<u>Experiment 01:-</u>

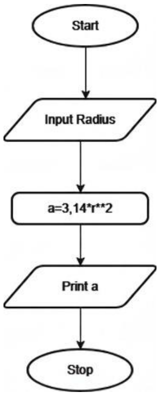
Problem Statement:-

Write a Python program that calculates the area of a circle when the radius is provided by the user. Use pi = 3.14 and display the area.

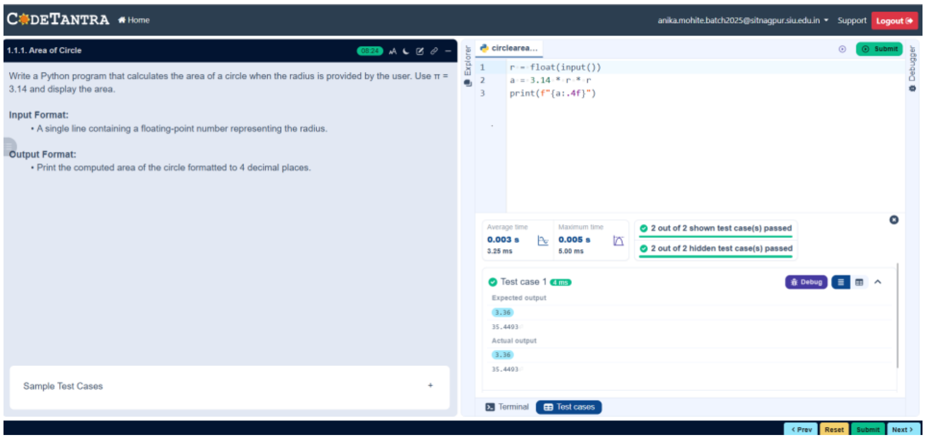
Algorithm:-

1. Start
2. Read the radius `r` from the user
3. Calculate the area using the formula:
   Area = 3.14 × r × r
4. Display the area
5. Stop.

Flowchart:-



Execution:-



1.1.1. Area of Circle

Write a Python program that calculates the area of a circle when the radius is provided by the user. Use π = 3.14 and display the area.

Input Format:

- A single line containing a floating-point number representing the radius.

Output Format:

- Print the computed area of the circle formatted to 4 decimal places.

Sample Test Cases +

```
r = float(input())
a = 3.14 * r * r
print(f"{a:.4f}")
```

Average time 0.003 s 3.25 ms

Maximum time 0.005 s 5.00 ms

2 out of 2 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Test case 1 4 ms

Expected output

```
3.36
35.4493
```

Actual output

```
3.36
35.4493
```

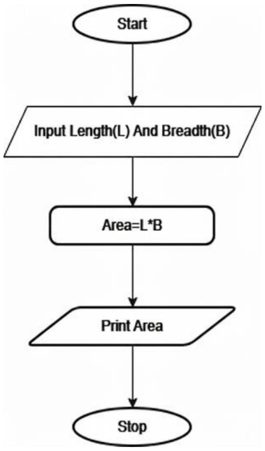
Problem Statement:-

Write a Python program to calculate the area of a rectangle given its length and width.

