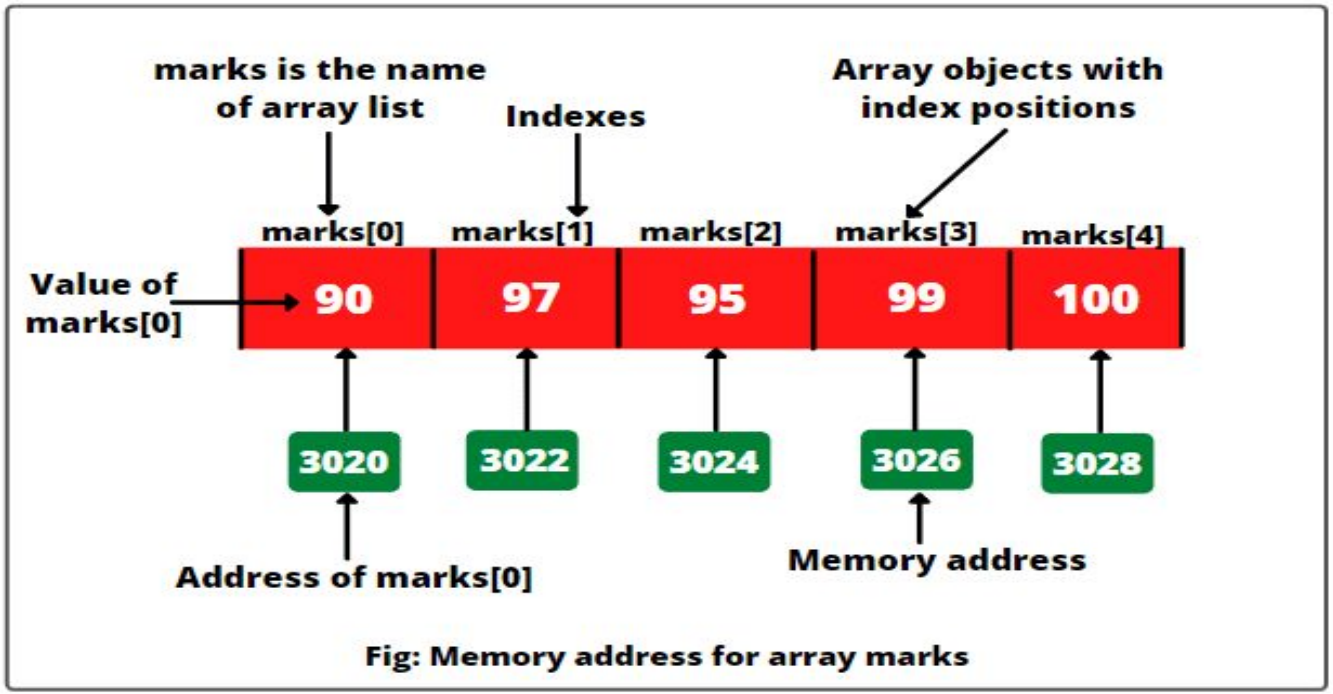


Fig: Memory address for array marks

ADVANTAGES

Code Optimization: It makes the code optimized, we can retrieve or sort the data efficiently.

Random access: We can get any data located at an index position.

Types of Array in java

- Single Dimensional Array
- Multidimensional Array

FEATURES OF ARRAYS

- They can even hold the reference variables of other objects
- They are created during runtime
- They are dynamic, created on the heap
- The Array length is fixed

ARRAY DECLARATION

The preceding program declares an array (named an Array) with the following line of code

```
int[] anArray;
```

- An array's type is written as type[], where type is the data type of the contained elements
- the brackets are special symbols indicating that this variable holds an array
- The size of the array is not part of its type (which is why the brackets are empty)

CREATING,INTINALING , ACCESSING AN ARRAY

One way to create an array is with the new operator

arrayRefVar = new dataType[arraySize];

- It creates an array using new dataType[arraySize]
- It assigns the reference of the newly created array to the variable arrayRefVar

CREATING,INTINALING , ACCESSING AN ARRAY

- If this statement is missing, then the compiler prints an error like the following, and compilation fails

```
anArray = new int[10];// create an array of integers
```

