

## 1 Navigate to <https://campus.binarykeeda.com>

- The total duration of the test is
- The test consists of
- Sections may be of type **Quiz** or **Coding**.
- Each **Quiz** section includes MCQs, MSQs, and short answer questions.
- Each **Coding** section presents a problem statement — you must solve it using the provided code editor.
- **Tab change monitoring** is enabled. Switching tabs may negatively impact your result.
- **Do not close or refresh the browser tab once the test starts.** Doing so may lead to data loss or disqualification.
- **Plagiarism is strictly prohibited.** All code submissions must be your own.
- In **Quiz sections**, question status is color-coded:
  - Attempted (Done)
  - Seen but not attempted
  - Unseen / Not visited
  - Current Question
- The test will be **auto-submitted** once the time runs out. Please track the on-screen timer.
- Once a section is submitted, you may **not be able to return** to it. Double-check before submitting.
- **Do not use any external help** like ChatGPT, online IDEs, or discuss answers with anyone.
- **Use a laptop or desktop.** Mobile devices are not recommended.
- **Ensure a stable internet connection.** Disruptions may affect your test progress.

**By clicking "Start Test", you agree to follow the rules above. Any violation may lead to test cancellation.**

## 2 Click "Start Test"

- Each **Quiz** section includes MCQs, MSQs, and short answer questions.
- Each **Coding** section presents a problem statement — you must solve it using the provided code editor.
- **Tab change monitoring** is enabled. Switching tabs may negatively impact your result.
- **Do not close or refresh the browser tab once the test starts.** Doing so may lead to data loss or disqualification.
- **Plagiarism is strictly prohibited.** All code submissions must be your own.
- In **Quiz sections**, question status is color-coded:
  - ☐ Attempted (Done)
  - ☐ Seen but not attempted
  - ☐ Unseen / Not visited
  - ☐ Current Question
- The test will be **auto-submitted** once the time runs out. Please track the on-screen timer.
- Once a section is submitted, you may **not be able to return** to it. Double-check before submitting.
- **Do not use any external help** like ChatGPT, online IDEs, or discuss answers with anyone.
- **Use a laptop or desktop.** Mobile devices are not recommended.
- **Ensure a stable internet connection.** Disruptions may affect your test progress.

By clicking "Start Test", you agree to follow the rules above. Any violation may lead to test cancellation.

Start Test

## 3 Click on Section with Blue background



### Aptitude

Quiz  
20 Questions for 30 minutes



### Computer Science Basics

Quiz  
10 Questions for 20 minutes



### Autmata

Coding  
2 Questions for 45 minutes



### Automata Fix

Coding  
3 Questions for 45 minutes



## 4 Attempt Question



29 min 56 sec

Q1. The average of 20 numbers is 35. Later it is found that a number 65 was misread as 56. What is the correct average?

☐ 35.45

☐ 35.25

☐ 34.75

☐ 35.50

Clear choice

Previous

Next

### Question Navigator



## 5 Click "Submit" to submit section

☐ 35.25

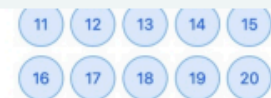
☐ 34.75

☐ 35.50

Clear choice

Previous

Next



- Current
- Attempted
- Seen
- Unseen

SUBMIT

## 6 Click "Confirm"

The screenshot shows a confirmation dialog box in the center of the screen. The dialog box has a white background and a thin border. It contains the text "Are you sure you want to submit?" in bold. Below this, it says "This action is **irreversible**. Once submitted, your changes cannot be undone." At the bottom of the dialog box, there are two buttons: "Cancel" and "Confirm". The "Confirm" button is green and has an orange circle highlighting it. In the background, a quiz interface is visible. On the left, there are four radio button options with values: 35.45 (selected), 35.25, 34.75, and 35.50. On the right, there is a grid of 20 numbered buttons (1-20). Below the grid, there is a legend with four items: "Current" (blue circle), "Attempted" (green circle), "Seen" (orange circle), and "Unseen" (blue circle). At the bottom right, there is a "SUBMIT" button.

## 7 Click on another section now

The screenshot shows a list of four quiz sections. Each section is represented by a horizontal card. The first two cards, "Aptitude" and "Computer Science Basics", have a green background and a green checkmark icon on the right. The third card, "Autmata", has a blue background, an orange circle icon in the center, and a right arrow icon on the right. The fourth card, "Automata Fix", has a light purple background and a lock icon on the right. Below the list, there is a note: "Note : All the sections are mandatory to attempt then only the submission would be counted".

**Aptitude**  
Quiz  
20 Questions for 30 minutes

**Computer Science Basics**  
Quiz  
10 Questions for 20 minutes

**Autmata**  
Coding  
2 Questions for 45 minutes

**Automata Fix**  
Coding  
3 Questions for 45 minutes

Note : All the sections are mandatory to attempt then only the submission would be counted

## 8 This is Coding interface , Click on arrow button to change section

### Explanation

To solve this problem, we start by converting the number 45 into its binary form.

The binary representation of 45 is 101101, which uses 6 bits — the minimum required to represent it without padding.

Next, we flip each bit:

1 becomes 0

0 becomes 1

So, flipping 101101 gives us 010010.

Now, we convert the flipped binary 010010 back to its decimal form, which is 18.