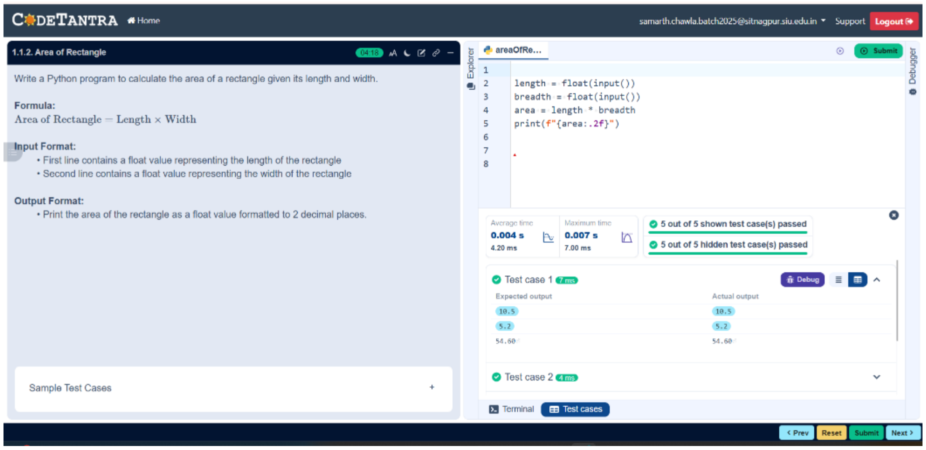
Algorithm:-

1. Start.
2. Read the length l and Width b from the user.
3. Calculate the area using the formula:
   Area = l*b.
4. Display the area.
5. Stop.

Flowchart:-



Execution:-



1.1.2. Area of Rectangle

Write a Python program to calculate the area of a rectangle given its length and width.

Formula:
Area of Rectangle = Length × Width

Input Format:
- First line contains a float value representing the length of the rectangle
- Second line contains a float value representing the width of the rectangle

Output Format:
- Print the area of the rectangle as a float value formatted to 2 decimal places.

Sample Test Cases +

```python

length = float(input())
breadth = float(input())
area = length * breadth
print(f"{area:.2f}")
```

Average time 0.004 s 4.20 ms

Maximum time 0.007 s 7.00 ms

5 out of 5 shown test case(s) passed

5 out of 5 hidden test case(s) passed

Test case 1 7 ms

Expected output	Actual output
10.5	10.5
5.2	5.2
54.60	54.60

Test case 2 4 ms

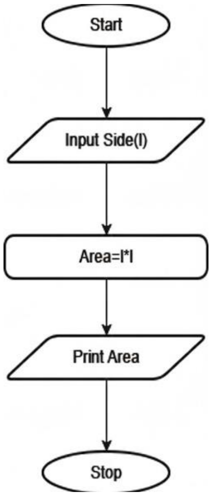
Problem Statement:-

Write a Python program that prompts the user to enter the Side Length of a square and computes the area of the square.

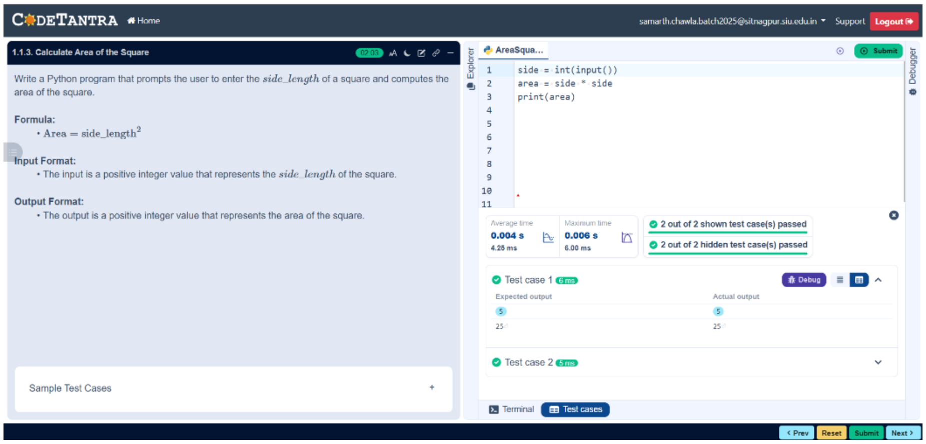
Algorithm:-

```
1. Start.
2. Read the side length s of the square.
3. Calculate the area using the formula: Area=$s×s$.
4. Display the area
5. Stop.
```

Flowchart:-



Execution:-



1.1.3. Calculate Area of the Square

Write a Python program that prompts the user to enter the *side_length* of a square and computes the area of the square.

Formula:

- $\text{Area} = \text{side\_length}^2$

Input Format:

- The input is a positive integer value that represents the *side_length* of the square.

