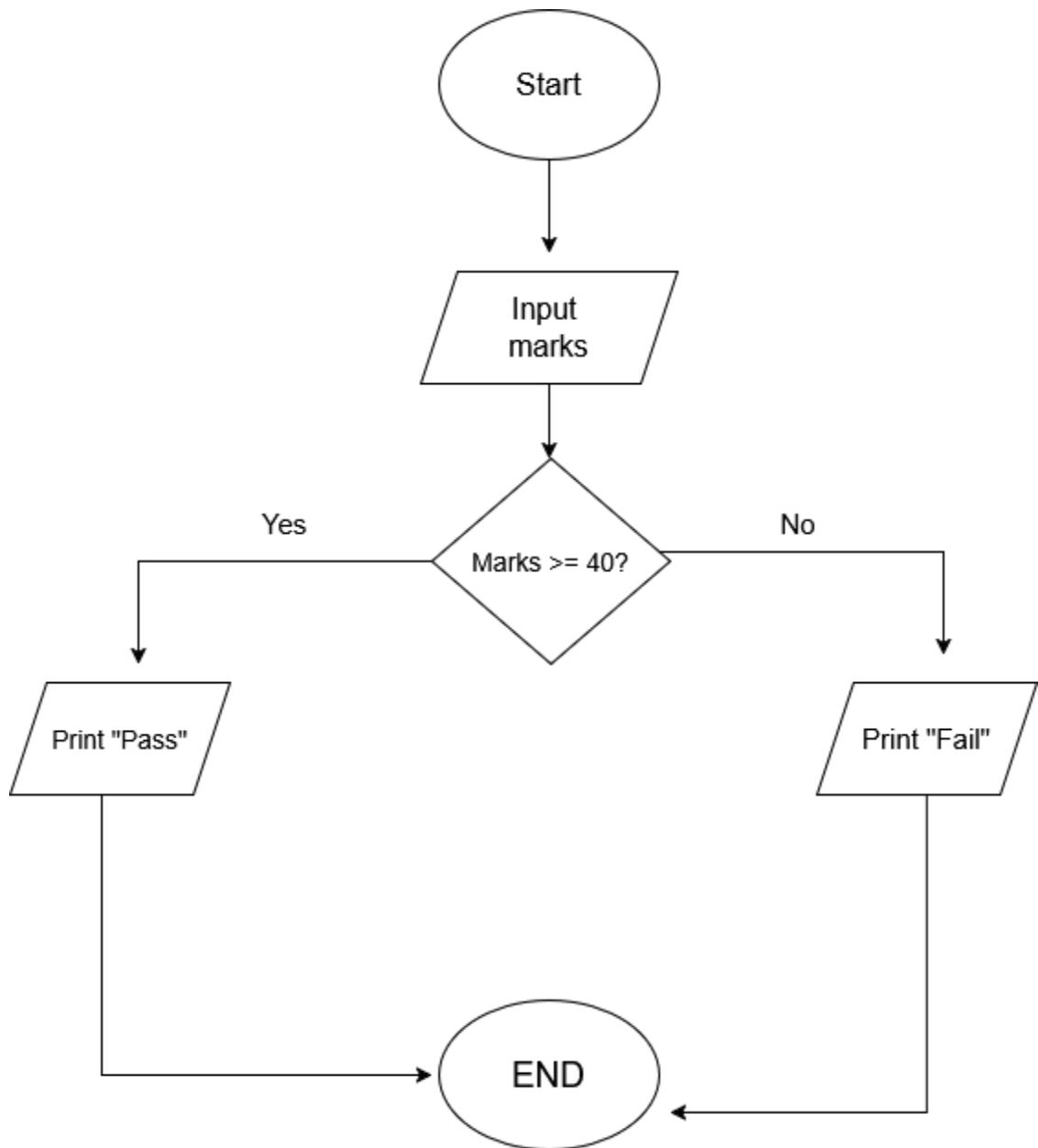


### Experiment 1.5: Student pass or fail

Algorithm:

```
Step 1: START
Step 2: INPUT marks (as integer)
Step 3: CHECK if marks >= 40
Step 4:   IF TRUE, then OUTPUT "Pass"
Step 5:   IF FALSE, then OUTPUT "Fail"
Step 6: STOP
```

Flowchart:



### 1.1.5. Student Pass or Fail Status

01:18 A C ☰ -

Write a Python program to determine whether a student passed the exam or not based on their marks.

