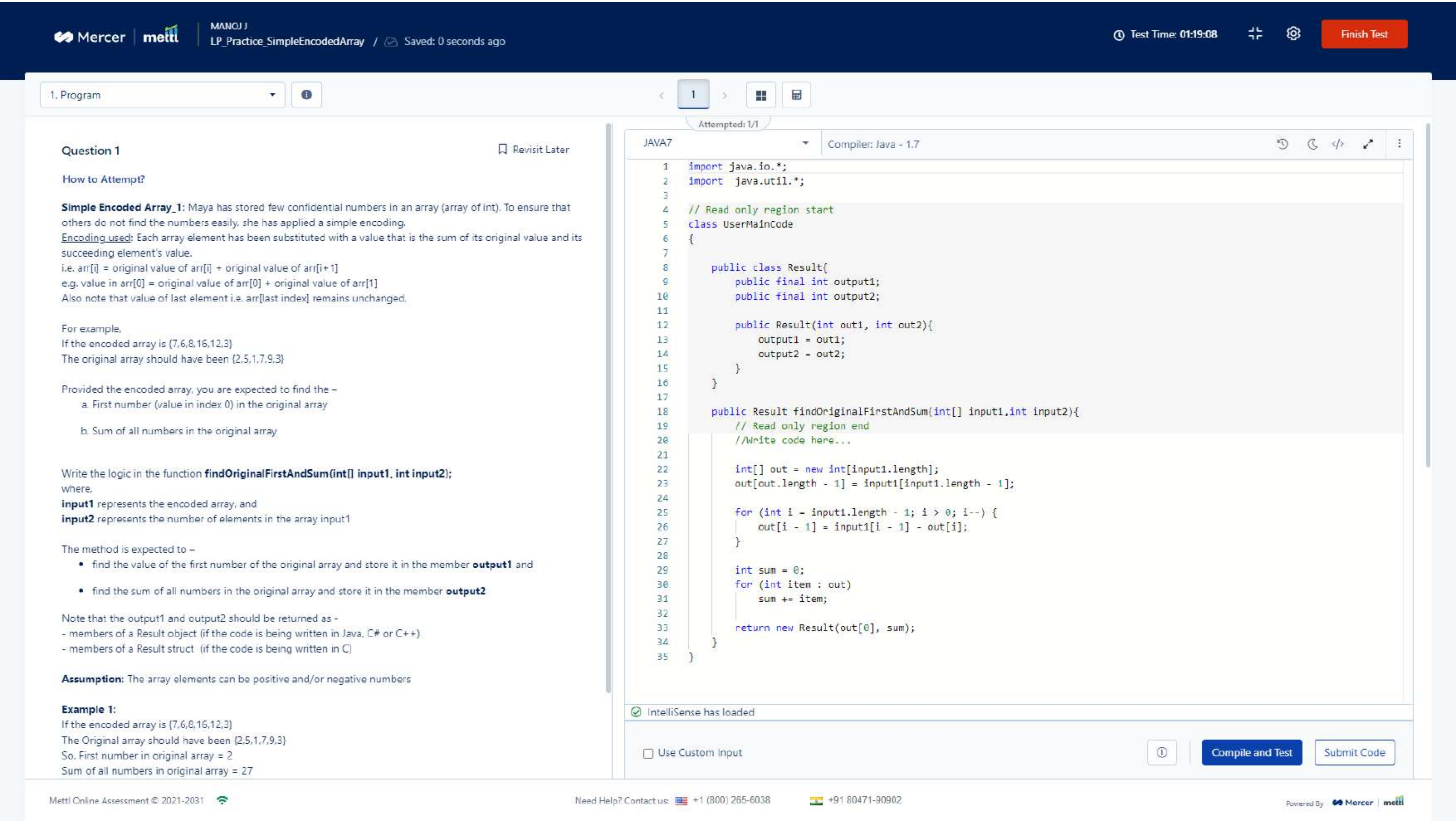


Question 1

How to Attempt?

Simple Encoded Array_1: Maya has stored few confidential numbers in an array (array of int). To ensure that others do not find the numbers easily, she has applied a simple encoding.
Encoding used: Each array element has been substituted with a value that is the sum of its original value and its succeeding element's value.
i.e. arr[i] = original value of arr[i] + original value of arr[i+1]
e.g. value in arr[0] = original value of arr[0] + original value of arr[1]
Also note that value of last element i.e. arr[last index] remains unchanged.