Output Format:

- The output is a positive integer value that represents the area of the square.

Sample Test Cases +

```python
side = int(input())
area = side * side
print(area)
```

Average time 0.004 s 4.26 ms

Maximum time 0.006 s 6.00 ms

2 out of 2 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Test case 1 6 ms

Expected output	Actual output
5	5
25	25

Test case 2 5 ms

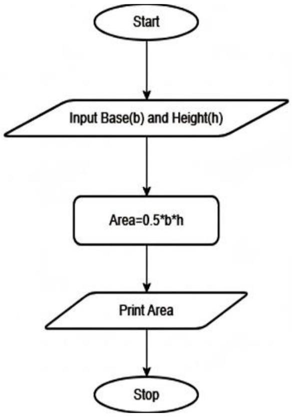
Problem Statement:-

Write a Python program that prompts the user to enter the triangle's base and height and computes the triangle's area.

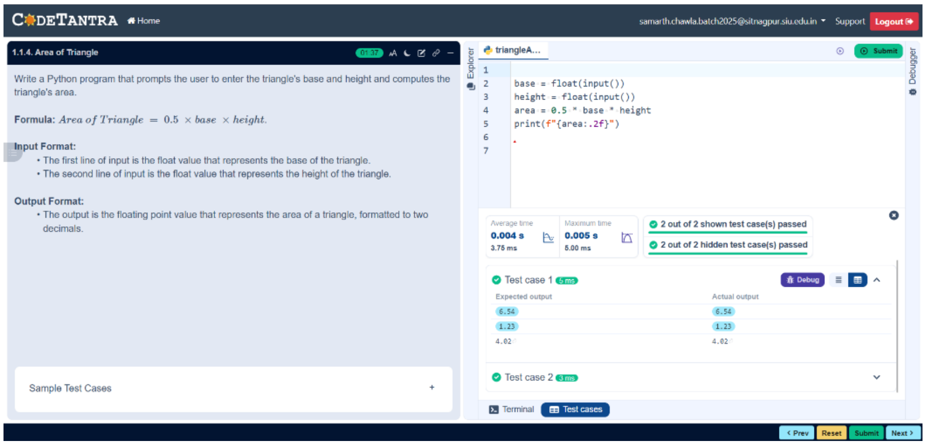
Algorithm:-

1. Start.
2. Read the base b and height h of the triangle.
3. Calculate the area using the formula: Area=0.5×b×h.
4. Display the area.
5. Stop.

Flowchart:-



Execution:-



1.1.4. Area of Triangle

Write a Python program that prompts the user to enter the triangle's base and height and computes the triangle's area.

Formula: $Area\ of\ Triangle = 0.5 \times base \times height.$

Input Format:

- The first line of input is the float value that represents the base of the triangle.
- The second line of input is the float value that represents the height of the triangle.

Output Format:

- The output is the floating point value that represents the area of a triangle, formatted to two decimals.

Sample Test Cases +

```

base = float(input())
height = float(input())
area = 0.5 * base * height
print(f"{area:.2f}")


