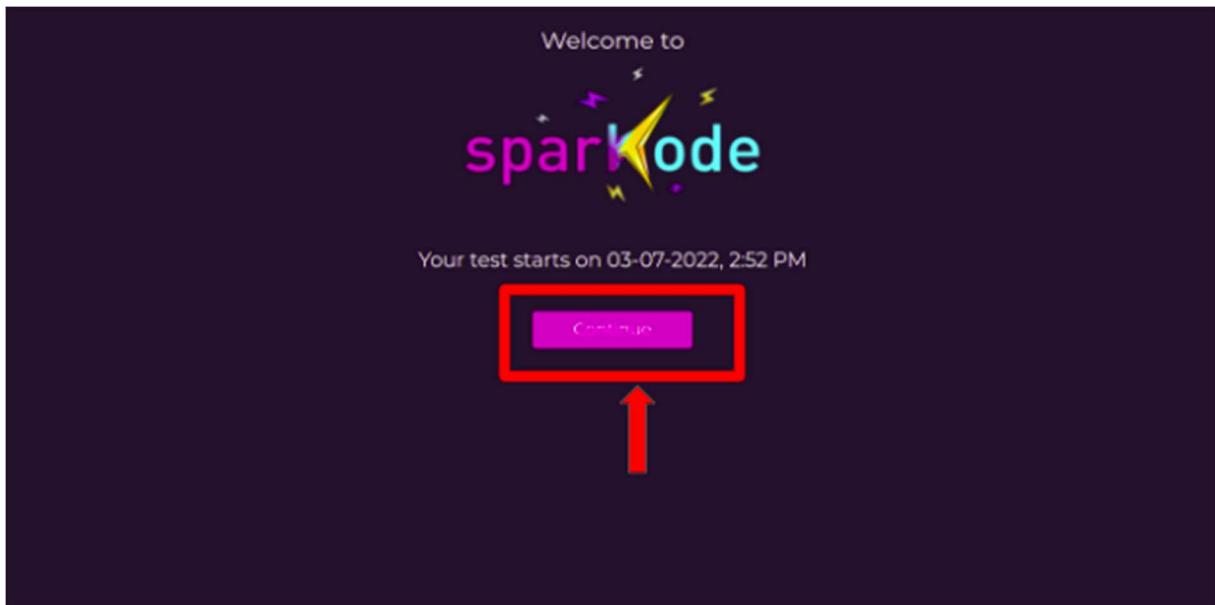


# Instructions for Candidates

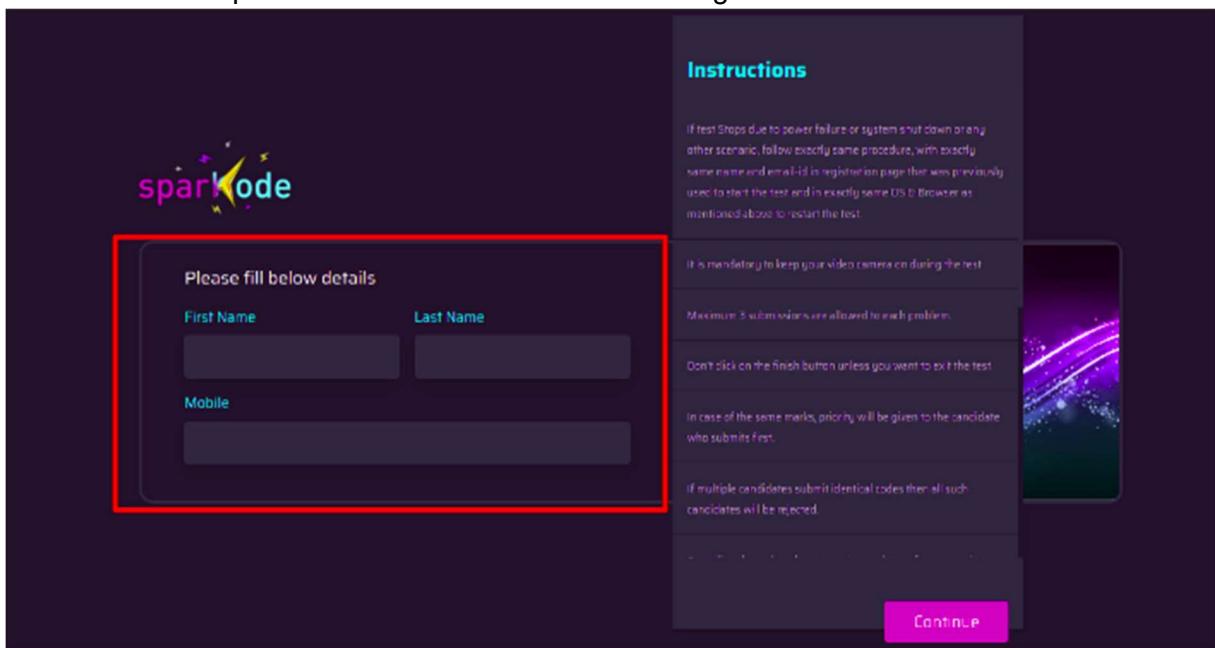
## Step 1:

After clicking the invite link that was sent to the user via email, the screen shown below will appear with the date and time and a button to continue.

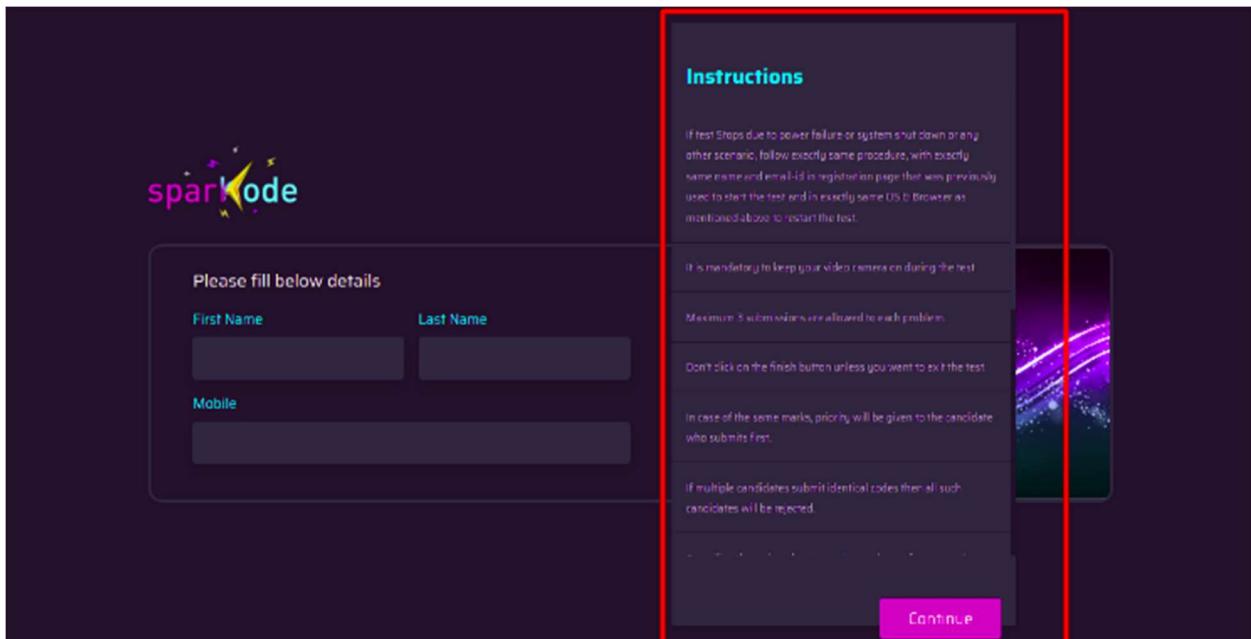


## Step 2:

The test screen opens Click on the Continue button to go to the next screen.

