**1D Array Solution**

Q1: Write a program to print the sum of all the elements present on even indices in the given array.

Input 1: arr[] = {3,20,4,6,9}

Output 1: 16

Input 1: arr[] = {4,3,6,7,1}

Output 1: 11

**public class SumEvenIndices {**

**public static void main(String[] args) {**

**int[] arr1 = {3, 20, 4, 6, 9};**

**int[] arr2 = {4, 3, 6, 7, 1};**

**System.out.println("Sum of elements at even indices in arr1: " + sumEvenIndices(arr1));**

**System.out.println("Sum of elements at even indices in arr2: " + sumEvenIndices(arr2));**

**}**

**public static int sumEvenIndices(int[] arr) {**

**int sum = 0;**

**for (int i = 0; i < arr.length; i += 2) {**

**sum += arr[i];**

**}**

**return sum;**

**}**

Q2: Write a program to traverse over the elements of the array using for each loop and print all even elements.

Input 1: arr[] = {34,21,54,65,43}

Output 1: 34 54

Input 1: arr[] = {4,3,6,7,1}

Output 1: 4 6

public class PrintEvenElements {

public static void main(String[] args) {

int[] arr1 = {34, 21, 54, 65, 43};

int[] arr2 = {4, 3, 6, 7, 1};

System.out.print("Even elements in arr1: ");

printEvenElements(arr1);

System.out.print("\nEven elements in arr2: ");

printEvenElements(arr2);

}

public static void printEvenElements(int[] arr) {

for (int element : arr) {

if (element % 2 == 0) {

System.out.print(element + " ");

}

}

}

}

Q3: Write a program to calculate the maximum element in the array.

Input 1: arr[] = {34,21,54,65,43}

Output 1: 65

Input 1: arr[] = {4,3,6,7,1}

Output 1: 7

public class FindMaxElement {

public static void main(String[] args) {

int[] arr1 = {34, 21, 54, 65, 43};

int[] arr2 = {4, 3, 6, 7, 1};

System.out.println("Maximum element in arr1: " + findMaxElement(arr1));

System.out.println("Maximum element in arr2: " + findMaxElement(arr2));

}

public static int findMaxElement(int[] arr) {

int max = arr[0]; // Assume first element is the maximum

for (int i = 1; i < arr.length; i++) {

if (arr[i] > max) {

max = arr[i]; // Update max if current element is greater

}

}

return max;

}

}

Q4: Write a program to find out the second largest element in a given array.

Input 1: arr[] = {34,21,54,65,43}

Output 1: 54

public class SecondLargestElement {

public static void main(String[] args) {

int[] arr1 = {34, 21, 54, 65, 43};

System.out.println("Second largest element in arr1: " + findSecondLargest(arr1));

}

public static int findSecondLargest(int[] arr) {

int firstLargest = Integer.MIN\_VALUE;

int secondLargest = Integer.MIN\_VALUE;

for (int element : arr) {

if (element > firstLargest) {

secondLargest = firstLargest;

firstLargest = element;

} else if (element > secondLargest && element != firstLargest) {

secondLargest = element;

}

}

return secondLargest;

}

}

Q5: Given an array. Find the first peak element in the array. A peak element is an element that is greater than its just left and just right neighbor.

Input 1: arr[] = {1,3,2,6,5}

Output 1: 6

Input 2:  arr[]

public class FirstPeakElement {

public static void main(String[] args) {

int[] arr1 = {1, 3, 2, 6, 5};

int[] arr2 = {4, 3, 6, 7, 1};

System.out.println("First peak element in arr1: " + findFirstPeak(arr1));

System.out.println("First peak element in arr2: " + findFirstPeak(arr2));

}

public static int findFirstPeak(int[] arr) {

int n = arr.length;

// If the array has less than 3 elements, there's no peak element possible

if (n < 3) {

return -1; // No peak found

}

// Traverse the array starting from the second element to the second last element

for (int i = 1; i < n - 1; i++) {

if (arr[i] > arr[i - 1] && arr[i] > arr[i + 1]) {

return arr[i]; // Return the first peak element

}

}

return -1; // No peak element found

}

}