```

Average time 0.004 s 3.75 ms

Maximum time 0.005 s 5.00 ms

2 out of 2 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Test case 1 5 ms

Expected output	Actual output
6.54	6.54
1.23	1.23
4.02	4.02

Test case 2 3 ms

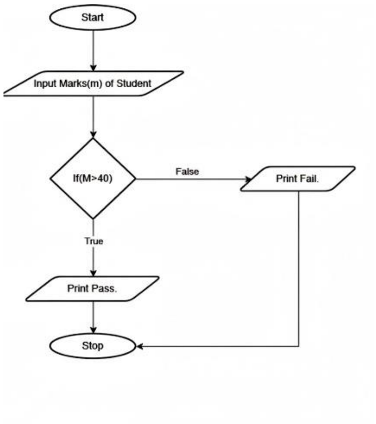
Problem Statement:-

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

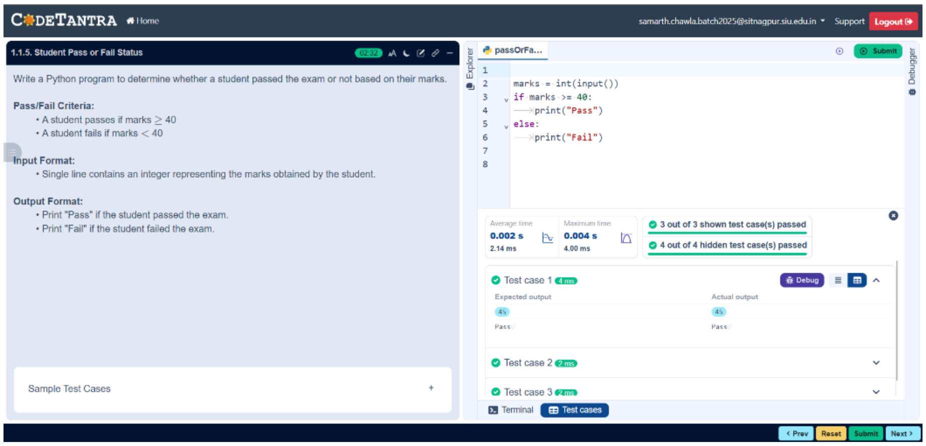
Algorithm:-

1. Start.
2. Read the marks obtained by the student.
3. If marks > 40 → Display “Pass”.
4. Else → Display “Fail”.
5. Stop.

Flowchart:-



Execution:-



1.1.5. Student Pass or Fail Status

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 ≥ 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.

Sample Test Cases

```

marks = int(input())
if marks >= 40:
    print("Pass")
else:
    print("Fail")
```

```
Average time        Maximum time
0.002 s             0.004 s
2.14 ms             4.00 ms
```

3 out of 3 shown test case(s) passed

4 out of 4 hidden test case(s) passed

Test case 1 4 ms

Expected output	Actual output
45	45
Pass	Pass

Test case 2 2 ms

Test case 3 2 ms

Experiment 02:-

Problem Statement:-

Write a program to find the roots of a quadratic equation, given its coefficients a, b, and c. Use the quadratic formula: (-b+-root(b**2-4ac))/2a.

Algorithm:-

1. Start.
2. Read the coefficients a, b, and c.
3. Calculate the discriminant using:- $D=b$^2−4ac.

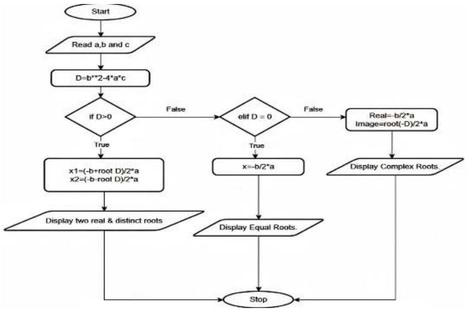
4. If D > 0 Compute:- $x1$=(−b+root$D$)/2a and $x2$=(−b−root$D$)/2a Display two real and distinct roots.

5. Else if D = 0 Compute:- x=−b/2a Display equal real roots.

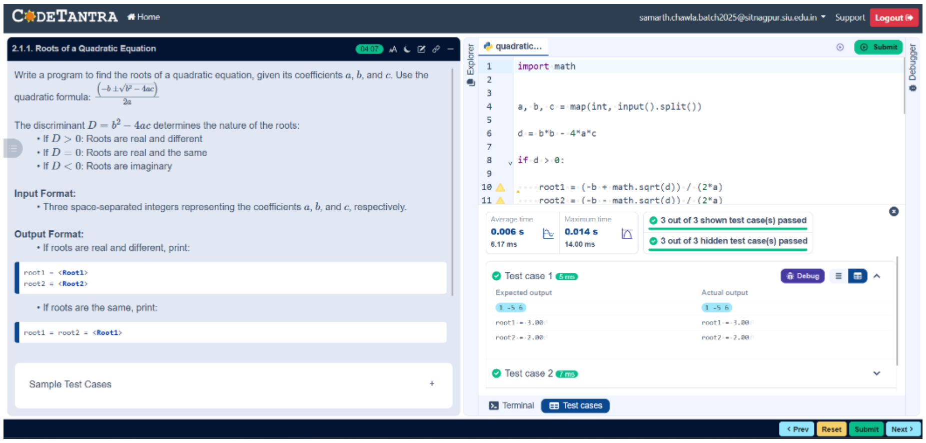
6. Else (D < 0) Compute:- Real part = −b/2a Imaginary part = root(-D)/2a Display complex roots.