#### Pass/Fail Criteria:

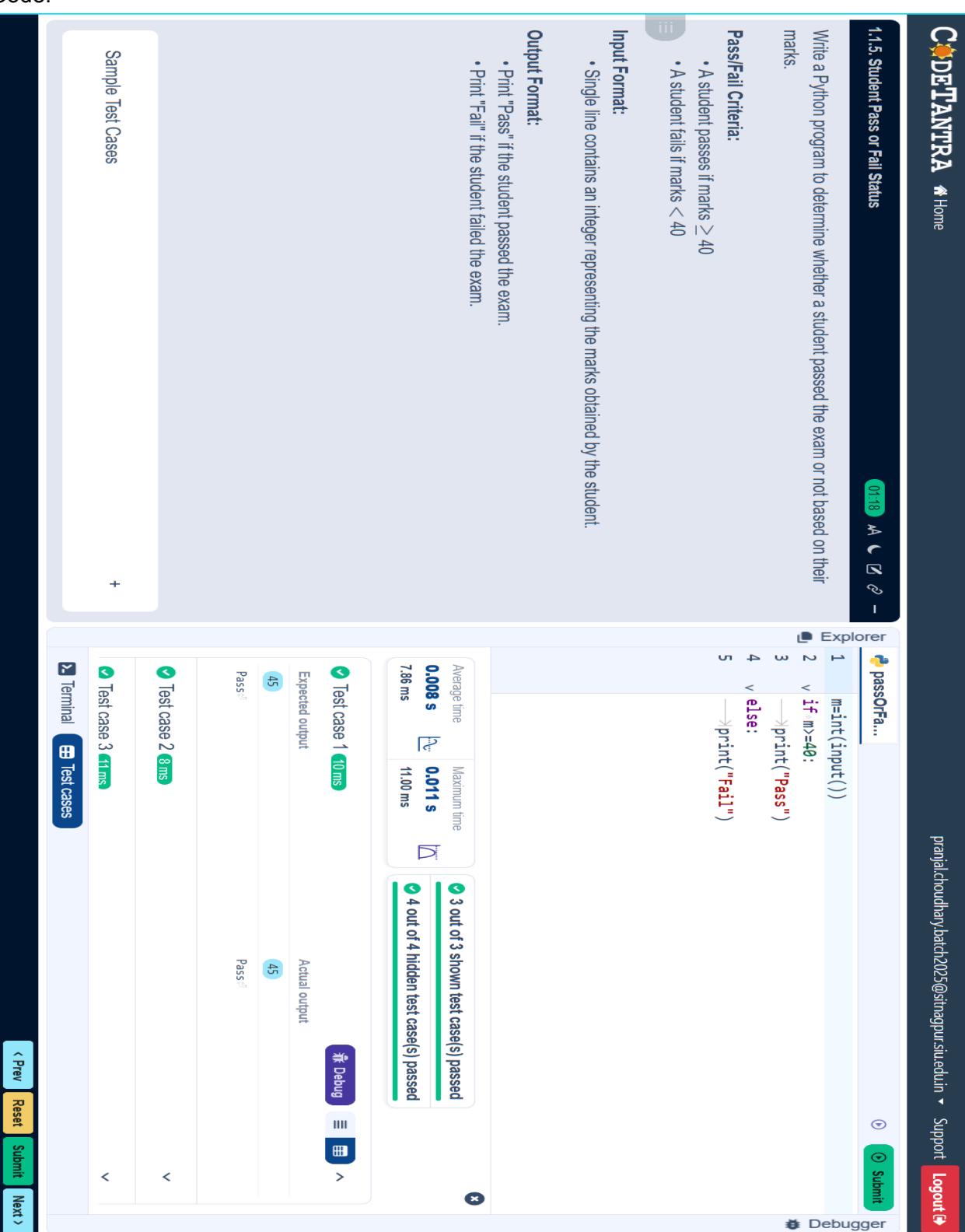
- A student passes if marks  $\geq 40$
- A student fails if marks  $< 40$

#### Input Format:

- Single line contains an integer representing the marks obtained by the student.

#### Output Format:

- Print "Pass" if the student passed the exam.
- Print "Fail" if the student failed the exam.



The screenshot shows a Python code editor with the following code:

```
passOrFail.py
1 m=int(input())
2 v if m>=40:
3     print("Pass")
4 v else:
5     print("Fail")
```

The code is part of a file named `passOrFail.py`. The output panel shows the results of running the code against three test cases:

Test Case	Average time	Maximum time	Status
Test case 1	0.008 s	0.011 s	Pass
Test case 2	7.86 ms	11.00 ms	Pass
Test case 3	4.11 ms	8 ms	Pass

The overall summary indicates that 3 out of 3 shown test case(s) passed, and 4 out of 4 hidden test case(s) passed.

At the bottom, there are navigation buttons: < Prev, Reset, Submit, Next >, and a Debugger button.