Software Testing Project

Testing Numpy

Group 22
Aliaksandra Kupreyeva,
Anton Gildebrand,
Egemen Yiğit Kömürcü,
Sotirios Chatzigeorgiou,
Tshering Gyeltshen

OUTLINE

Numpy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

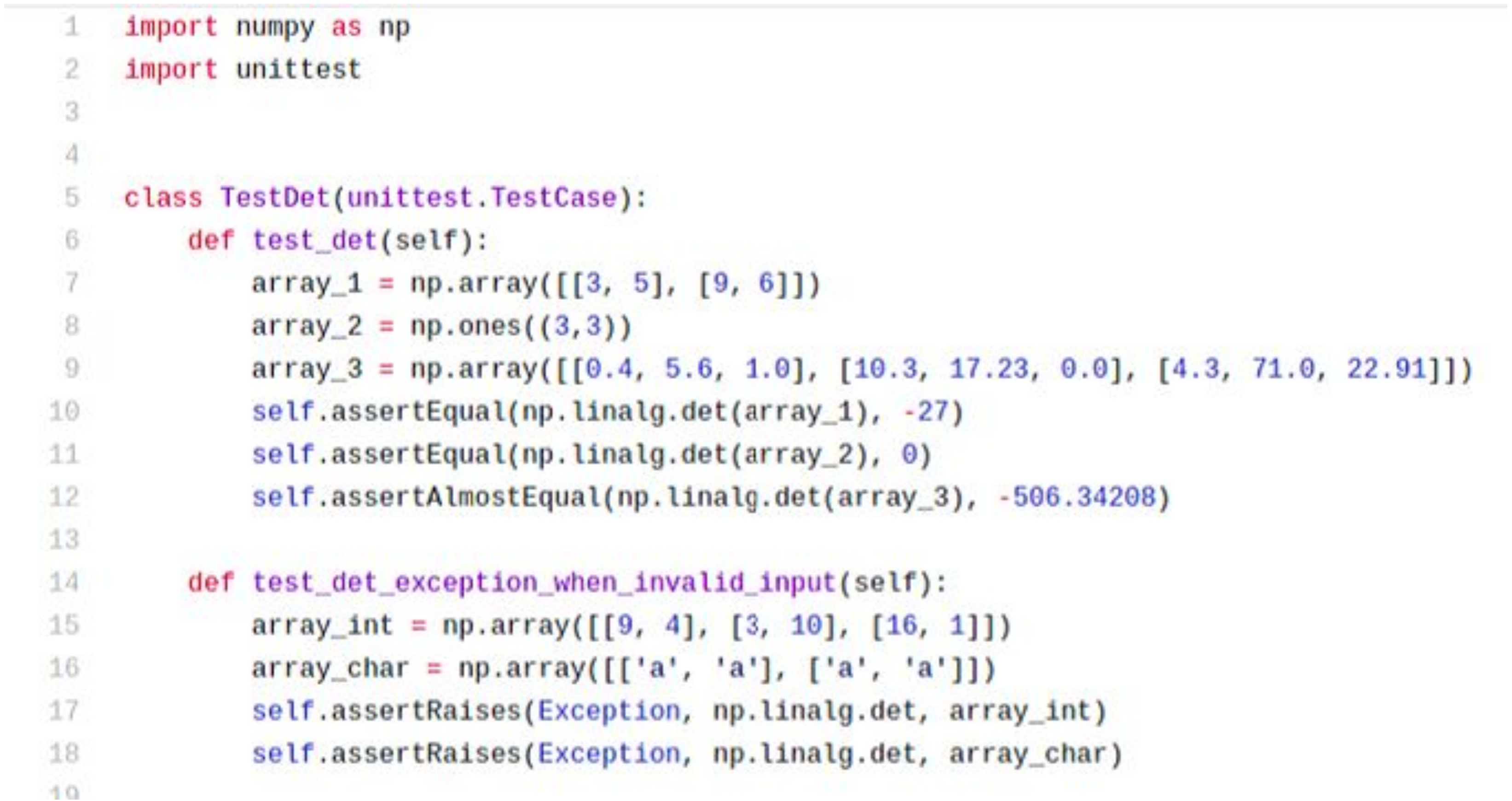
BLACK BOX TESTING

- **Searching:** argmax, argmin, where
- **Linear Algebra:** det, dot, norm
- **Ndarray:** dtype, shape, indexing
- **Sorting:** heap, quick, merge, stable
- **Statistics:** median, average, std, var