7. Stop.

Flowchart:-



Execution:-



2.1.1. Roots of a Quadratic Equation

Write a program to find the roots of a quadratic equation, given its coefficients a, b, and c. Use the quadratic formula: $\frac{\left(-b \pm \sqrt{b^2-4ac}\right)}{2a}$

The discriminant $D = b^2 - 4ac$ determines the nature of the roots:
- If $D > 0$: Roots are real and different
- If $D = 0$: Roots are real and the same
- If $D < 0$: Roots are imaginary

Input Format:
- Three space-separated integers representing the coefficients a, b, and c, respectively.

Output Format:
- If roots are real and different, print:

```
root1 = <Root1>
root2 = <Root2>
```

- If roots are the same, print:

```
root1 = root2 = <Root1>
```

Sample Test Cases +

```
import math


a, b, c = map(int, input().split())

d = b*b - 4*a*c

if d > 0:

    root1 = (-b + math.sqrt(d)) / (2*a)
    root2 = (-b - math.sqrt(d)) / (2*a)
```

Average time 0.006 s 6.17 ms
Maximum time 0.014 s 14.00 ms
3 out of 3 shown test case(s) passed
3 out of 3 hidden test case(s) passed

Test case 1 5 ms

Expected output

```
1 -5 6
root1 = 3.00
root2 = 2.00
```

Actual output

```
1 -5 6
root1 = 3.00
root2 = 2.00
```

Test case 2 7 ms

Experiment 03:-

Problem Statement:-

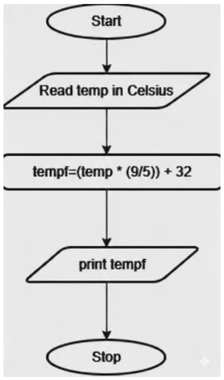
Write a Python program to convert temperature from Celsius to Fahrenheit.

Algorithm:-

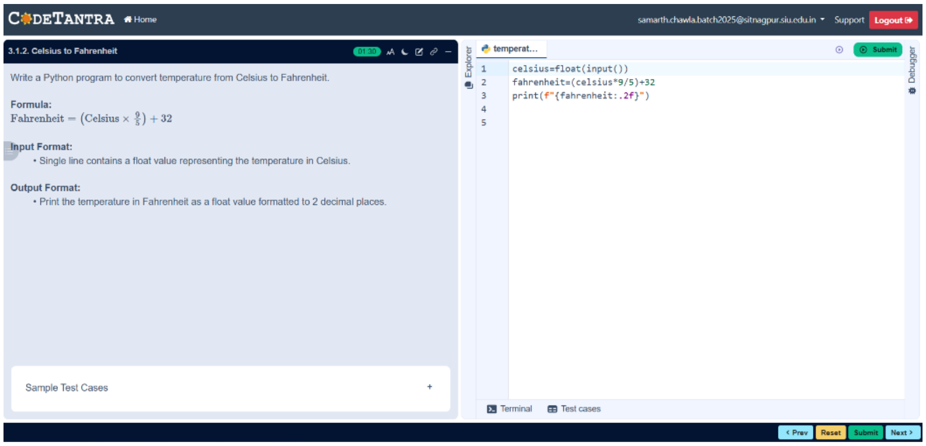
1. Start.
2. Input temperature in Celsius C.
3. Calculate Fahrenheit using the formula F = (C × 9/5) + 32.