For example,
If the encoded array is {7,6,8,16,12,3}
The original array should have been {2,5,1,7,9,3}

Provided the encoded array, you are expected to find the –

a. First number (value in index 0) in the original array

b. Sum of all numbers in the original array

Write the logic in the function **findOriginalFirstAndSum(int[] input1, int input2);**
where,
input1 represents the encoded array, and
input2 represents the number of elements in the array input1

The method is expected to –

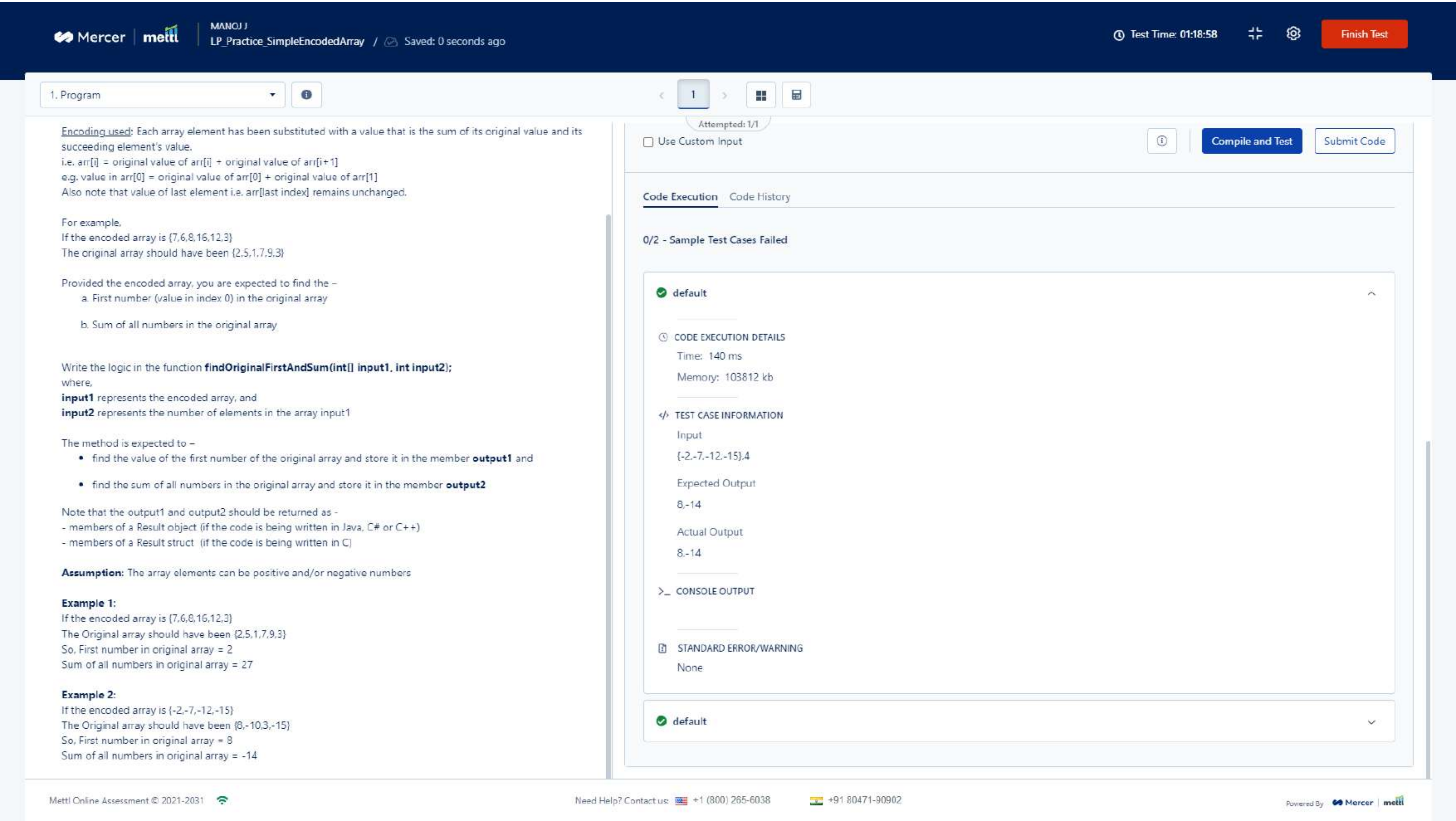
- find the value of the first number of the original array and store it in the member **output1** and
- find the sum of all numbers in the original array and store it in the member **output2**