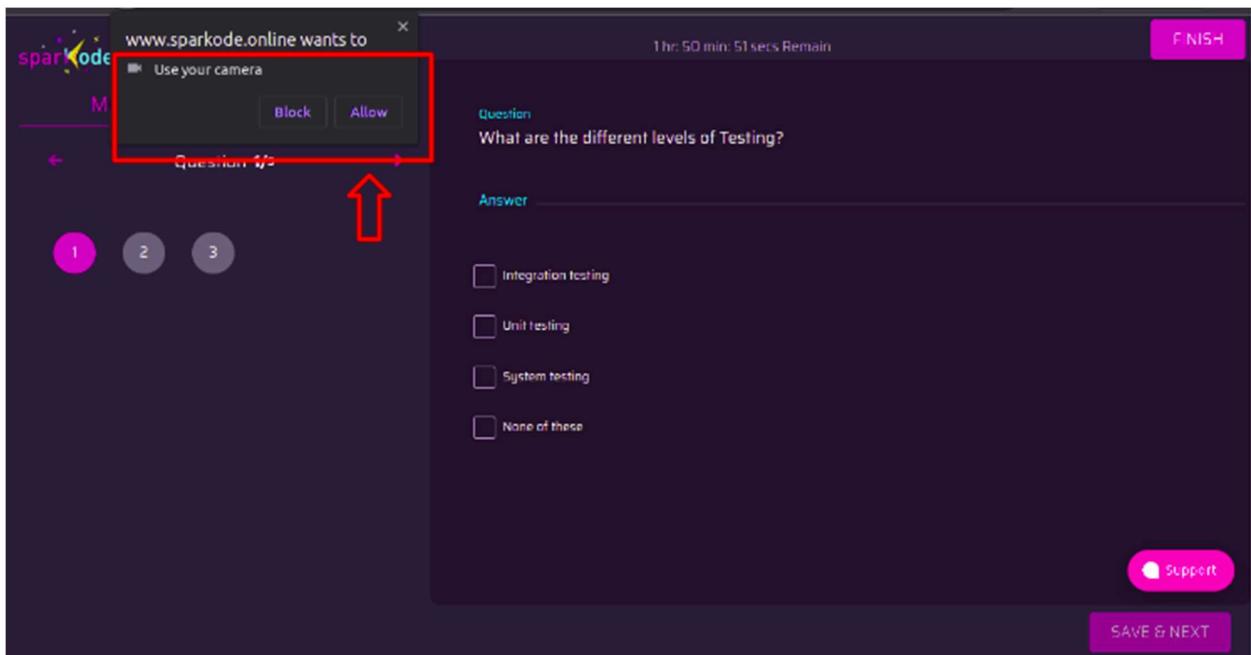


Please fill in the details like first and last name and mobile number. The candidate should carefully read the scrollable instructions. Click Continue.



### Step 3:

To grant access to the camera—which is necessary while giving the test—click Allow.



The highlighted region displays the remaining test time.

A screenshot of the Sparkode platform interface. On the left, there's a sidebar with the Sparkode logo and navigation links for 'MCQs' and 'Problems'. The main area shows 'Problem 1/3' titled '[DO NOT USE]- TEST- TO LOWER CASE'. It contains code examples and constraints. A red arrow points to a box containing '1 hr: 19 min: 35 secs Remain' at the top right of the screen. At the bottom right are buttons for 'Support', '3 Submissions Left', 'Add Custom Input', 'SAVE', 'RUN', and 'SUBMIT'.

## Step 4:

Click on the Start Conversation button Enter the Email, Message and click on the Start Conversation button.

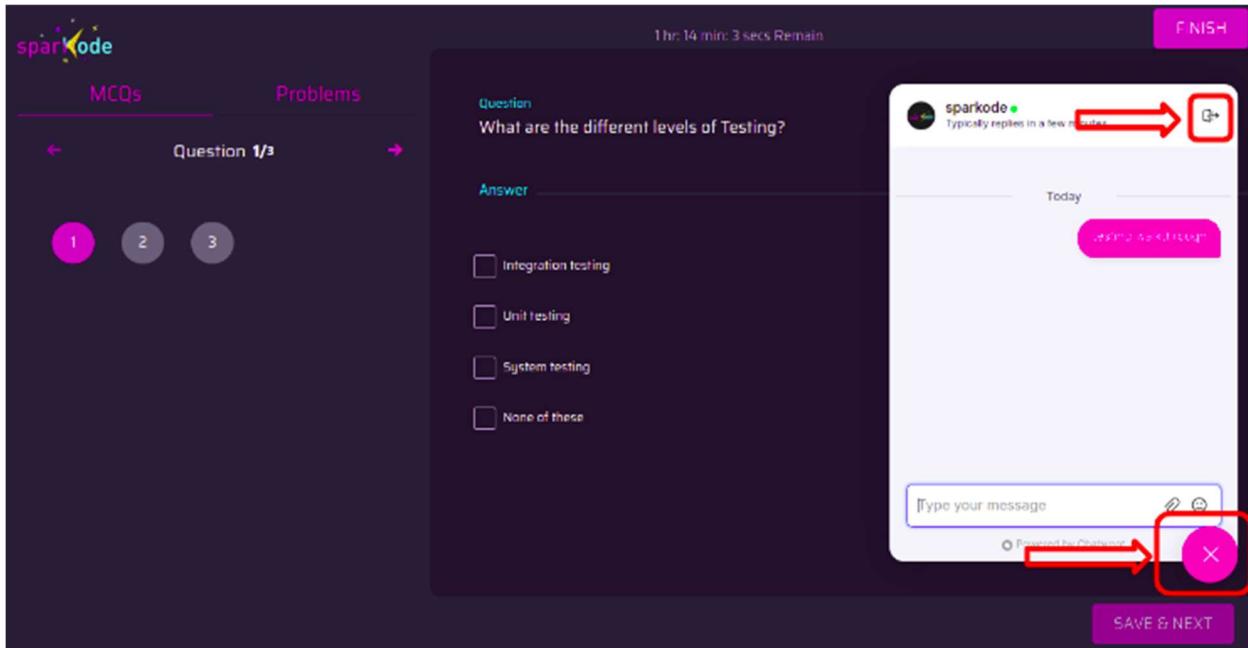
A screenshot of the Sparkode platform interface. On the left, there's a sidebar with the Sparkode logo and navigation links for 'MCQs' and 'Problems'. The main area shows 'Question 1/3' with the question 'What are the different levels of Testing?'. Below it is an 'Answer' section with four options: 'Integration testing' (selected), 'Unit testing', 'System testing', and 'None of these'. To the right is a 'Welcome' message box with a red arrow pointing to a 'Start Conversation' button. At the bottom right are buttons for 'SAVE & NEXT'.

