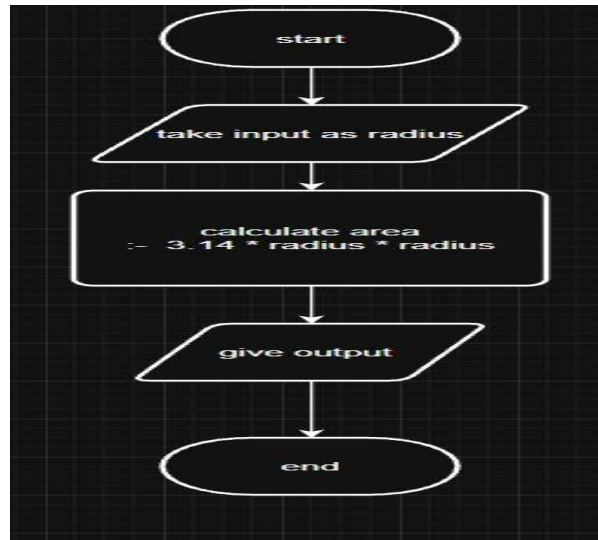


1.1.1. Area of Circle

Algorithm (Step-by-Step)

1. Start
2. Input: Read an integer value representing the year.
3. Check divisibility:
 - If the year is divisible by 400 → It is a leap year.
 - Else if the year is divisible by 100 → It is not a leap year.
 - Else if the year is divisible by 4 → It is a leap year.
 - Otherwise → It is not a leap year.
4. Output: Print "Leap year" or "Not a leap year" accordingly.
5. End



The screenshot displays the CodeTANTRA IDE interface. On the left, the problem statement for '1.1.1. Area of Circle' is shown, including input and output formats. The main editor contains a Python script that reads a radius from the user and calculates the area using the formula $A = 3.14 * r * r$. The output is formatted to 4 decimal places. Below the editor, the test cases section shows two sample test cases. The right sidebar displays the execution results, indicating that 2 out of 2 shown test cases passed, with a detailed view of the first test case showing the expected and actual outputs.

```
1 r = float(input())
2 a = 3.14 * r * r
3
4
5
6 print(f"{a:.4f}")
```

Test case 1

Expected output	Actual output
35.4493	35.4493

Test case 2

Expected output	Actual output
12.5600	12.5600

Practicle 1.1.2

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

***Algorithm: Area of Rectangle**

Step 1. Start

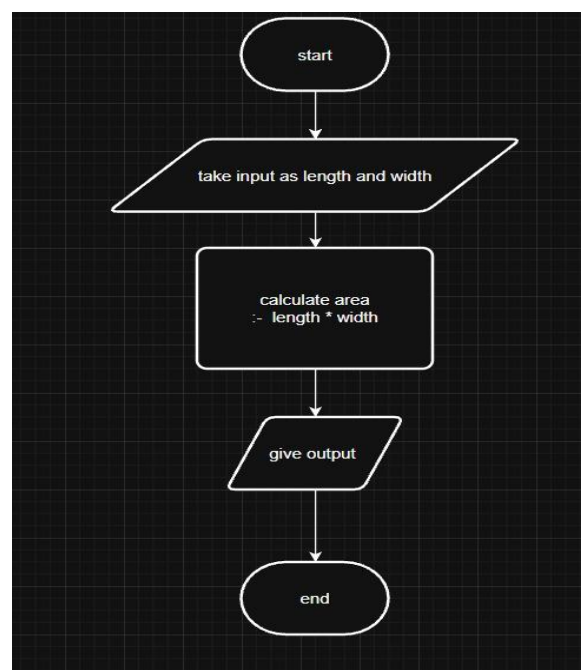
Step 2. Input length L and breadth B

Step 3. Calculate area $A = L \times B$

Step 4. Display the value of A

Step 5. Stop

***Flowchart :-**



The screenshot shows the CodedTANTRA IDE interface. On the left, there are sample test cases for the program. The main editor displays the following Python code:

```
1 #-Type Content here...
2 l=float(input())
3 w=float(input())
4 area=l*w
5 print(f"area:.2f")
6
7
8
9
10
11
12
13
```

Below the code editor, the test results are shown:

- Average time: 0.003 s, Maximum time: 0.006 s, 3.20 ms
- 5 out of 5 shown test case(s) passed
- 5 out of 5 hidden test case(s) passed

The test cases are as follows:

Test case	Expected output	Actual output
Test case 1	11.35	11.35
Test case 2	54.60	54.60
Test case 3	8.75	8.75
Test case 4	5886.00	5886.00
Test case 5		

At the bottom, there are buttons for 'PREV', 'RESET', 'SUBMIT', and 'NEXT'.