Note that the output1 and output2 should be returned as -
- members of a Result object (if the code is being written in Java, C# or C++)
- members of a Result struct (if the code is being written in C)

Assumption: The array elements can be positive and/or negative numbers

Example 1:
If the encoded array is {7,6,8,16,12,3}
The Original array should have been {2,5,1,7,9,3}
So, First number in original array = 2
Sum of all numbers in original array = 27

```java
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

    public class Result{
        public final int output1;
        public final int output2;

        public Result(int out1, int out2){
            output1 = out1;
            output2 - out2;
        }
    }

    public Result findOriginalFirstAndSum(int[] input1,int input2){
        // Read only region end
        //Write code here...

        int[] out = new int[input1.length];
        out[out.length - 1] = input1[input1.length - 1];

        for (int i - input1.length - 1; i > 0; i--) {
            out[i - 1] = input1[i - 1] - out[i];
        }

        int sum = 0;
        for (int item : out)
            sum += item;

        return new Result(out[0], sum);
    }
}
```



Encoding used: Each array element has been substituted with a value that is the sum of its original value and its succeeding element's value.
i.e. arr[i] = original value of arr[i] + original value of arr[i+1]
e.g. value in arr[0] = original value of arr[0] + original value of arr[1]
Also note that value of last element i.e. arr[last index] remains unchanged.

For example,
If the encoded array is {7,6,8,16,12,3}
The original array should have been {2,5,1,7,9,3}

Provided the encoded array, you are expected to find the –

a. First number (value in index 0) in the original array

b. Sum of all numbers in the original array

Write the logic in the function **findOriginalFirstAndSum(int[] input1, int input2);**
where,
input1 represents the encoded array, and
input2 represents the number of elements in the array input1

The method is expected to –

- find the value of the first number of the original array and store it in the member **output1** and
- find the sum of all numbers in the original array and store it in the member **output2**

Note that the output1 and output2 should be returned as -
- members of a Result object (if the code is being written in Java, C# or C++)
- members of a Result struct (if the code is being written in C)

Assumption: The array elements can be positive and/or negative numbers

Example 1:
If the encoded array is {7,6,8,16,12,3}
The Original array should have been {2,5,1,7,9,3}
So, First number in original array = 2
Sum of all numbers in original array = 27

Example 2:
If the encoded array is {-2,-7,-12,-15}
The Original array should have been {8,-10,3,-15}
So, First number in original array = 8
Sum of all numbers in original array = -14

