Write a function which solves a quadratic equation, $x^2 + bx + c = 0$. Your function must receive two values for coefficients and return two real or complex roots. Always return two roots, even if the equation has a single double root.

In [ ]:

```
from math import sqrt
from cmath import sqrt as c_sqrt

def solve_quad(b, c):
    '''Solve the quadratic equation `x**2 + b*x + c == 0`.

    Parameters
    ----------
    b, c : float
       Coefficients

    Returns
    -------
    x1, x2 : float or complex
        Roots.
    '''
    ### BEGIN SOLUTION
    d = b**2 - 4*c
    sqd = sqrt(d) if d >= 0 else c_sqrt(d)
    if b > 0:
        x1 = (-b - sqd) / 2.
    else:
        x1 = (-b + sqd) / 2.
    return x1, c / x1
    ### END SOLUTION
```

In [ ]:

```
#
# Your implementation should pass tests in this cell.
#
# Do not remove or alter this cell. You may run it, but do not remove or alter i
t.
# Your changes to this cell will be ignored on grading.
# There will be additional tests.
#
from numpy.testing import assert_allclose

variants = [{'b': 4.0, 'c': 3.0},
            {'b': 2.0, 'c': 1.0},
            {'b': 0.5, 'c': 4.0},
            {'b': 1e10, 'c': 3.0},
            {'b': -1e10, 'c': 3.0},
           ]

for var in variants:
    x1, x2 = solve_quad(**var)
    assert_allclose(x1*x2, var['c'], rtol=1e-15)
```

In [ ]:

```
### BEGIN HIDDEN TESTS
from numpy.random import default_rng
rng = default_rng(12345)

c_vals = rng.uniform(size=10)*20 - 10
vars2 = [{"b": b, "c": c} for b in [1e10, -1e10, 0.5]
                          for c in c_vals]

from numpy.testing import assert_allclose

for var in vars2:
    x1, x2 = solve_quad(**var)
    assert_allclose(x1*x2, var['c'], rtol=1e-15)
### END HIDDEN TESTS
```

In [ ]:

```

```

Before you turn this problem in, make sure everything runs as expected. First, **restart the kernel** (in the menubar, select Kernel→Restart) and then **run all cells** (in the menubar, select Cell→Run All).

Make sure you fill in any place that says `YOUR CODE HERE` or "YOUR ANSWER HERE", as well as your name and collaborators below:

In [ ]:

```
NAME = ""
COLLABORATORS = ""
```

Write a function which solves a quadratic equation, $x^2 + bx + c = 0$. Your function must receive two values for coefficients and return two real or complex roots. Always return two roots, even if the equation has a single double root.

In [ ]:

```
from math import sqrt
from cmath import sqrt as c_sqrt

def solve_quad(b, c):
    '''Solve the quadratic equation `x**2 + b*x + c == 0`.

    Parameters
    ----------
    b, c : float
       Coefficients

    Returns
    -------
    x1, x2 : float or complex
        Roots.
    '''
    # YOUR CODE HERE
    raise NotImplementedError()
```

In [ ]:

```
#
# Your implementation should pass tests in this cell.
#
# Do not remove or alter this cell. You may run it, but do not remove or alter i
t.
# Your changes to this cell will be ignored on grading.
# There will be additional tests.
#
from numpy.testing import assert_allclose

variants = [{'b': 4.0, 'c': 3.0},
            {'b': 2.0, 'c': 1.0},
            {'b': 0.5, 'c': 4.0},
            {'b': 1e10, 'c': 3.0},
            {'b': -1e10, 'c': 3.0},
           ]

for var in variants:
    x1, x2 = solve_quad(**var)
    assert_allclose(x1*x2, var['c'], rtol=1e-15)
```

In [ ]:

```

```

In [ ]:

```

```