Practicle 1.1.3

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

***Algorithm: Area of a Square**

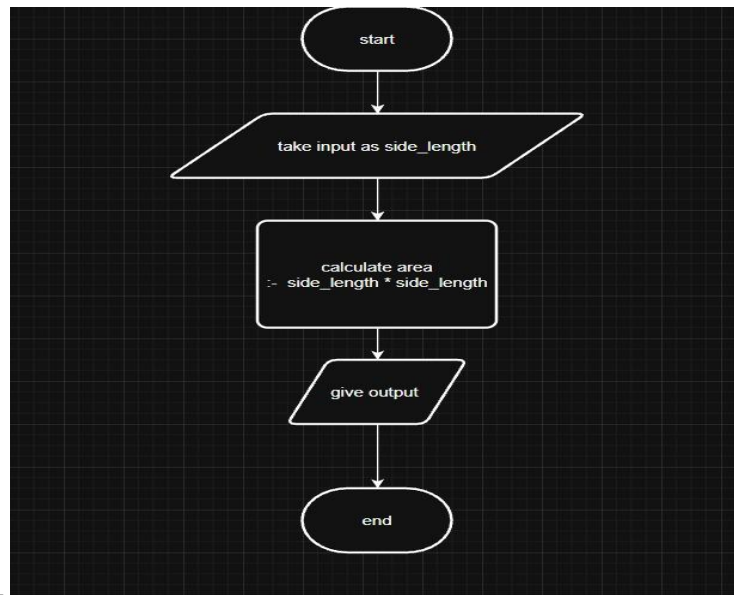
Step 1. Start

Step 2. Input the value of *side_length*

Step 3. Calculate area using formula:
 $\text{area} = \text{side_length} \times \text{side_length}$

Step 4. Display the value of area

Step 5. Stop



***Flowchat:-**

The screenshot shows the CodeTANTRA IDE interface. On the left, the problem statement for '1.1.3. Calculate Area of the Square' is displayed, including the formula $\text{Area} = \text{side_length}^2$ and input/output formats. The main editor shows a Python program:

```
1 side_length = int(input())
2 area = side_length ** 2
3 print(area)
```

Below the code editor, the test results are shown. The program has passed 2 out of 2 shown test cases and 2 out of 2 hidden test cases. The test cases are:

Test Case	Expected Output	Actual Output
Test case 1	25	25
Test case 2	36	36

The bottom of the interface includes a terminal, a 'Test Cases' button, and navigation buttons like 'PREV', 'RESET', 'SUBMIT', and 'NEXT'.

Practicle 1.1.4

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

***Algorithm: Area of a Triangle**

Step 1. Start

Step 2. Input the value of base (B) and height (H)

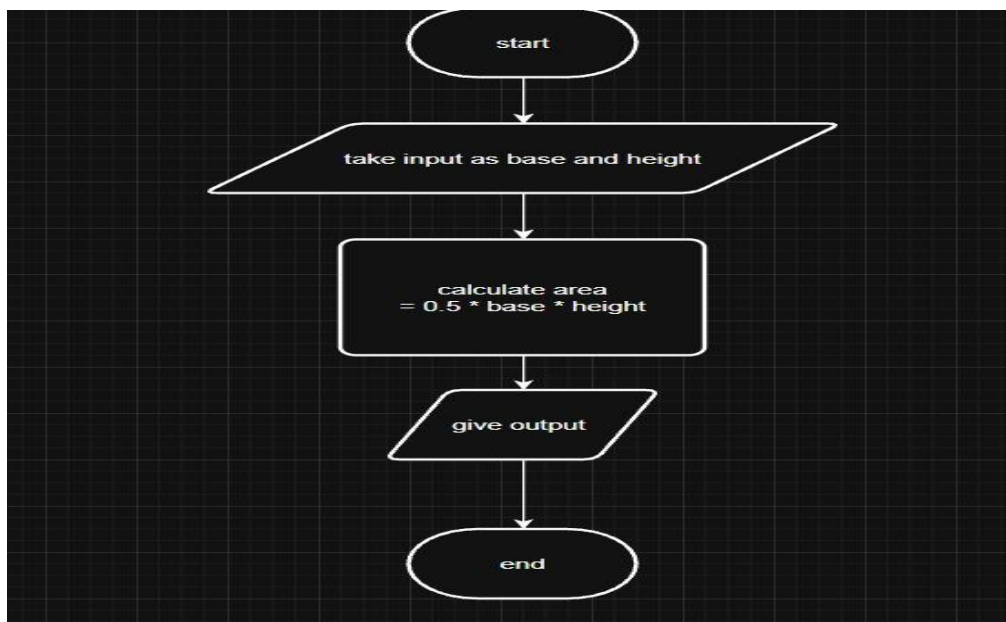
Step 3. Calculate area using formula:

$$\text{area} = (B \times H) / 2$$

Step 4. Display the value of area

Step 5. Stop

***Flowchat:-**



CODETANTRA Home shreyas.gabhiye.batch2025@sitnagpur.su.edu.in Support Logout

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: $\text{Area of Triangle} = 0.5 \times \text{base} \times \text{height}$.

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

Test case 1

4.5

4.82

Test case 2

100.80

triangleA...

```
1 base = float(input())
2 height = float(input())
3 area = 0.5 * base * height
4 print(f"{area:.2f}")
5
6
```

Average time: 0.003 s Maximum time: 0.004 s 2 out of 2 shown test case(s) passed 2 out of 2 hidden test case(s) passed

Test case 1

Expected output: 4.52 Actual output: 4.52

Test case 2

Expected output: 100.80 Actual output: 100.80

Terminal Test cases

PREV RESET SUBMIT NEXT

Practicle 1.1.5 :-

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

***Algorithm: Pass or Fail**

Step 1. Start

Step 2. Input the marks obtained by the student

Step 3. If marks ≥ 40

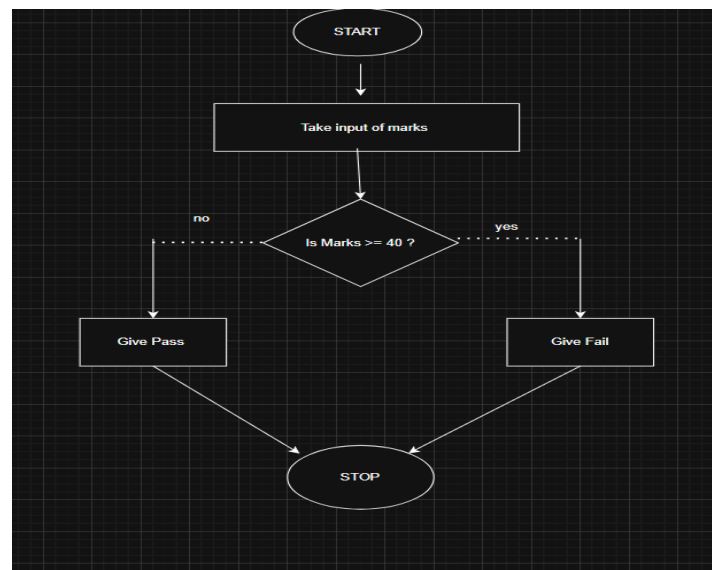
→ Display "Pass"

1. **Else**

→ Display "Fail"

Step 4. Stop

Flowchart :-



The screenshot shows the CodeTANTRA IDE interface. On the left, the problem statement for '1.1.5. Student Pass or Fail Status' is displayed, including the criteria (marks ≥ 40 for Pass, marks < 40 for Fail) and sample test cases. The main editor shows the following Python code:

```
1 marks = int(input())
2 if marks >= 40:
3     print("Pass")
4 else:
5     print("Fail")
6
7
```