4. Display the temperature in Fahrenheit.
5. Stop.

Flowchart:-



Execution:-



3.1.2. Celsius to Fahrenheit

Write a Python program to convert temperature from Celsius to Fahrenheit.

Formula:

$\text{Fahrenheit} = \left(\text{Celsius} \times \frac{9}{5}\right) + 32$

Input Format:

- Single line contains a float value representing the temperature in Celsius.

Output Format:

- Print the temperature in Fahrenheit as a float value formatted to 2 decimal places.

```
celsius=float(input())
fahrenheit=(celsius*9/5)+32
print(f"{fahrenheit:.2f}")
```

Problem Statement:-

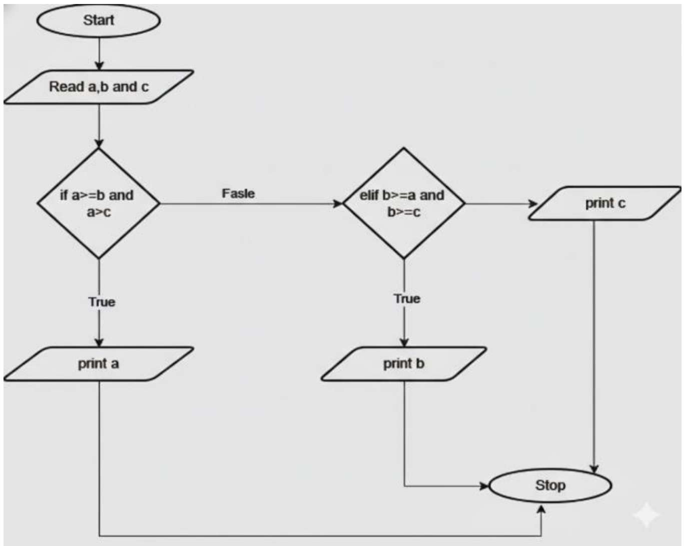
Write a Python program that prompts the user to enter three integers. Print the largest of the three integers.

Algorithm:-

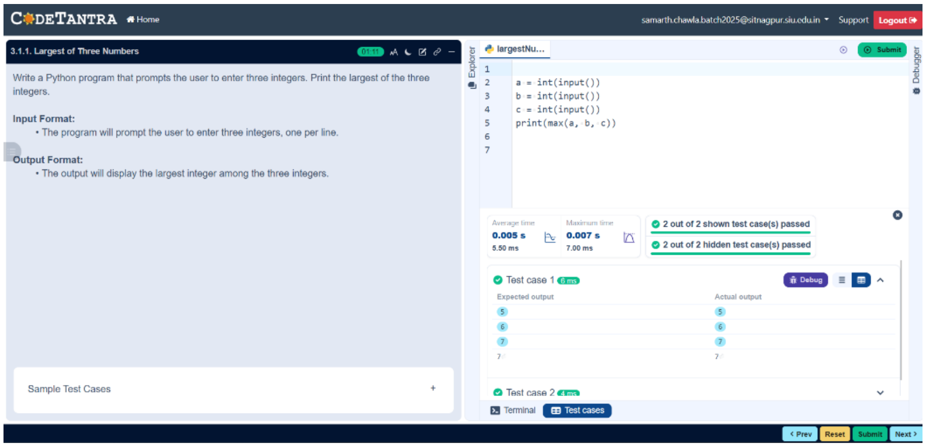
1. Start.

2. Input three integers a, b, and c.
3. If a ≥ b and a ≥ c → Print a as the largest number.
4. Else if b ≥ a and b ≥ c → Print b as the largest number.
5. Else → Print c as the largest number.
6. Stop

Flowchart:-



Execution:-



3.1.1. Largest of Three Numbers

Write a Python program that prompts the user to enter three integers. Print the largest of the three integers.

Input Format:

- The program will prompt the user to enter three integers, one per line.

Output Format:

- The output will display the largest integer among the three integers.