The screenshot shows a dark-themed user interface for a learning platform. At the top right are 'FINISH' and 'SAVE & NEXT' buttons. In the center, a question is displayed: 'What are the different levels of Testing?'. Below it is an 'Answer' section with four options: 'Integration testing', 'Unit testing', 'System testing', and 'None of these', each preceded by an empty checkbox. To the right of the question is a support chat window from 'sparkode'. The chat header says 'We are away at the moment.' It contains three input fields: 'Email Address' (containing 'Please enter your email address'), 'Message' (containing 'Please enter your message'), and a 'Start Conversation' button. Red arrows point from the right side of the question area to each of these three fields. At the bottom of the support window is a note 'Powered by Chatwoot' and a close 'X' icon.

Click on the 'X' icon to close the support chat window.

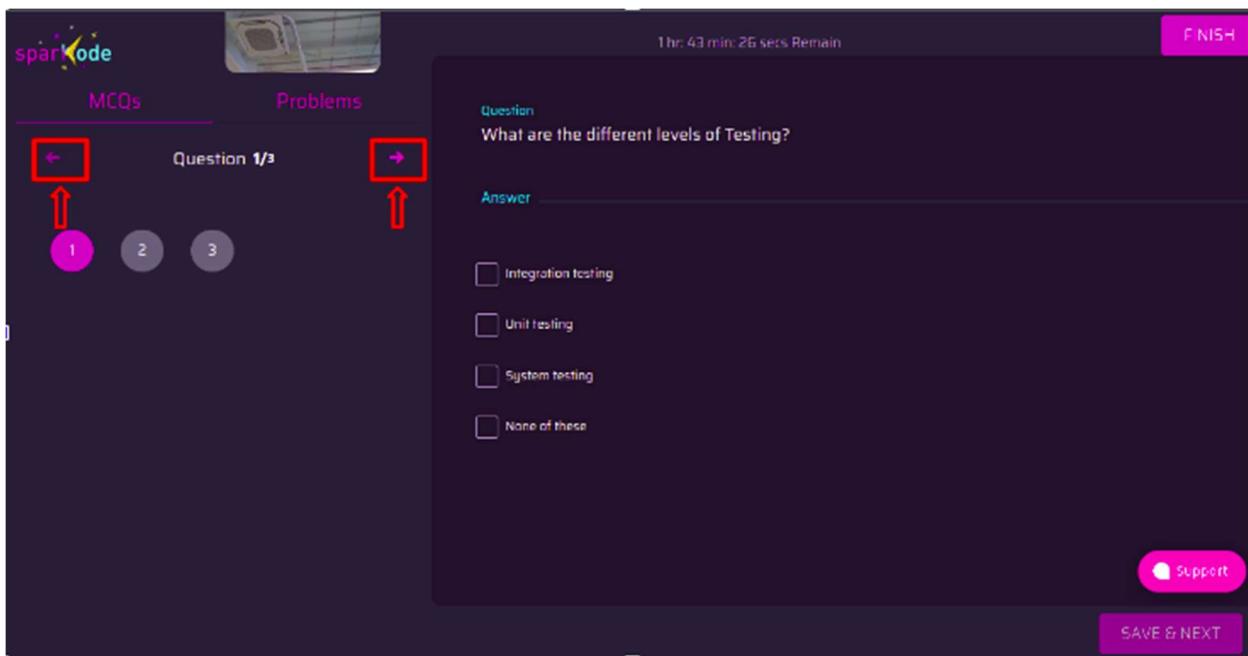
This screenshot shows the same learning platform interface after the support chat window has been closed. The support window is now empty, showing a timestamp 'Today' and a message placeholder 'Type your message'. A red arrow points to the close 'X' icon at the bottom right of the support window. The rest of the interface remains the same, including the question, answer options, and navigation buttons.

Click on the top right corner of the chat window to end the conversation.



## Step 5:

Click on the arrows shown to navigate between the Questions.



In the part where you provide your answers, check the appropriate boxes. To initiate a chat with the support staff in the event of any problems, click the Support button.

The screenshot shows a user interface for a quiz or test. At the top, there's a navigation bar with the 'sparkode' logo, a camera icon, and a timer indicating '1 hr: 35 min: 54 secs Remain'. On the right side of the bar are 'FINISH' and 'Support' buttons. Below the bar, there are tabs for 'MCQs' and 'Problems', with 'MCQs' being the active tab. The main area displays a question: 'What are the different levels of Testing?'. Below the question is a list of four options with checkboxes:

- Integration testing
- Unit testing
- System testing
- None of these

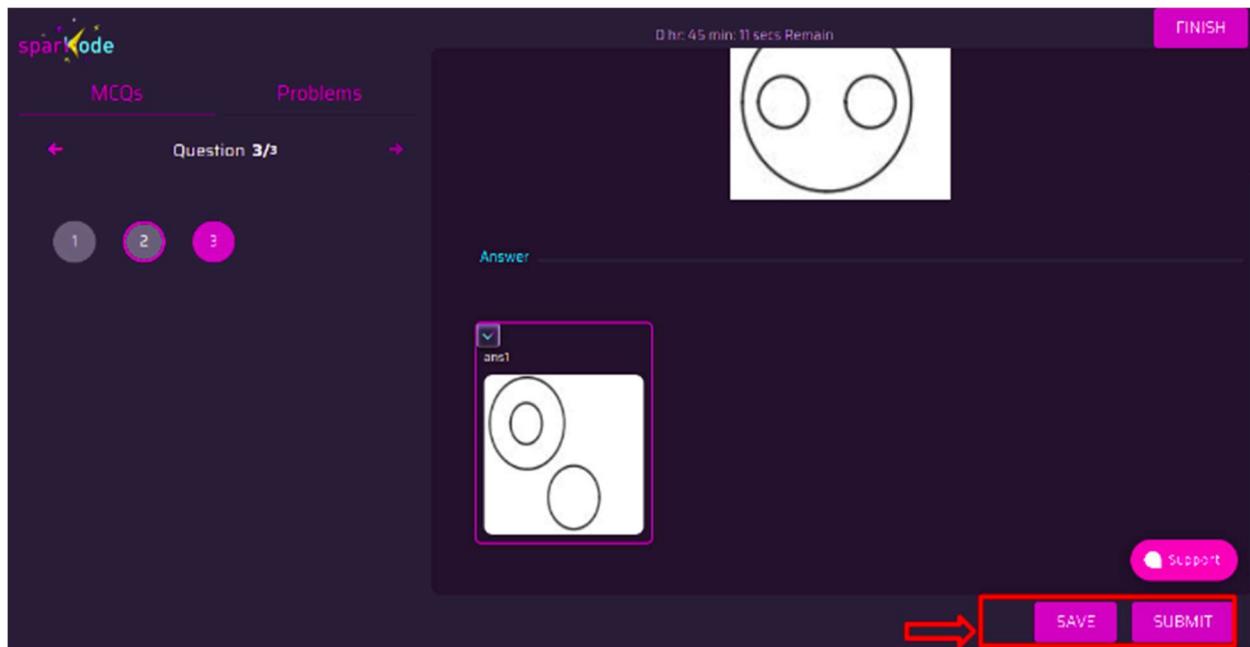
A red rectangular box highlights the first two options ('Integration testing' and 'Unit testing'). A red arrow points from the bottom right towards the 'Support' button, which is also highlighted with a red box. At the very bottom right of the screen, there's a 'SAVE & NEXT' button.