- ArrayDemo.java:4: Variable an Array may not have been initialized

INITIALIZING AN ARRAY

The next few lines assign values to each element of the array

```
anArray[0] = 100;
// initialize first element
anArray[1] = 200;
// initialize second element
anArray[2] = 300; // and so forth
```

ACCESSING AN ARRAY

Each array element is accessed by its numerical index

```
System.out.println("Element 1 at index 0: " + anArray[0]);
System.out.println("Element 2 at index 1: " + anArray[1]);
System.out.println("Element 3 at index 2: " + anArray[2]);
```

ARRAY INITIALIZATION -2

```
int[] age = {12, 4, 5, 2, 5};
```

- This statement creates an array and initializes it during declaration
- The length of the array is determined by the number of values provided which is separated by commas. In our example, the length of age array is 5

EXAMPLE TO INITIALIZE

```
class Main {
    public static void main(String[] args) {
        int[] age = {12, 4, 5, 2, 5};
        for (int i = 0; i < 5; ++i) {
            System.out.println("Element at index " + i +": " + age[i]);
        }
    }
```

```
Element at index 0 :12
Element at index 1: 14
Element at index  2: 5
```

SINGLE DIMENSIONAL ARRAY

Explore | Expand | Enrich

Syntax to Declare an Array in Java

dataType[] arr; (or)
dataType []arr; (or)
dataType arr[];

Instantiation of an Array in Java

arrayRefVar=**new** datatype[size];

EXAMPLE OF JAVA ARRAY

```
class Testarray{
public static void main(String args[]){
int a[]=new int[5];//declaration and instantiation
a[0]=10;//initialization
a[1]=20;
a[2]=70;
a[3]=40;
a[4]=50;
//traversing array
for(int i=0;i<a.length;i++)//length is the property of array
System.out.println(a[i]);
}}
```

JAVA ARRAYS

Declaration, instantiation and initialization

```
class Testarray1{
public static void main(String args[]){
int a[]={33,3,4,5};//declaration, instantiation and initialization
//printing array
for(int i=0;i<a.length;i++)//length is the property of array
System.out.println(a[i]);
}}
```

For-each Loop JAVA ARRAYS

We can also print the Java array using **for-each loop.**

The Java for-each loop prints the array elements one by one.

It holds an array element in a variable, then executes the body of the loop.

The syntax of the for-each loop is given below:

```
for(data_type variable:array){
//body of the loop
}
```

EXAMPLE

```
class Testarray1{

public static void main(String args[]){

int arr[]={33,3,4,5};

//printing array using for-each loop

for(int i:arr)

System.out.println(i);

}}
```

JAVA ARRAYS

ETHNUS
Explore | Expand | Enrich

Passing Array to a Method in Java

```
class Testarray2{
 static void min(int arr[]){
int min=arr[0];
for(int i=1;i<arr.length;i++)
 if(min>arr[i])
  min=arr[i];
  System.out.println(min);
}
public static void main(String args[]){
int a[]={33,3,4,5};//declaring and initializing an array
min(a);//passing array to method  }}
```

JAVA ARRAYS

ArrayIndexOutOfBoundsException

The Java Virtual Machine (JVM) throws an ArrayIndexOutOfBoundsException if length of the array in negative, equal to the array size or greater than the array size while traversing the array.

```
public class TestArrayException{
public static void main(String args[]){
int arr[]={50,60,70,80};
for(int i=0;i<=arr.length;i++){
System.out.println(arr[i]);
}
}}
```

EXAMPLE

Let’s create a program where we will find the length of an array.

```
public class OneDArr {
public static void main(String[] args) {
    int[ ] num = {2, 4, 6, 8, 10, 12, 14};// Declare and initialize an array of five integer values.
    System.out.println("Length of array: " +num.length);// Display the length of array.
 }
}
```

EXAMPLE

Let’s create a program where we will accept the marks obtained by a student into a one-dimensional array from the keyboard and finds total marks and percentage of marks. Assume that the maximum mark in any subject is 100.

```
import java.util.Scanner;
public class OneDArr {
public static void main(String[] args) {
 Scanner sc = new Scanner(System.in);
  System.out.println("In how many subject have you given exams?");
    int n = sc.nextInt();
      int[ ] marks = new int[n];
    System.out.println("Enter your marks obtained in subjects:");
```