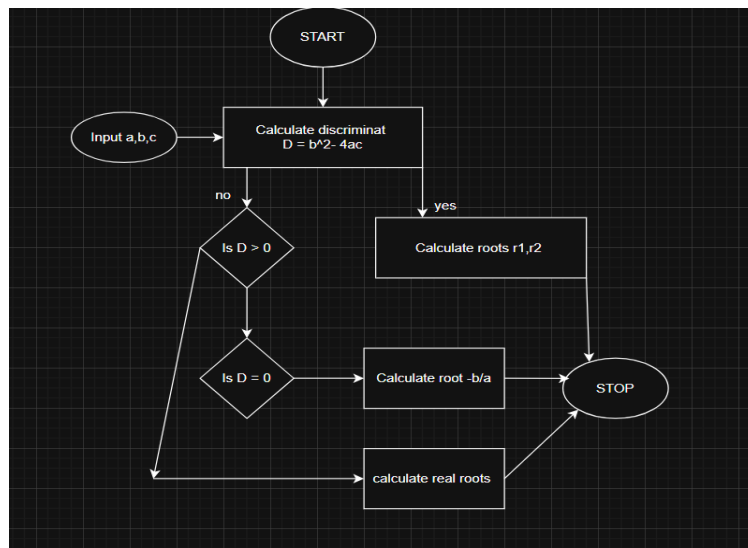
At the bottom, the test results are shown: '3 out of 3 shown test case(s) passed' and '4 out of 4 hidden test case(s) passed'. The test cases are listed as follows:

Test Case	Expected Output	Actual Output	Status
Test case 1	Pass	Pass	Passed
Test case 2	Fail	Fail	Passed
Test case 3	Pass	Pass	Passed

Practicle 2.1.1

Aim :- Write a program to find the roots of a quadratic equation, given its coefficients a , b , and c . Use the quadratic formula:

- Algorithm:- Step 1: Start
- Step 2: Input the coefficients a , b , and c
- Step 3: Calculate the discriminant $D = b^2 - 4ac$
- Step 4: If $D > 0$, calculate $\text{root1} = (-b + \sqrt{D}) / (2a)$ and $\text{root2} = (-b - \sqrt{D}) / (2a)$ and display both roots
- Step 5: Else if $D = 0$, calculate $\text{root} = -b / (2a)$ and display $\text{root1} = \text{root2} = \text{root}$
- Step 6: Else if $D < 0$, calculate real part $= -b / (2a)$ and imaginary part $= \sqrt{-D} / (2a)$ and display $\text{root1} = \text{real} + \text{imaginary } i$ and $\text{root2} = \text{real} - \text{imaginary } i$
- Step 7: Stop



CODETANTRA Home

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:

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

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

Sample Test Cases

Test case 1

root1 = 3.00
root2 = 2.00

Test case 2

root1 = root2 = 2.00

Test case 3

root1 = -1.00+2.00i
root2 = -1.00-2.00i

quadratic...

```
1 import math
2
3 a, b, c = map(int, input().split())
4 D = b**2 - 4*a*c
5
6 if D > 0:
7     root1 = (-b + math.sqrt(D)) / (2*a)
8     root2 = (-b - math.sqrt(D)) / (2*a)
9     print(f"root1 = {root1:.2f}")
10    print(f"root2 = {root2:.2f}")
11 elif D == 0:
12     root = -b / (2*a)
13     print(f"root1 = root2 = {root:.2f}")
14 else:
15     real = -b / (2*a)
16     imag = math.sqrt(-D) / (2*a)
17     print(f"root1 = {real:.2f}+{imag:.2f}i")
```

Average time: 0.002 s, Maximum time: 0.003 s, 3 out of 3 shown test case(s) passed, 3 out of 3 hidden test case(s) passed

Test case 1: Expected output: root1 = 3.00, root2 = 2.00; Actual output: root1 = 3.00, root2 = 2.00

Test case 2: Expected output: root1 = root2 = 2.00; Actual output: root1 = 2.00, root2 = 2.00

Test case 3: Expected output: root1 = -1.00+2.00i, root2 = -1.00-2.00i; Actual output: root1 = -1.00+2.00i, root2 = -1.00-2.00i

Terminal, Test cases, < PREV, RESET, SUBMIT, NEXT >

Practicle 3.1.1

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

*Algorithm :- Step 1. **Start**: The entry point of the program.

Step 2. **Input a, b, c**: You enter the three integers.

Step 3. **Decision 1 (is a >= b AND a >= c?)**:

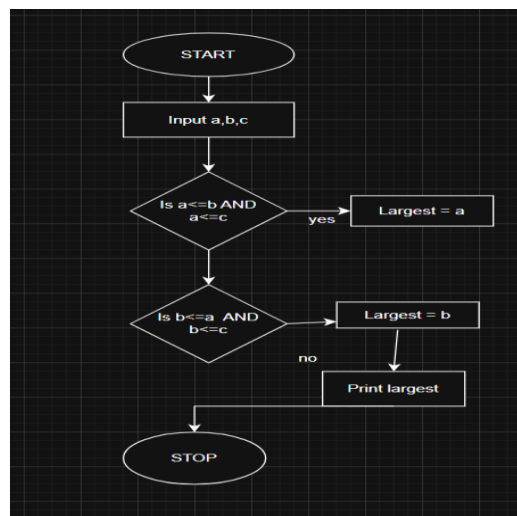
- **Yes**: largest = a.
- **No**: Move to the next check.