## Step 6:

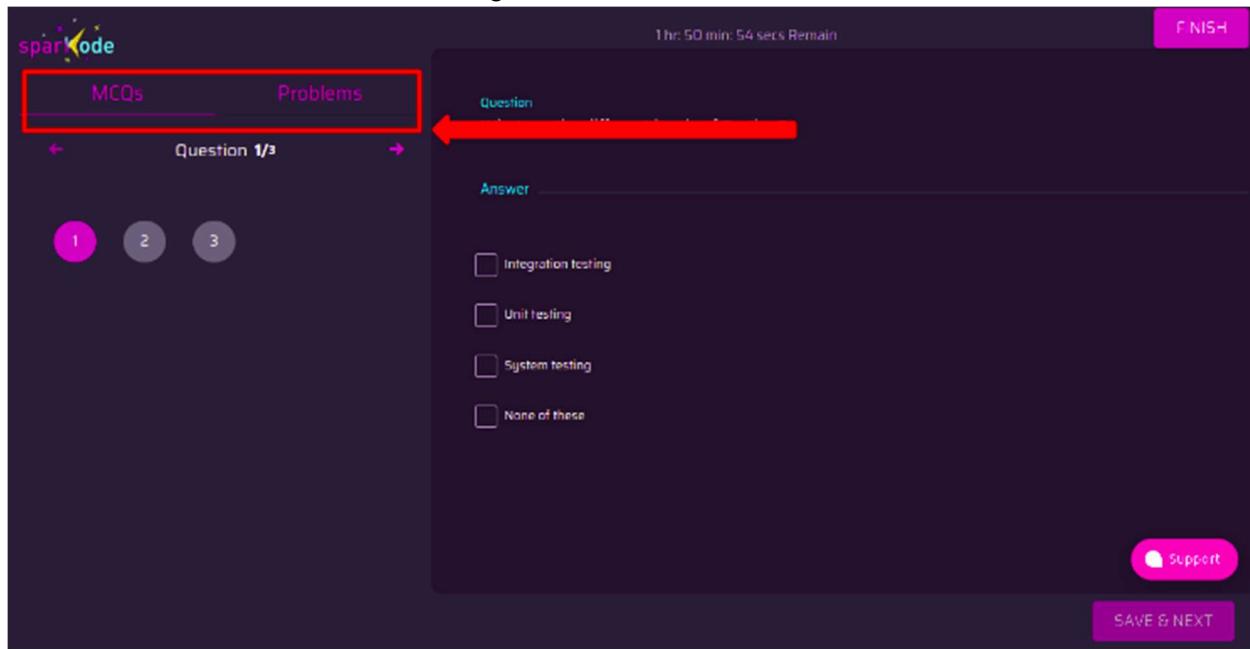
To save the selected response and move on to the following question, click the Save & Next option.

This screenshot shows the same interface as the previous one, but the 'Support' button has been clicked, as indicated by a red arrow pointing to it. The 'Support' button is now grayed out, and a new button labeled 'SAVE & NEXT' has appeared at the bottom right. The rest of the interface remains the same, including the question, answer options, and the 'SAVE & NEXT' button at the bottom right.

Click on the Save and then Click on the Submit button to submit MCQs.

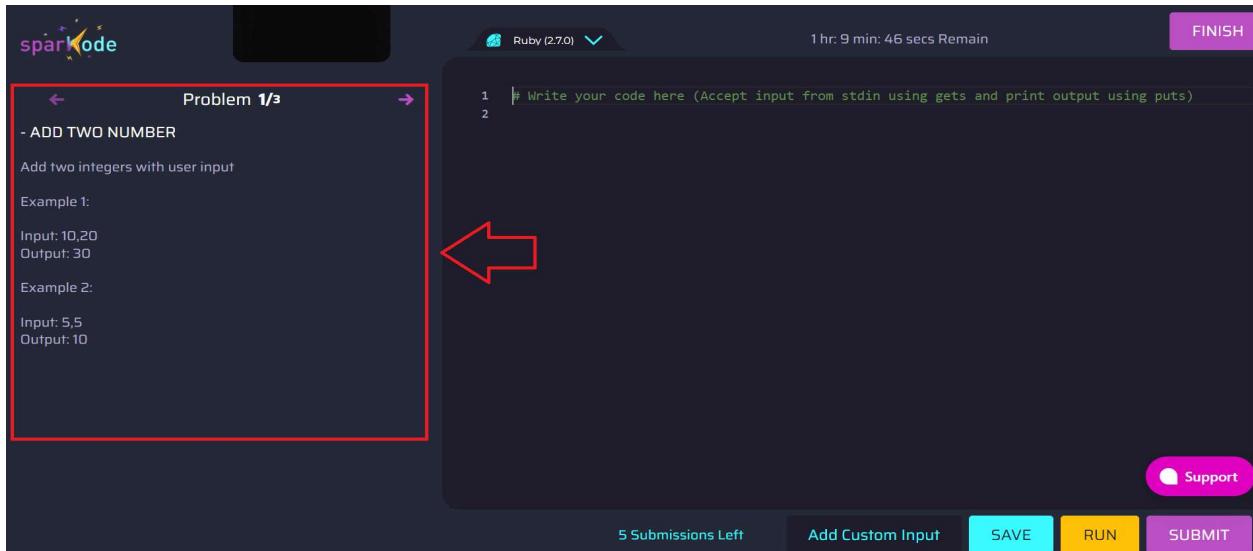


To switch between the Question categories, candidates can click on MCQs and Problems.



## Step 7:

The problem statement is displayed in the portion that is highlighted.