Sample Test Cases +

```

a = int(input())
b = int(input())
c = int(input())
print(max(a, b, c))


```

```
Average time     Maximum time
0.005 s          0.007 s
5.50 ms          7.00 ms

2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed

Test case 1 6 ms
Expected output      Actual output
5                    5
6                    6
7                    7
7                    7

Test case 2 4 ms
```

Experiment 04:-

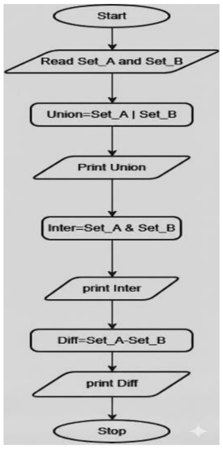
Problem Statement:-

Write a Python program to perform union, intersection and difference operations on Set A and Set B.

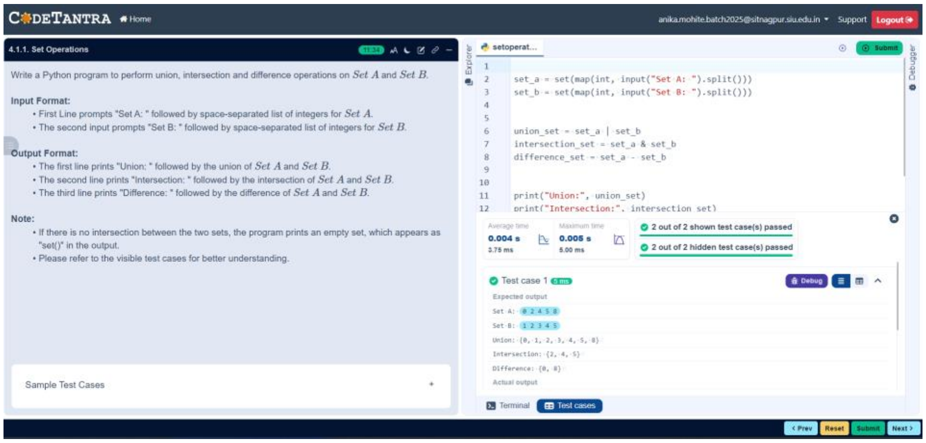
Algorithm:-

1. Start.
2. Declare two sets A and B
3. Read elements of Set A from the user
4. Read elements of Set B from the user
5. Perform Union operation Union = A ∪ B.
6. Perform Intersection operation Intersection = A ∩ B.
7. Perform Difference operations
8. Difference1 = A − B.
9. Display Set A and Set B
10. Display Union, Intersection, and Difference results
11. Stop.

Flowchart:-



Execution:-



4.1.1. Set Operations

Write a Python program to perform union, intersection and difference operations on *Set A* and *Set B*.

Input Format:

- First Line prompts "Set A: " followed by space-separated list of integers for *Set A*.
- The second input prompts "Set B: " followed by space-separated list of integers for *Set B*.

Output Format:

- The first line prints "Union: " followed by the union of *Set A* and *Set B*.
- The second line prints "Intersection: " followed by the intersection of *Set A* and *Set B*.
- The third line prints "Difference: " followed by the difference of *Set A* and *Set B*.

Note:

- If there is no intersection between the two sets, the program prints an empty set, which appears as "set()" in the output.
- Please refer to the visible test cases for better understanding.

```

set_a = set(map(int, input("Set A: ").split()))
set_b = set(map(int, input("Set B: ").split()))


union_set = set_a | set_b
intersection_set = set_a & set_b
difference_set = set_a - set_b


print("Union:", union_set)
print("Intersection:", intersection_set)
```

Average time 0.004 s 3.75 ms

Maximum time 0.005 s 5.00 ms

2 out of 2 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Test case 1 5 ms

Expected output

```
Set A: 0 2 4 5 8
Set B: 1 2 3 4 5
Union: {0, 1, 2, 3, 4, 5, 8}
Intersection: {2, 4, 5}
Difference: {0, 8}
```

Actual output