Step 4. **Decision 2 (is b >= a AND b >= c?)**:

- **Yes**: largest = b.
- **No**: By elimination, largest = c.

Step 5. **Output**: Print the value of largest.

Step 6. **End**: The program terminates.



The screenshot shows the CodeTANTRA IDE interface. On the left, there's a sidebar with the problem title "3.1.1. Largest of Three Numbers" and instructions: "Write a Python program that prompts the user to enter three integers. Print the largest of the three integers." Below this, it shows the input and output formats, and sample test cases. The main editor area contains the following Python code:

```
1 a = int(input())
2 b = int(input())
3 c = int(input())
4
5 largest = max(a, b, c)
6 print(largest)
7
8
```

On the right, there's a terminal window showing the execution results. It indicates that 2 out of 2 shown test case(s) passed and 2 out of 2 hidden test case(s) passed. The average time is 0.004 s and the maximum time is 0.005 s. The expected output and actual output are both shown as "1".

Practicle 3.1.2

*Aim :- Write a Python program to convert temperature from Celsius to Fahrenheit.

*Algorithm:- **Celsius to Fahrenheit Conversion**

Step 1. Start: Initialize the program.

Step 2. Input: Prompt the user and read a float value representing the temperature in Celsius.

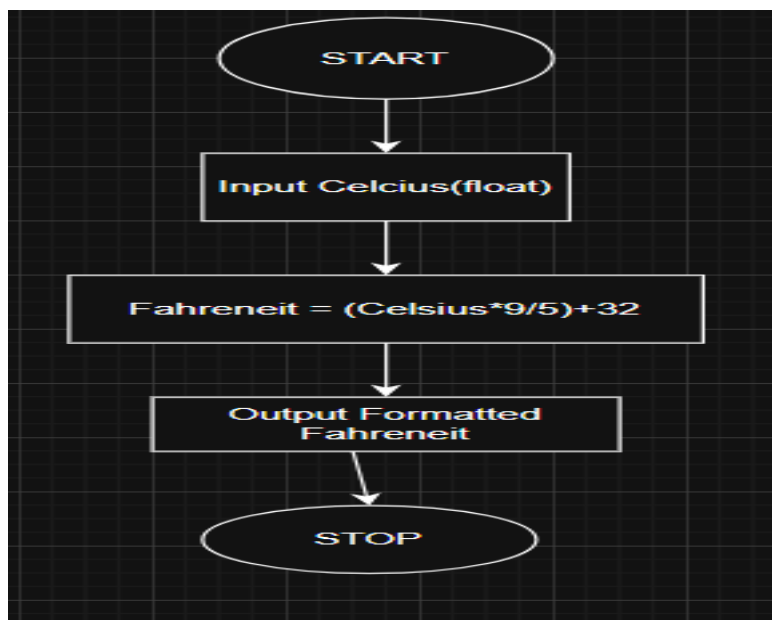
Step 3. Calculation: Apply the conversion formula:

- $\text{Fahrenheit} = (\text{Celsius} * 9/5) + 32$

Step 4. Formatting : Format the resulting fahrenheit value to **2 decimal places**.

Step 5. Output: Display the formatted Fahrenheit value.

Step 6. End: Terminate the program.



The screenshot shows the CodeTANTRA IDE interface. On the left, the problem statement for '3.1.2. Celsius to Fahrenheit' is displayed, including the formula $\text{Fahrenheit} = (\text{Celsius} \times \frac{9}{5}) + 32$ and sample test cases. The main editor shows the following Python code:

```
1 celsius = float(input())
2 fahrenheit = (celsius * 9/5) + 32
3 print(f"{fahrenheit:.2f}")
4
5
```

On the right, the test results are shown, indicating that all test cases passed:

- 4 out of 4 shown test case(s) passed
- 4 out of 4 hidden test case(s) passed

Test Case	Expected output	Actual output
Test case 1	32.00	32.00
Test case 2	99.50	99.50
Test case 3	-40.00	-40.00
Test case 4	77.00	77.00

Practicle :- 4.1.1

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

***Algorithm: Set Operations (Union, Intersection, Difference)**

Step 1. Start

Step 2 . Prompt the user to enter elements of **Set A** as space-separated integers.