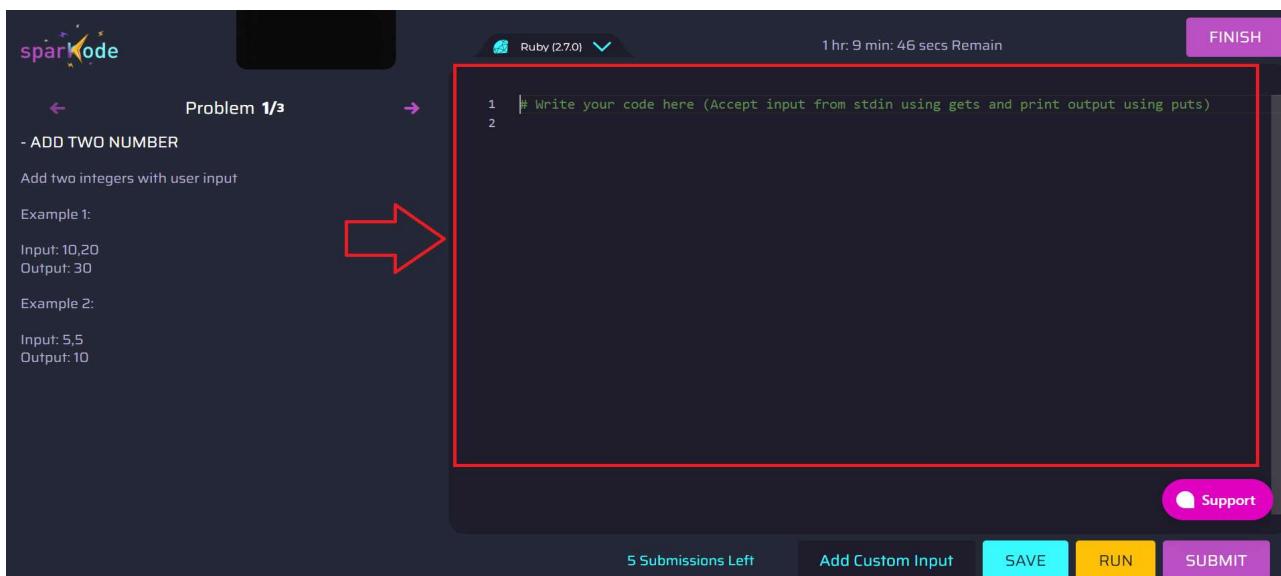


A screenshot of the sparkode platform interface. On the left, there's a sidebar with the sparkode logo and navigation links. The main area shows a problem titled "Problem 1/3" with the sub-task "- ADD TWO NUMBER". It asks to "Add two integers with user input" and provides two examples: Example 1 (Input: 10,20, Output: 30) and Example 2 (Input: 5,5, Output: 10). A red box highlights the problem statement and examples. To the right, there's a code editor window with the following code:

```
1  # Write your code here (Accept input from stdin using gets and print output using puts)
2
```

Below the code editor are buttons for "Support", "5 Submissions Left", "Add Custom Input", "SAVE" (in blue), "RUN" (in yellow), and "SUBMIT" (in purple).

The Editor in which the candidate can write his code is displayed in the highlighted session.



A screenshot of the sparkode platform interface, similar to the previous one but with a different focus. A red box highlights the code editor area where the candidate would write their code. To the left of the code editor, a red arrow points towards it from the left side of the screen. The rest of the interface is identical to the first screenshot, including the sidebar, problem statement, and submission buttons.

## Step 8:

To change the writing language, click on the language template.

The screenshot shows a programming challenge interface. At the top right, there is a language dropdown menu set to "Ruby (2.7.0)". A red box highlights this dropdown. Below it, the code editor contains the following code:

```
1  # Write your code here (Accept input from stdin using gets and print output using puts)
2
```

The challenge title is "Problem 1/3" and the subtitle is "- ADD TWO NUMBER". The instructions say "Add two integers with user input". Examples are provided: Example 1 (Input: 10,20, Output: 30) and Example 2 (Input: 5,5, Output: 10). At the bottom, there are buttons for "Support", "5 Submissions Left", "Add Custom Input", "SAVE", "RUN", and "SUBMIT".

From the languages list drop-down, choose the necessary language template.

This screenshot shows the same challenge interface, but the language dropdown menu is now expanded, revealing a list of available templates. A red box highlights this expanded list. The options include:

- Ruby (2.7.0)
- C (GCC 8.3.0)
- C++ (GCC 9.2.0)
- Java (OpenJDK 13.0.1)
- Python (3.8.1)
- JavaScript (Node.js 12.14.0)
- Go (1.13.5)

The rest of the interface remains the same, with the challenge title, subtitle, instructions, examples, and submission buttons visible at the bottom.

Click on the RUN button after entering the code to check the Output.

The screenshot shows a dark-themed interface for a coding challenge. At the top, there's a logo for "sparkode" and some navigation icons. Below that, the title "Problem 1/3" and the subtitle "- ADD TWO NUMBER". The task description says "Add two integers with user input". It includes two examples: Example 1 (Input: 10,20, Output: 30) and Example 2 (Input: 5,5, Output: 10). The code area contains a C++ code template:

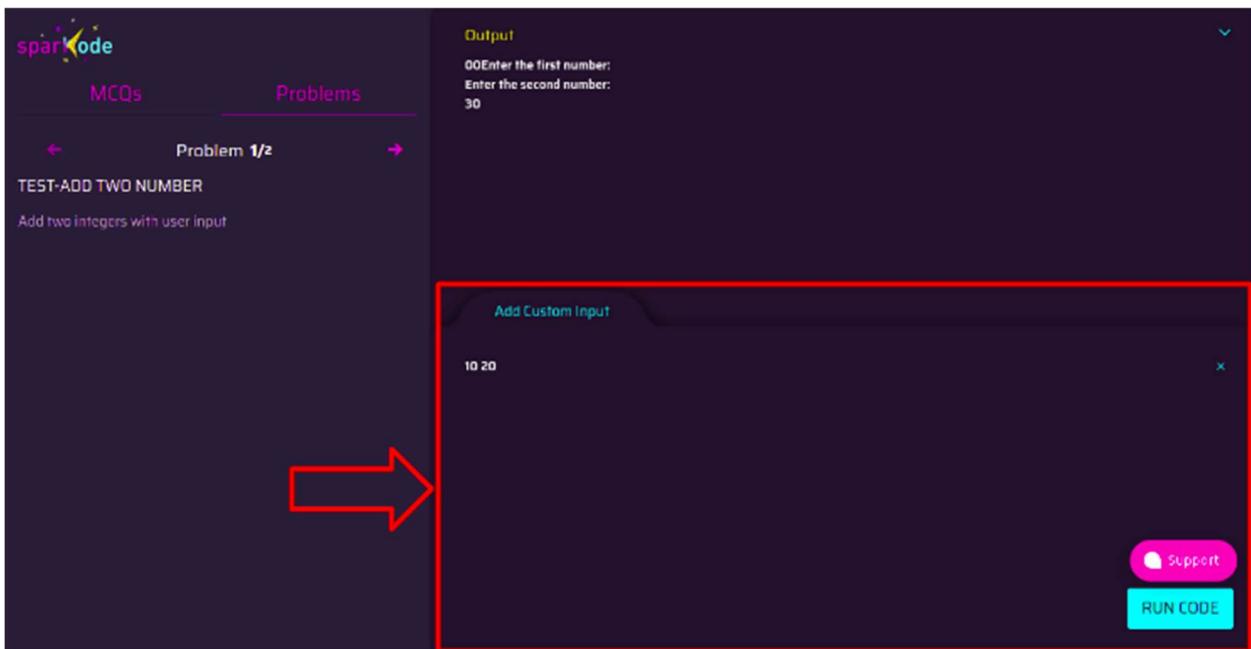
```
1 #include<bits/stdc++.h>
2 using namespace std;
3
4 int main() {
5     // your code goes here
6     int num1, num2;
7
8     cout<<num1;
9     cout<<num2;
10    cout<<"Enter the first number: \n";
11    cin>>num1;
12    cout<<"Enter the second number: \n";
13    cin>>num2;
14    cout<<{num1 + num2};
15 }
16
```

At the bottom, there are buttons for "SAVE", "RUN" (which is highlighted with a red border and a red arrow pointing to it), and "SUBMIT". There are also "Support" and "Add Custom Input" buttons. The status bar at the bottom shows "5 Submissions Left" and "0 hr: 43 min: 36 secs Remain".