0/2 - Sample Test Cases Failed

default

CODE EXECUTION DETAILS
Time: 140 ms
Memory: 103812 kb

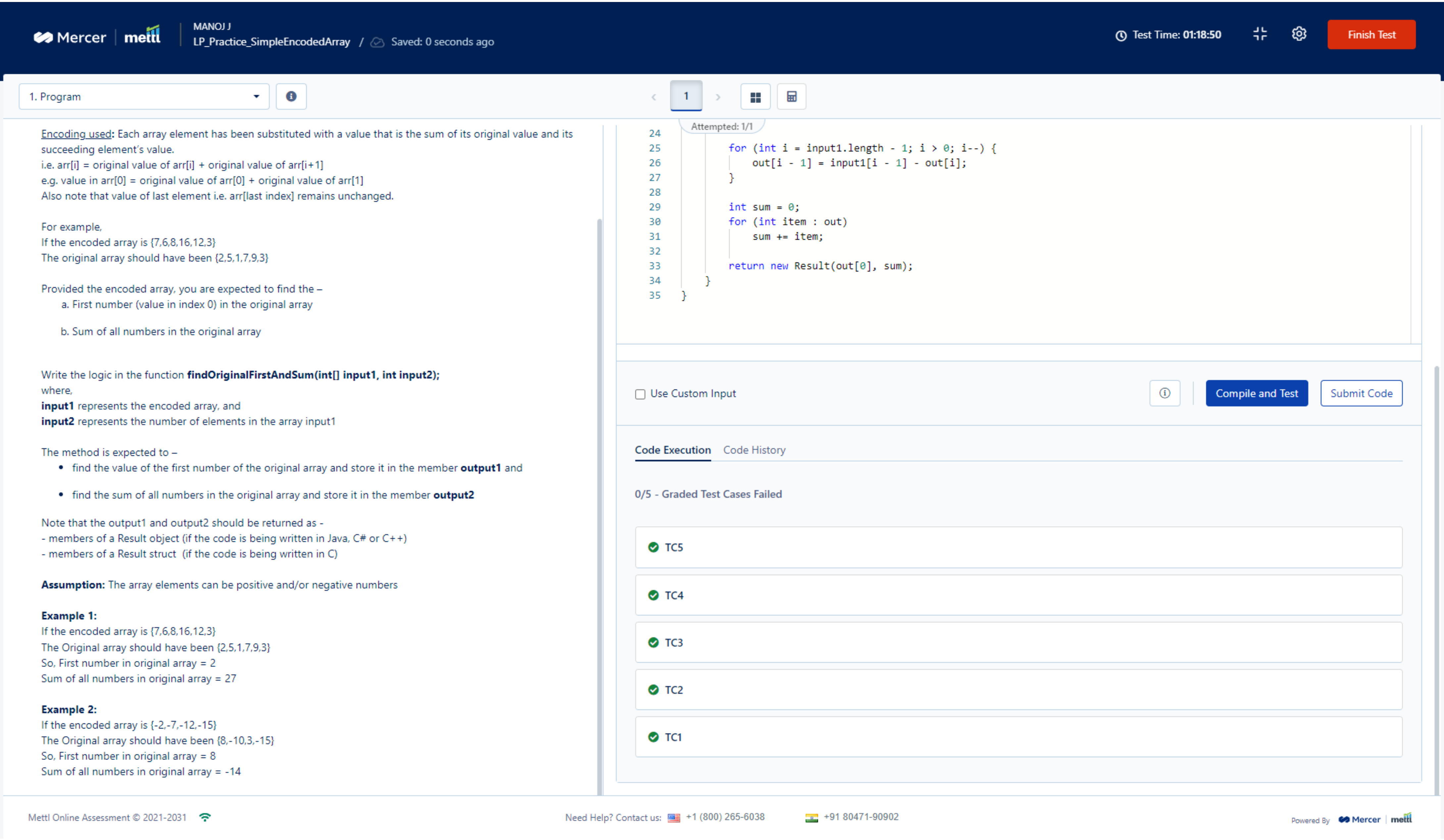
TEST CASE INFORMATION
Input
{-2,-7,-12,-15},4
Expected Output
8,-14
Actual Output
8,-14

CONSOLE OUTPUT

STANDARD ERROR/WARNING
None

default

Mettl Online Assessment © 2021-2031
Need Help? Contact us: +1 (800) 265-6038 +91 80471-90902
Powered By Mercer | mettl



Encoding used: Each array element has been substituted with a value that is the sum of its original value and its succeeding element's value.
i.e. arr[i] = original value of arr[i] + original value of arr[i+1]
e.g. value in arr[0] = original value of arr[0] + original value of arr[1]
Also note that value of last element i.e. arr[last index] remains unchanged.

For example,
If the encoded array is {7,6,8,16,12,3}
The original array should have been {2,5,1,7,9,3}

Provided the encoded array, you are expected to find the –

a. First number (value in index 0) in the original array

b. Sum of all numbers in the original array

Write the logic in the function **findOriginalFirstAndSum(int[] input1, int input2);**
where,
input1 represents the encoded array, and
input2 represents the number of elements in the array input1

The method is expected to –

- find the value of the first number of the original array and store it in the member **output1** and
- find the sum of all numbers in the original array and store it in the member **output2**

Note that the output1 and output2 should be returned as -
- members of a Result object (if the code is being written in Java, C# or C++)
- members of a Result struct (if the code is being written in C)

Assumption: The array elements can be positive and/or negative numbers

Example 1:
If the encoded array is {7,6,8,16,12,3}
The Original array should have been {2,5,1,7,9,3}
So, First number in original array = 2
Sum of all numbers in original array = 27

Example 2:
If the encoded array is {-2,-7,-12,-15}
The Original array should have been {8,-10,3,-15}
So, First number in original array = 8
Sum of all numbers in original array = -14

```
        for (int i = input1.length - 1; i > 0; i--) {
            out[i - 1] = input1[i - 1] - out[i];
        }

        int sum = 0;
        for (int item : out)
            sum += item;

        return new Result(out[0], sum);
    }
}
```

0/5 - Graded Test Cases Failed

✓ TC5

✓ TC4

✓ TC3

✓ TC2

✓ TC1