

EXPERIMENT - 2

Name : Kanhaiya Tyagi

PRN : 25070521157

2.1.1 Roots of Quadratic Equation

ALGORITHM

Step 1 :-Start

Step 2 :- Import the math library.

Step 3 :- Read three integers a, b, and c (coefficients of the quadratic equation).

Step 4 :-Calculate the discriminant

$$D = b^2 - 4ac$$

Step 5 :- If $D > 0$:

Calculate two real and different roots using:

$$\frac{-b+\sqrt{D}}{2a}, \frac{-b-\sqrt{D}}{2a}$$

Print both roots up to 2 decimal places.

Step 6 :- Else if $D == 0$:

Calculate the single repeated root:

$$\frac{-b}{2a}$$

Print the root twice up to 2 decimal places.

Step 7 :- Else ($D < 0$):

Calculate real part:

$$\frac{-b}{2a}$$

Calculate imaginary part:

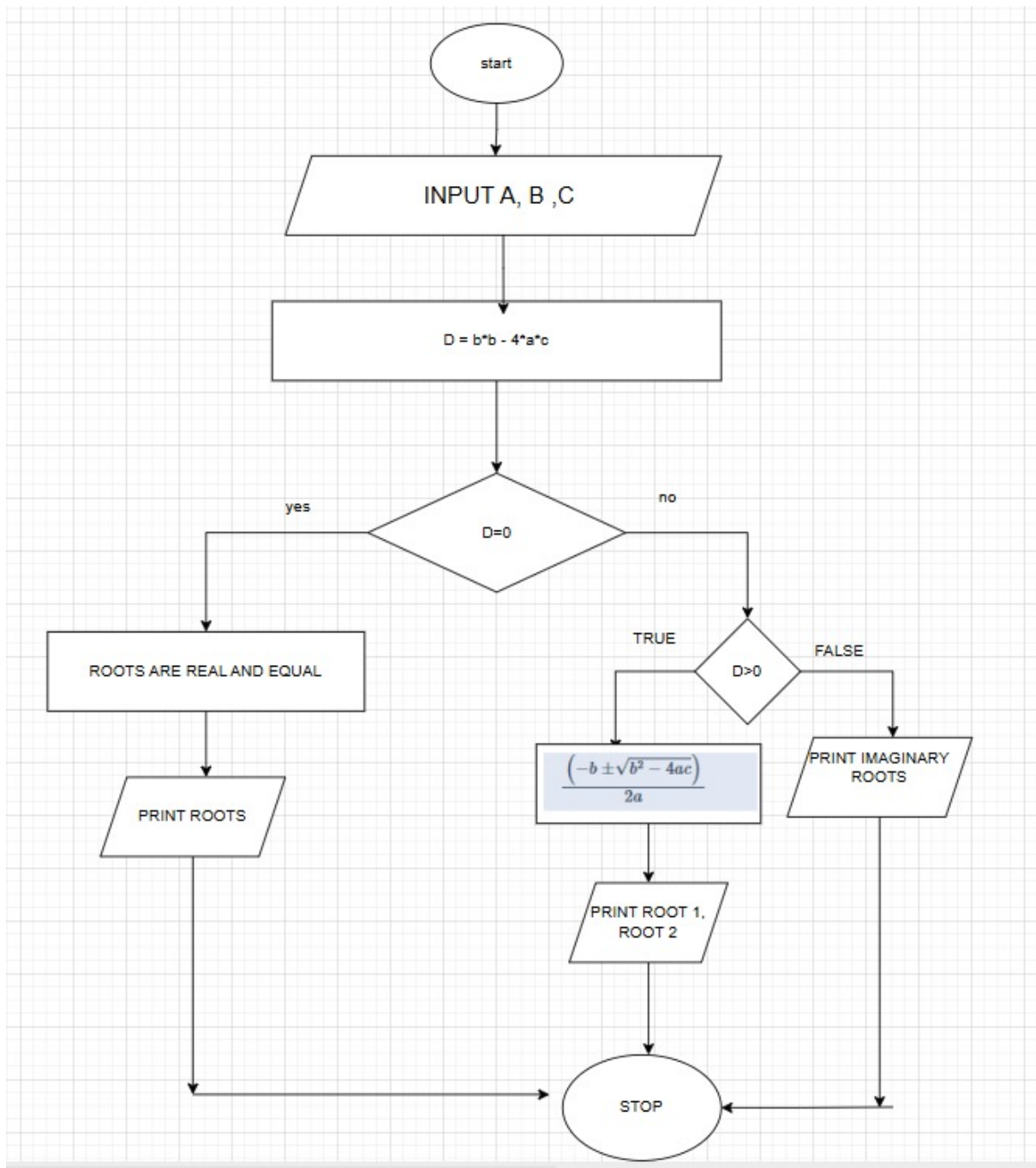
$$\frac{\sqrt{-D}}{2a}$$

Print both complex roots up to 2 decimal places.

Step 8 :- Stop

FLOWCHART

EXPERIMENT - 2



PHYTHON CODE

```
import math
```

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

```
D = b*b - 4*a*c
```

EXPERIMENT - 2

if $D > 0$:

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

```
root2 = (-b - math.sqrt(D)) / (2*a)
```

```
print(f"root1 = {root1:.2f}")
```

```
print(f"root2 = {root2:.2f}")
```

elif $D == 0$:

```
root = (-b) / (2*a)
```

```
print(f"root1 = root2 = {root:.2f}")
```

else:

```
real = (-b) / (2*a)
```

```
imag = math.sqrt(-D) / (2*a)
```

```
print(f"root1 = {real:.2f}+{imag:.2f}i")
```

```
print(f"root2 = {real:.2f}-{imag:.2f}i")
```

EXECUTION

The screenshot displays the CODETANTRA online IDE interface. On the left, the problem statement for "2.1.1. Roots of a Quadratic Equation" is visible, including the quadratic formula and instructions on how to handle different discriminant values. The main editor shows the Python code implementing the solution. The code reads coefficients a, b, and c, calculates the discriminant D, and prints the roots according to the logic defined in the experiment. The output section shows the results of three test cases, all of which passed. The first test case shows roots 1.50 and -3.00, the second shows roots 3.00 and 2.00, and the third shows complex roots 1.00+1.00i and 1.00-1.00i. The bottom of the interface includes a terminal, test cases section, and navigation buttons like Prev, Reset, Submit, and Next.

```
1 a, b, c = map(float, input().split())
2 D = (b*b) - (4*a*c)
3 sqrtD = D ** 0.5
4 root1 = (-b+sqrtD)/(2*a)
5 root2 = (-b-sqrtD)/(2*a)
6 if D > 0:
7     print(f"root1 = {root1:.2f}")
8     print(f"root2 = {root2:.2f}")
9 elif D == 0:
10    print(f"root1 = root2 = {root1:.2f}")
11 else:
12    print(f"root1 = {root1.real:.2f}{root1.imag:+.2f}i")
13    print(f"root2 = {root2.real:.2f}{root2.imag:+.2f}i")
```

Test case 1: Expected output: 1.50, -3.00; Actual output: 1.50, -3.00. Test case 2: Expected output: 3.00, 2.00; Actual output: 3.00, 2.00. Test case 3: Expected output: 1.00+1.00i, 1.00-1.00i; Actual output: 1.00+1.00i, 1.00-1.00i.