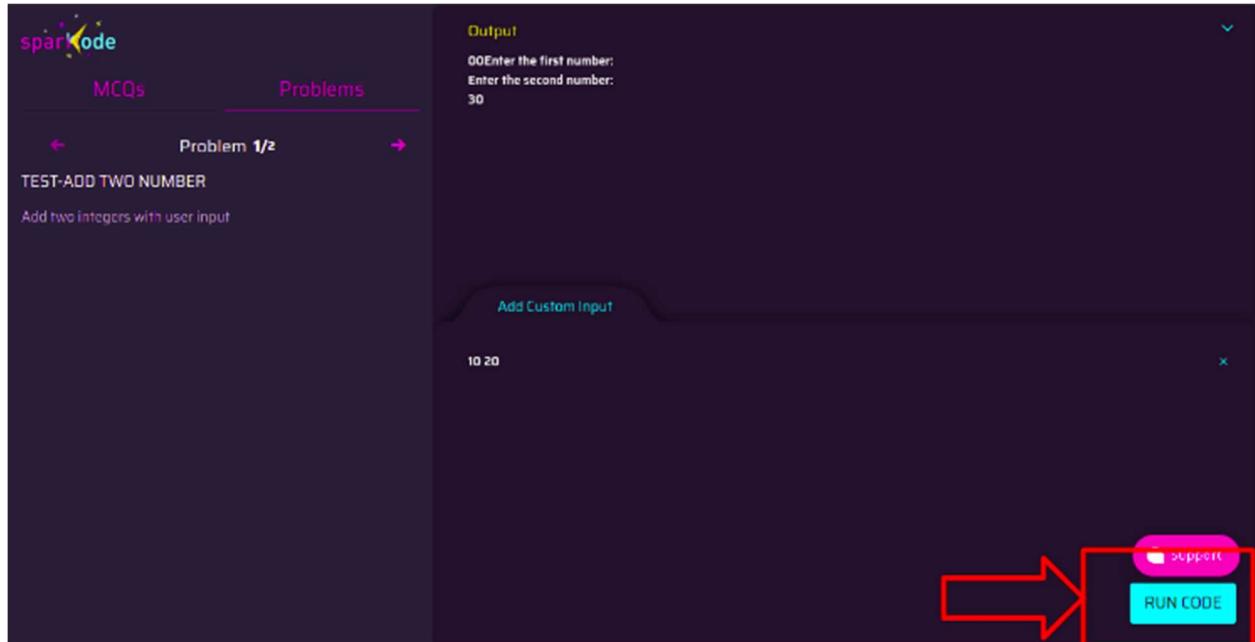
Click on the Add Custom Input to enter the custom input.

This screenshot shows the same interface as the previous one, but with a red arrow pointing to the "Add Custom Input" button at the bottom left. The rest of the interface is identical, including the code template and the "RUN" button highlighted with a red border.

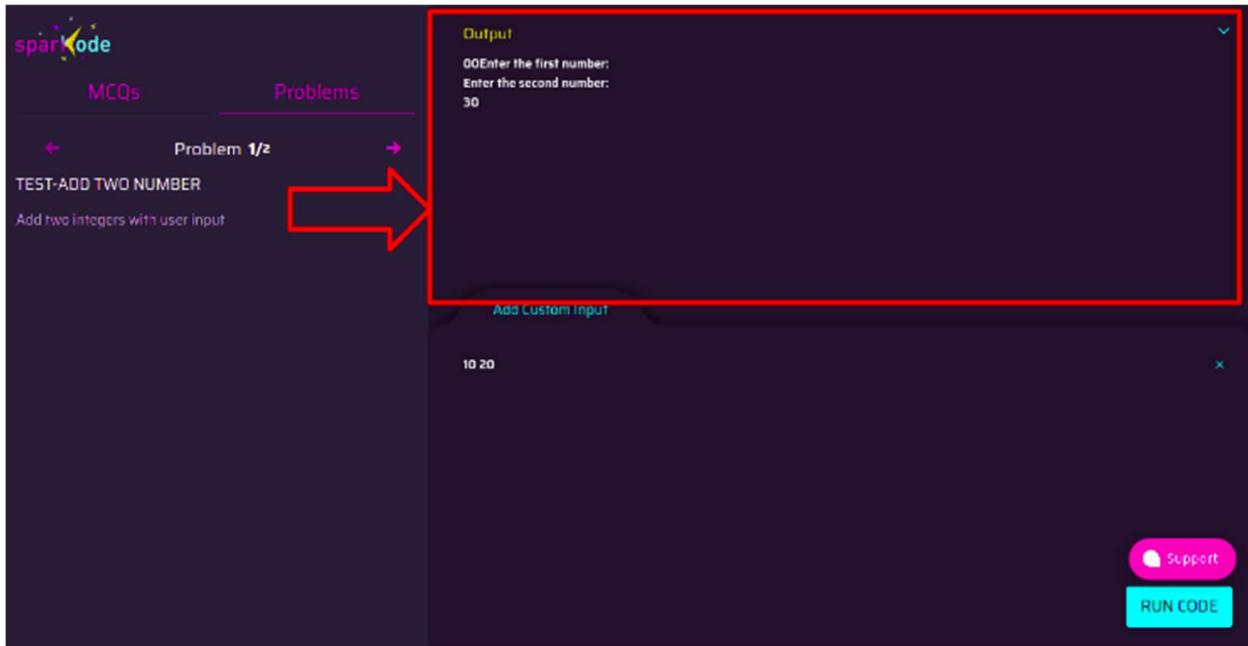
Enter Input in the highlighted field.



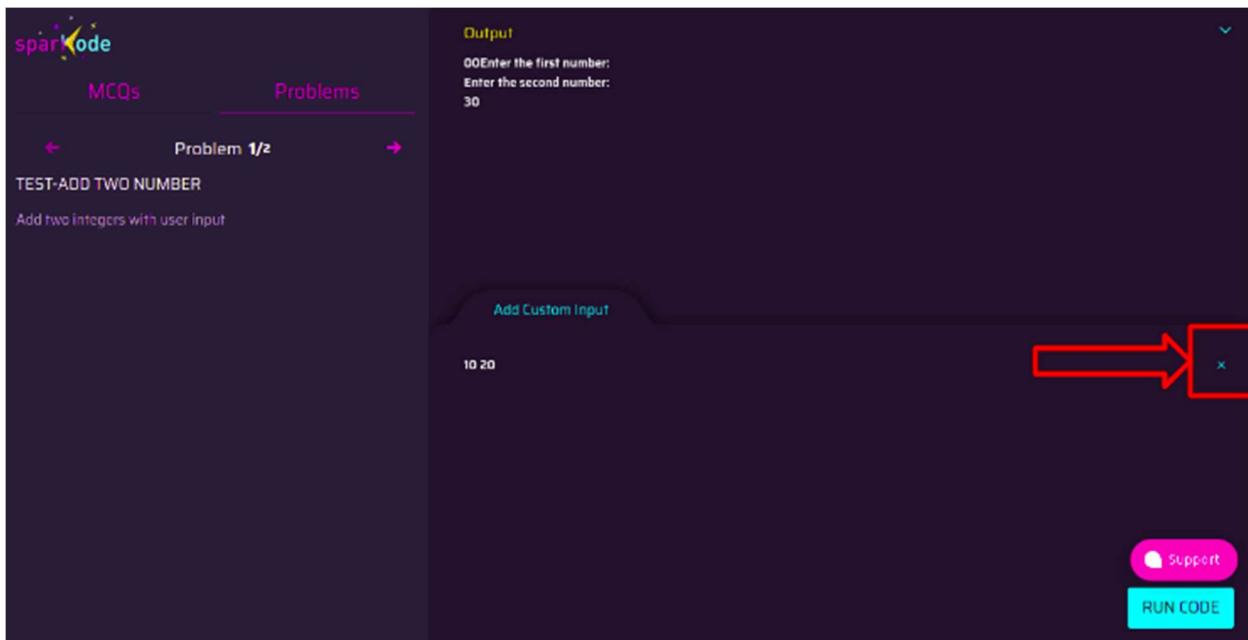
Click on the Run Code to check the Output.