Testcases >

Input:

1

Expected Output:


0

Input:

```
8
9 public class Main {
10     public static void main(String[] args) {
11         Scanner sc = new Scanner(System.in);
12         int N = sc.nextInt();
13
14         Solution sol = new Solution();
15         int result = sol.flipBits(N);
16
17         System.out.println(result);
18     }
19 }
20
```

< 1/2 >

## 9 Click "Submit Section" to submit the complete section ( note: individual code submission not accepted , complete writing all codes and then submit )



### Sid and Rid's Subarray Game

You are given an array A of N integers which may include both positive and negative numbers. Two friends, Sid and Rid, decide to play a game: Sid loves positive numbers. He creates a new array S consisting only of the positive numbers from A, preserving their original order. Rid works on the original array A. Each of them does the following: Finds the maximum sum subarray from their respective array. Checks whether their resulting maximum sum subarray contains the majority element (an element that occurs more than floor(N/2) times in the original array A). You are to write a program that: Computes the maximum subarray sum for both Sid and Rid. Returns whether the majority element (if it exists) is present in each of their maximum subarrays.

#### Constraints

- $1 \leq N \leq 10^5$
- Array A may contain both positive and negative integers

#### Sample testcases

Input:

9 1 -2 3 4 -12 1 -5 4

Testcases >

00:44:55

JAVA

Run

Submit Section

```
1 import java.util.*;
2
3 class Solution {
4     public String evaluateGame(int[] A) {
5         int n = A.length;
6         Integer majority = getMajority(A);
7
8         // Rid: max subarray on A
9         Result ridRes = kadane(A);
10        boolean ridHas = majority != null && contains(ridRes.subarr
11
12        // Sid: max subarray on positives
13        List<Integer> posList = new ArrayList<>();
14        for (int x : A) if (x > 0) posList.add(x);
15        int[] S = posList.stream().mapToInt(i -> i).toArray();
16
17        Result sidRes = kadane(S);
18        boolean sidHas = majority != null && contains(sidRes.subarr
19
20        return "Rid: " + ridRes.maxSum + " " + (ridHas ? "Yes" : "No")
21        + "Sid: " + sidRes.maxSum + " " + (sidHas ? "Yes" : "No");
22    }
23
24    private boolean contains(int[] arr, int val) {
25        for (int x : arr) if (x == val) return true;
26        return false;
27    }
28}
```

Made with Scribe - <https://scribehov.com>

5

## 10 Click "Confirm" to submit section

Each of them does the following: Finds the maximum sum subarray from their respective array. Checks whether their resulting maximum sum subarray contains the majority element (an element that occurs more than  $\text{floor}(N/2)$  times in original array A). You are to write a program that finds the maximum subarray sum for both Sid and Rid. Return the majority element (if it exists) is present in either of the maximum subarrays.

**Constraints**

- $1 \leq N \leq 10^5$
- Array A may contain both positive and negative integers.

**Sample testcases**

**Input:**  
9 1 -2 3 4 -1 2 1 -5 4

**Testcases** >

**Input:**  
9 1 -2 3 4 -1 2 1 -5 4

**Expected Output:**  
Rid: 9 No Sid: 15 No

**Input:**

Are you sure you want to submit?

Note : This will submit the whole coding section  
Please make sure you have written code for each question on given editor  
This action is **irreversible**. Once submitted, your changes cannot be undone.

Cancel **Confirm**

```
8 // Rid: max subarray on A
9 Result ridRes = kadane(A);
10
11 // Sid: max subarray on A
12 Result sidRes = kadane(A);
13
14 // Check if the majority element is present in either of the maximum subarrays
15 boolean hasMajorityElement(int arr[], int val) {
16     int count = 0;
17     for (int x : arr) {
18         if (x == val) count++;
19     }
20     return count > arr.length / 2;
21 }
22
23 // Main function
24 public static void main(String[] args) {
25     Scanner scanner = new Scanner(System.in);
26     int n = scanner.nextInt();
27     int[] arr = new int[n];
28     for (int i = 0; i < n; i++) {
29         arr[i] = scanner.nextInt();
30     }
31     scanner.close();
32
33     int ridRes = kadane(arr);
34     int sidRes = kadane(arr);
35     boolean hasMajorityElement = hasMajorityElement(arr, ridRes);
36     boolean hasMajorityElementSid = hasMajorityElement(arr, sidRes);
37     System.out.println("Rid: " + ridRes + " Sid: " + sidRes);
38     if (hasMajorityElement) {
39         System.out.println("Majority element is present in either of the maximum subarrays.");
40     } else {
41         System.out.println("Majority element is not present in either of the maximum subarrays.");
42     }
43 }
```

## 11 You can adjust screen view by dragging dots as shown in screen

$1 \leq N \leq 10^3$   
 $-10^4 \leq A[i] \leq 10^4$

**Sample testcases**

**Input:**  
5 2 8 1 4 7

**Output:**  
8

**Explanation:** 8 is the largest element.

**Input:**  
4 1 2 3 4

**Output:**  
4

Testcases > **Output**

```
8 for (int num : arr) {
9     if (num < max) {
10         max = num;
11     }
12 }
13 return max;
14 }
15
16 public static void main(String[] args) {
17     Scanner scanner = new Scanner(System.in);
18
19     int n = scanner.nextInt();
20     int[] arr = new int[n];
21
22     for (int i = 0; i < n; i++) {
23         arr[i] = scanner.nextInt();
24     }
25
26     int largest = findLargest(arr);
27     System.out.print(largest);
28
29     scanner.close();
30 }
31
32 }
33
34 }
```

12

After final submission your screen would look like this

**Test Submitted**

Your report will be emailed to you shortly.

Thank you for attempting the test!

Close 4