Step 3. Read the input and convert it into a set set_a.

Step 4. Prompt the user to enter elements of **Set B** as space-separated integers.

Step 5. Read the input and convert it into a set set_b.

Step 6. Find the **union** of set_a and set_b using the union operation.

Step 7. Find the **intersection** of set_a and set_b using the intersection operation.

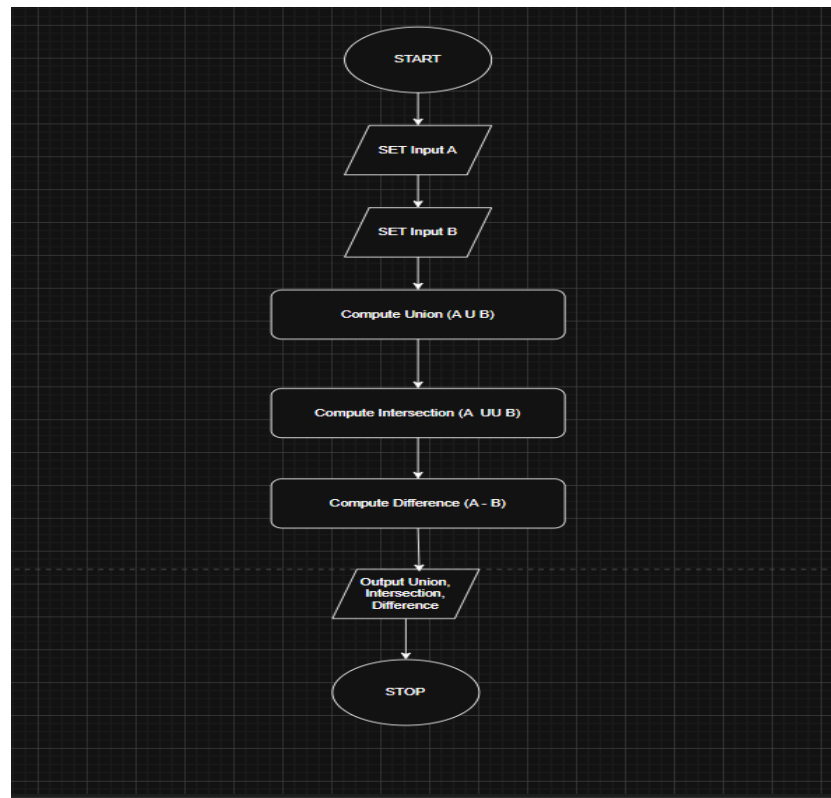
Step 8. Find the **difference** of set_a and set_b (elements present in Set A but not in Set B).

Step 9. Display the union result.

Step 10. Display the intersection result.

Step 11. Display the difference result.

Step 12. Stop



CODETANTRA Home

shreyas.gajbiye.hatch2025@sitnagpur.siu.edu.in Support Logout

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:

Sample Test Cases

Test case 1

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

Test case 2

Set A: 10 11 22 33 44 55
Set B: 15 16 17 18 19 20
Union: {10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 22, 33, 44, 55}
Intersection: set()
Difference: {33, 10, 11, 44, 22, 55}

```

1 # Read input sets
2 setA = set(map(int, input("Set A: ").split()))
3 setB = set(map(int, input("Set B: ").split()))
4
5 # Perform operations
6 union_set = setA.union(setB)
7 intersection_set = setA.intersection(setB)
8 difference_set = setA.difference(setB)
9
10 # Print results
11 print("Union:", union_set)
12 print("Intersection:", intersection_set)
13 print("Difference:", difference_set)
14
15

```

Average time: 0.003 s Maximum time: 0.004 s
3.29 ms 4.00 ms

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

Test case 1

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

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

Terminal Test cases

PREV RESET SUBMIT NEXT

5.1.1. Leap Year Checker

1. **Start**
2. **Input:** Read an integer value for the year.
3. **Decision 1:** If the year is divisible by 400 → Print "Leap year".
4. **Decision 2:** Else if the year is divisible by 100 → Print "Not a leap year".
5. **Decision 3:** Else if the year is divisible by 4 → Print "Leap year".
6. **Else** → Print "Not a leap year".
7. **End**