Output can be shown in the highlighted section.



Click on the 'x' icon to close the custom input window.



## Step 9:

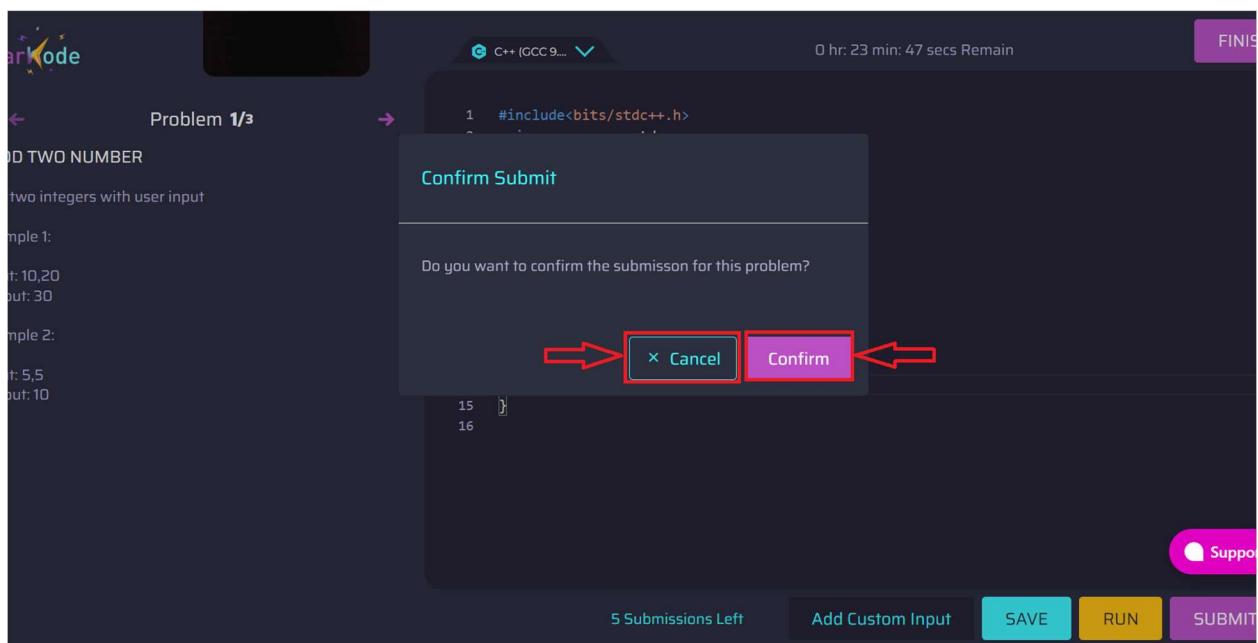
Click on the Submit button to submit the code and run the test cases.

The screenshot shows a dark-themed interface for a coding challenge. At the top right, there's a purple 'FINISH' button. Below it, a status bar shows '0 hr: 43 min: 36 secs Remain'. In the center, there's a code editor window titled 'C++ (GCC 9...)' with a dropdown menu. The code itself is:

```
1 #include<bits/stdc++.h>
2 using namespace std;
3
4 int main() {
5     // your code goes here
6     int num1, num2;
7
8     cout<<num1;
9     cout<<num2;
10    cout<<"Enter the first number: \n";
11    cin>>num1;
12    cout<<"Enter the second number: \n";
13    cin>>num2;
14    cout<<(num1 + num2);
15 }
16
```

To the left of the code editor, there's a sidebar with sections for 'Problem 1/3', '- ADD TWO NUMBER', 'Example 1:', and 'Example 2:'. Below these examples are their respective input and output values. At the bottom of the screen, there are several buttons: '5 Submissions Left', 'Add Custom Input', 'SAVE' (in light blue), 'RUN' (in yellow), and 'SUBMIT' (in purple). A large red arrow points directly at the 'SUBMIT' button.

Click on the Confirm button for submission of the code or click on the Cancel button to cancel the submission.



## Step 10:

After clicking the confirm button a popup will appear showing the submission count, Total test cases number, Passed Test Cases with count and Status as Accepted or Failed. When all the test cases fail then the status will be shown as below.

The screenshot shows a programming challenge interface. The problem title is "Problem 1/3" and the subproblem is "[DO NOT USE]- TEST- TO LOWER CASE". The code editor contains the following Ruby code:

```
1 # Write your code here (Accept input from stdin using gets and print output using puts)
2 print ("Hello")
```

The "Final Output" window displays the following information:

- Submission Left: 2
- Total Test Cases: 3
- Passed Test Cases: 0

A red arrow points to a message in a red-bordered box: "Code is submitted successfully. But all test cases are not passed."

At the bottom right of the main interface are buttons for "Support", "SAVE", "RUN", and "SUBMIT".

When some of the test cases pass then the status will be shown as below.

The screenshot shows the same programming challenge interface. The "Final Output" window now displays:

- Submission Left: 2
- Total Test Cases: 3
- Passed Test Cases: 1

A red arrow points to a message in a red-bordered box: "Partially Accepted."

At the bottom right of the main interface are buttons for "Support", "SAVE", "RUN", and "SUBMIT".

When all the test cases pass then the status will be shown as below.

The screenshot shows a programming challenge titled "Problem 1/3" on the sparkode platform. The challenge description is "[DO NOT USE]- TEST- TO LOWER CASE". It asks to return a string s that has had every uppercase letter swapped over for the corresponding lowercase letter. Examples are provided for input "Hello" and output "hello", and for input "here" and output "here". Constraints state that the input length is between 1 and 100, and it consists of printable ASCII characters. The code editor shows the following Ruby code:

```
1 # Write your code here (Accept input from stdin using gets and print output using puts)
2 print ["Hello"]
3
```

The "Final Output" section displays the results of the submission. It shows "Submission Left: 2", "Total Test Cases: 3", and "Passed Test Cases: 3". A green box highlights the message "Code is submitted successfully. All test cases are passed." A red arrow points to this message. At the bottom, there are buttons for "2 Submissions Left", "Add Custom Input", "SAVE" (highlighted with a red box), "RUN", and "SUBMIT".