CONT.,

```
for(int i = 0; i < n; i++) {
        marks[i] = sc.nextInt();}
int total = 0;
   for(int i = 0; i < n; i++) {
         total += marks[i]; }
     System.out.println("Total marks: " +total);
     float percentage = (float)total/n; // Casting.
     System.out.println("Percentage: " +percentage+ "%");}}
```

MERGE TWO SORT: Q01

Given two sorted arrays, the task is to merge them in a sorted manner.

Examples:

Input: arr1[] = { 1, 3, 4, 5}, arr2[] = {2, 4, 6, 8}

Output: arr3[] = {1, 2, 3, 4, 4, 5, 6, 8}

Input: arr1[] = { 5, 8, 9}, arr2[] = {4, 7, 8}

Output: arr3[] = {4, 5, 7, 8, 8, 9}

CODE: Q01


```
class Main
{
public static void mergeArrays(int[] arr1,
int[] arr2, int n1, int n2,zint[] arr3)
    {
        int i = 0, j = 0, k = 0;
      // Traverse both array
        while (i<n1 && j <n2)
        {
 if (arr1[i] < arr2[j])
                arr3[k++] = arr1[i++];
            else
                arr3[k++] = arr2[j++];
        }
 // Store remaining elements of first
array
 while (i < n1)
            arr3[k++] = arr1[i++];
 // Store remaining elements of second
array
        while (j < n2)
            arr3[k++] = arr2[j++];
    }
```

CONT.,

```
public static void main (String[] args)
    {
        int[] arr1 = {1, 3, 5, 7};
        int n1 = arr1.length;

        int[] arr2 = {2, 4, 6, 8};
        int n2 = arr2.length;

        int[] arr3 = new int[n1+n2];

        mergeArrays(arr1, arr2, n1, n2, arr3);

        System.out.println("Array after merging");
        for (int i=0; i < n1+n2; i++)
            System.out.print(arr3[i] + " ");
    }
}
```

EQUILIBRIUM INDEX: Q02

Explore | Expand | Enrich

Equilibrium index of an array is an index such that the sum of elements at lower indexes is equal to the sum of elements at higher indexes. For example, in an array A

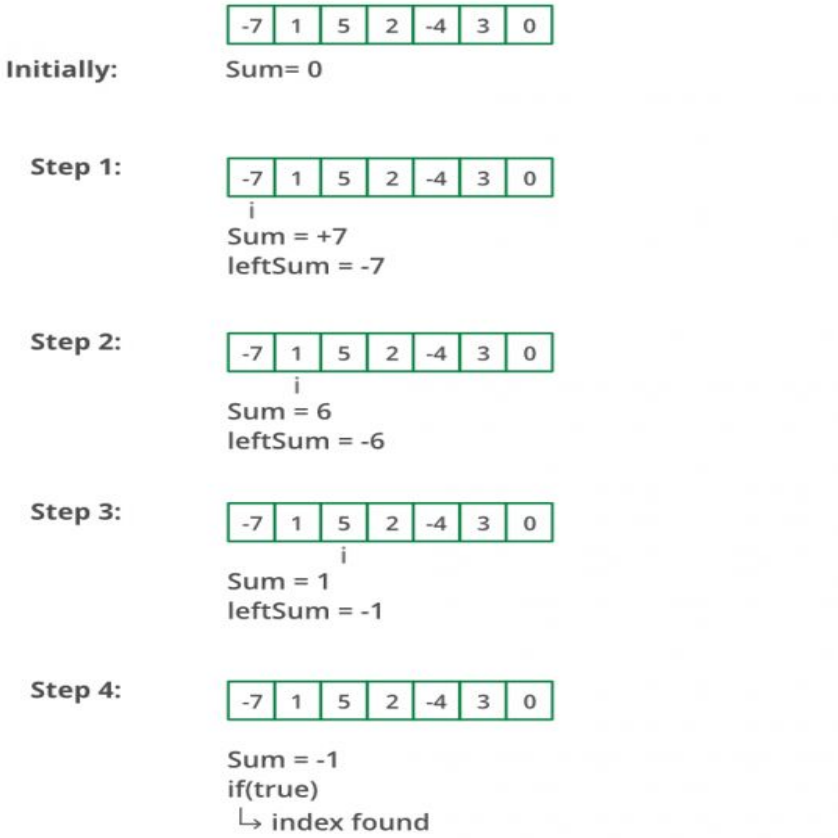
Input: A[] = {-7, 1, 5, 2, -4, 3, 0}

Output: 3

3 is an equilibrium index, because:

A[0] + A[1] + A[2] = A[4] + A[5] + A[6]

EXPLANATION: Q02



CODE: Q02


```
Class Main {
int ei(int arr[], int n)
{
int sum = 0;
 // initialize sum of whole array
int leftsum = 0;
// initialize leftsum

/* Find sum of the whole array */
for (int i = 0; i < n; ++i)
sum += arr[i];
for (int i = 0; i < n; ++i) {
sum -= arr[i];
// sum is now right sum for index i

if (leftsum == sum)
return i;
leftsum += arr[i];
}

// If no equilibrium index found, then
return 0

return -1;

}
```

CONT.,

```
public static void main(String[] args)
    {
        int arr[] = { -7, 1, 5, 2, -4, 3, 0 };
        int arr_size = arr.length;
                int k=ei(arr, arr_size);
        System.out.println("First equilibrium index is " + k);

    }
}
```

K'TH UNSORTED ARRAY: Q03

Given an array and a number k where k is smaller than size of array, we need to find the k'th smallest element in the given array. It is given that ll array elements are distinct.

1.

Input:

arr[] = {7, 10, 4, 3, 20, 15}

k = 3

Output: 7.

2.

Input:

arr[] = {7, 10, 4, 3, 20, 15}

k = 4

Output: 10

CODE: Q03

Explore | Expand | Enrich

```
import java.util.Arrays;
import java.util.Collections;

Class Main
{
public static int kthSmallest(Integer [] arr, int k)
    {
        // Sort the given array
        Arrays.sort(arr);

        // Return k'th element in
        // the sorted array
        return arr[k-1];
    }
public static void main(String[] args)
    {
 Integer arr[] = new Integer[]{12, 3, 5, 7, 19};
        int k = 2;
System.out.print( "K'th smallest element is "+ kthSmallest(arr, k) );
    }
}
```

PAIR OF INTEGERS ARRAY: Q04

Given an array arr of size N and an integer K.

The task is to find the pair of integers such that their sum is maximum and but less than K

Input : arr = {30, 20, 50} ,

K = 70

Output : 30, 20

30 + 20 = 50 which is maximum possible sum which is less than K

CODE: Q04


```
import java.util.Arrays;

class Main
{

static void Max_Sum(int arr[], int n, int k)
{
    // To store the break point
    int p = n;

    // Sort the given array
    Arrays.sort(arr);

    // Find the break point
    for (int i = 0; i < n; i++)
    {
        // No need to look beyond i'th index
        if (arr[i] >= k)
        {
            p = i;
            break;
        }
    }
```

CODE: Q04

```
int maxsum = 0, a = 0, b = 0;

    // Find the required pair
    for (int i = 0; i < p; i++)
    {
        for (int j = i + 1; j < p; j++)
        {
            if (arr[i] + arr[j] < k &&
                arr[i] + arr[j] > maxsum)
            {
                maxsum = arr[i] + arr[j];

                a = arr[i];
                b = arr[j];
            }
        }
    }

    // Print the required answer
    System.out.print( a + " " + b); }
```

```
public static void main (String[] args)
{
    int []arr = {5, 20, 110, 100, 10};
    int k = 85;

    int n = arr.length;

    // Function call
    Max_Sum(arr, n, k);
}
}
```

/ethnuscodemithra

Ethnus Codemithra

/ethnus

/code_mithra

<Codemithra />™

https://learn.codemithra.com

ETHNUS™

Explore | Expand | Enrich

codemithra@ethnus.com

+91 7815 095 095

+91 9019 921 340