```
#!/usr/bin/env python

import functions as F
import numpy as N
import unittest

class TestFunctions(unittest.TestCase):
    def testApproxJacobian1(self):
        slope = 3.0
        def f(x):
            return slope * x + 5.0
        x0 = 2.0
        dx = 1.e-3
        Df_x = F.ApproximateJacobian(f, x0, dx)
        self.assertEqual(Df_x.shape, (1,1))
        self.assertAlmostEqual(Df_x, slope)

    def testApproxJacobian2(self):
        A = N.matrix("1. 2.; 3. 4.")
        def f(x):
            return A * x
        x0 = N.matrix("5; 6")
        dx = 1.e-6
        Df_x = F.ApproximateJacobian(f, x0, dx)
        self.assertEqual(Df_x.shape, (2,2))
        N.testing.assert_array_almost_equal(Df_x, A)

    def testPolynomial(self):
        # p(x) = x^2 + 2x + 3
        p = F.Polynomial([1, 2, 3])
        for x in N.linspace(-2,2,11):
            self.assertEqual(p(x), x**2 + 2*x + 3)

if __name__ == '__main__':
    unittest.main()
```