Click on the Save Button to Save the written Code.

The screenshot shows the same programming challenge setup as the previous one. The code editor contains the following Ruby code:

```
1 # Write your code here (Accept input from stdin using gets and print output using puts)
2 a = 10
3 b = 20
4 puts a + b
5
```

At the bottom of the interface, there are four buttons: "2 Submissions Left", "Add Custom Input", "SAVE" (highlighted with a red box and a red arrow pointing to it), "RUN", and "SUBMIT".

To complete the test, click the Finish button.

The screenshot shows a coding environment on the Sparkode platform. At the top right, there is a purple 'FINISH' button with a red arrow pointing to it. The code editor contains the following Ruby code:

```
1 # Write your code here (Accept input from stdin using gets and print output using puts)
2 a = 10
3 b = 20
4 puts a + b
5
```

Below the code editor, there are several buttons: '2 Submissions Left', 'Add Custom Input', 'SAVE' (in blue), 'RUN' (in yellow), and 'SUBMIT' (in purple). On the far right, there is a 'Support' button with a gear icon.

Click on the Finish button to complete the test or click on the Cancel button to cancel the Finish action.

The screenshot shows a modal dialog box with the following text:

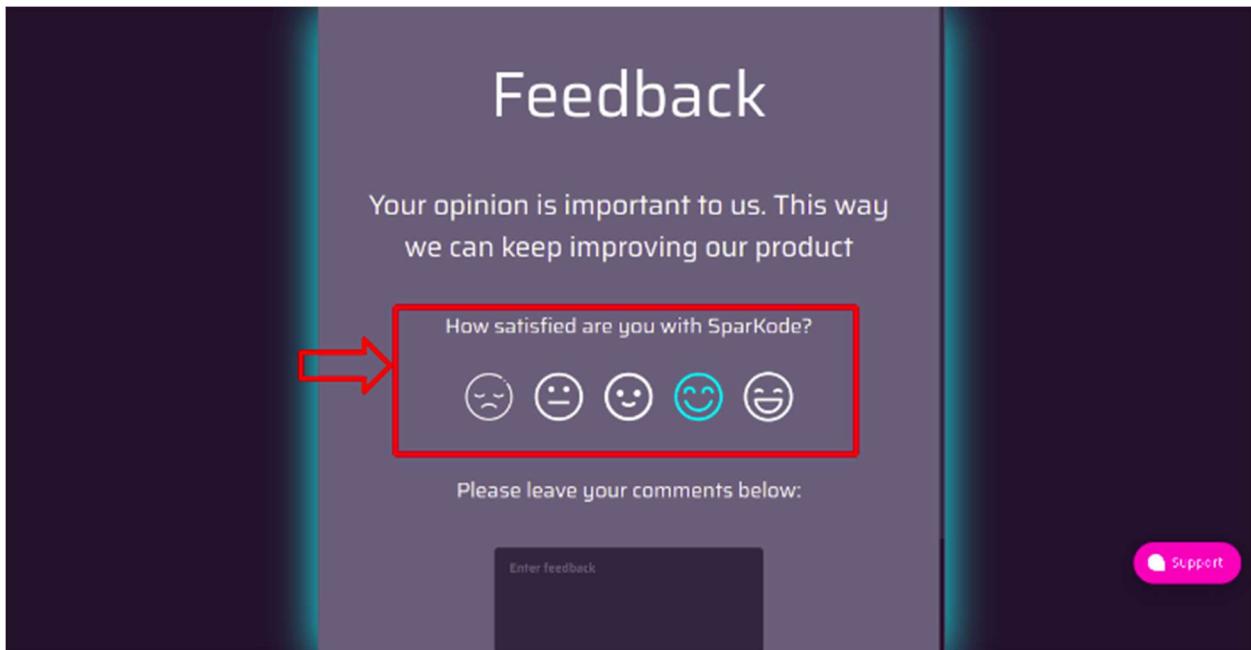
Do you want to Finish the test?  
Once you click on Finish you won't have access for the test.  
You will logout from test.

At the bottom of the dialog, there are two buttons: 'Cancel' (with a red arrow pointing to it) and 'Finish' (also with a red arrow pointing to it).

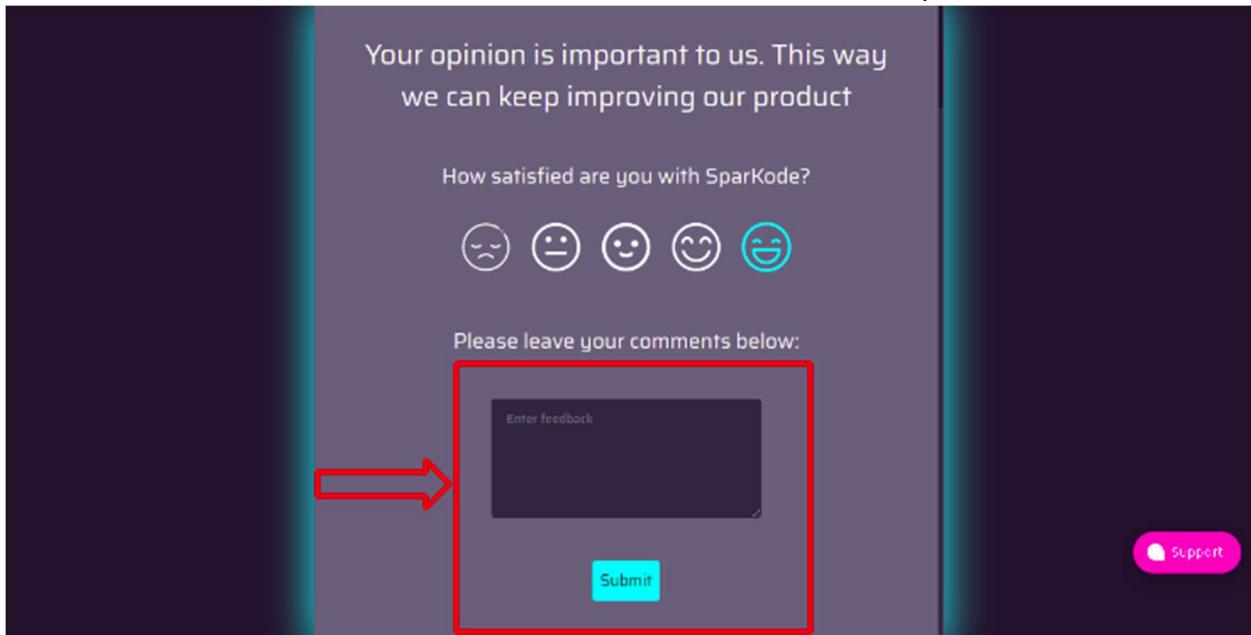
Below the dialog, the main interface shows the same problem details and submission buttons as the first screenshot.

## Step 11:

After Clicking on the Finish button candidates will be redirected to the Feedback Page where the Candidate should select the Feedback emoticon rating from the 1-5 score range.



Enter the Feedback in the text field and Click the Submit button to complete the test.



Once the feedback is submitted then the Test completion page will be displayed.

