Demo 1: Quadratic Equation

1. Define a quadratic equation

A quadratic equation can be written as:

$$ax^2 + bx + c = 0 \tag{1}$$

where x is the variable, and a, b, c are the coefficients.

Task:

Define a function solve(a, b, c) to find the solution for Equation (1).

If there is valid solution, return None.

Code Format:

def solve(a, b, c):

```
    # Write your code here.

    if has_solution:
        return x1, x2
    else:
        return None
```

Noting that sqrt function in Numpy library can implement find the squared root of a variable.

import numpy as np; np.sqrt(x)

Hide Solution

Solutions for the quadratic equation:

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

holds if $b^2 - 4ac \geq 0$

In [1]:

```
import numpy as np

def solve(a, b, c):
    b2_4ac = b**2 - 4*a*c
    has_solution = b2_4ac >= 0
    if has_solution:
        x1 = (-b + np.sqrt(b2_4ac))/(2*a)
        x2 = (-b - np.sqrt(b2_4ac))/(2*a)
        return (x1, x2)
    else:
        return None
```

2. Plot the quadratic function (using matploblib library)

Matploblib is a third-party librar for data visualization in Python.

Capable of plotting: Lines, Charts, Curves, Scatter, etc.

Task: Use Matploblib.pyplot to plot the curve for equation:

$$f(x) = ax^2 + bx + c \tag{3}$$

E.g. a = 1, b = -2, c = 1

Hide Solution

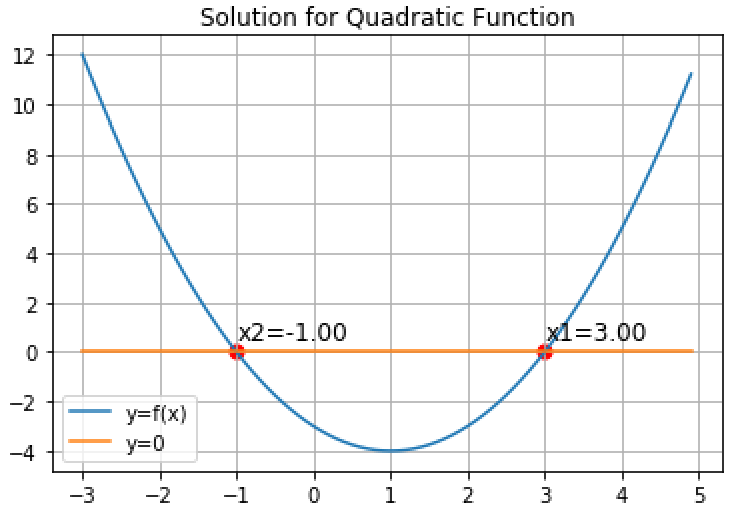
In [2]:

```
import matplotlib.pyplot as plt
%matplotlib inline

fig = plt.figure(1)
x = np.arange(-3.0, 5.0, 0.1)
a, b, c = 1.0, -2.0, -3.0
y = a*x**2 + b*x + c
plt.plot(x, y)
plt.plot(x, [0 for _ in x])
plt.legend(['y=f(x)', 'y=0'])

x1, x2 = solve(a, b, c)
plt.scatter([x1, x2], [0, 0], color='red', marker='o', s=50)
plt.text(x1, 0.5, s='x1={:.2f}'.format(x1), size='large')
plt.text(x2, 0.5, s='x2={:.2f}'.format(x2), size='large')

plt.grid(True)
plt.title('Solution for Quadratic Function')
plt.show()
```


Solution for Quadratic Function
x2=-1.00
x1=3.00
y=f(x)
y=0