For example:

- The **det** function of the Linear Algebra submodule takes a matrix and returns it's determinant
- The **median** function of the Statistics submodule takes an array and returns the median value of that array
- The **shape** function of the Ndarray submodule returns the dimensions of a matrix

TEST CASES



```python
import numpy as np
import unittest


class TestDet(unittest.TestCase):
    def test_det(self):
        array_1 = np.array([[3, 5], [9, 6]])
        array_2 = np.ones((3,3))
        array_3 = np.array([[0.4, 5.6, 1.0], [10.3, 17.23, 0.0], [4.3, 71.0, 22.91]])
        self.assertEqual(np.linalg.det(array_1), -27)
        self.assertEqual(np.linalg.det(array_2), 0)
        self.assertAlmostEqual(np.linalg.det(array_3), -506.34208)

    def test_det_exception_when_invalid_input(self):
        array_int = np.array([[9, 4], [3, 10], [16, 1]])
        array_char = np.array([['a', 'a'], ['a', 'a']])
        self.assertRaises(Exception, np.linalg.det, array_int)
        self.assertRaises(Exception, np.linalg.det, array_char)
```

WHITE BOX TESTING

- We are testing the *polynomial* submodule which provides functions for dealing with polynomial series
- One function in the polynomial submodule is the **polyval** function, which evaluates a polynomial for a given value of *x*:

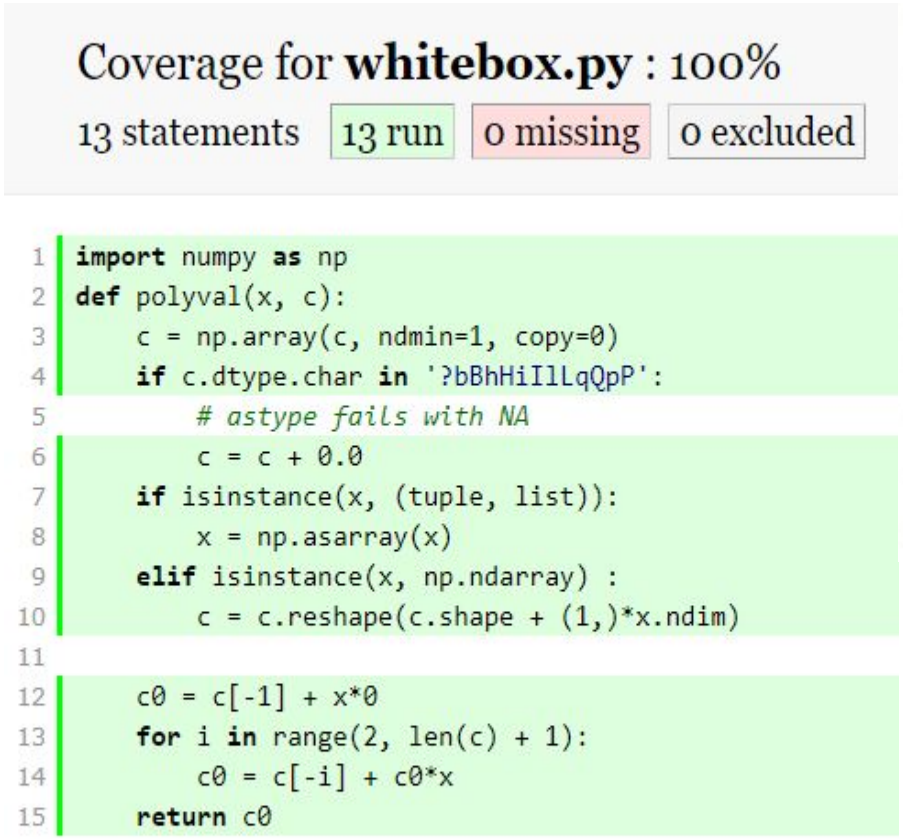
```python
import numpy as np

print('(3x^2 + 2x + 1)(2)= ' + str(np.polyval(2,[1, 2, 3])))
#prints (3x^2 + 2x + 1)(2)= 17
```

- Interesting piece of code found in the implementation of **polyval**:

```python
    if c.dtype.char in '?bBhHiIlLqQpP':
        # astype fails with NA
        c = c + 0.0
```

TEST CASES

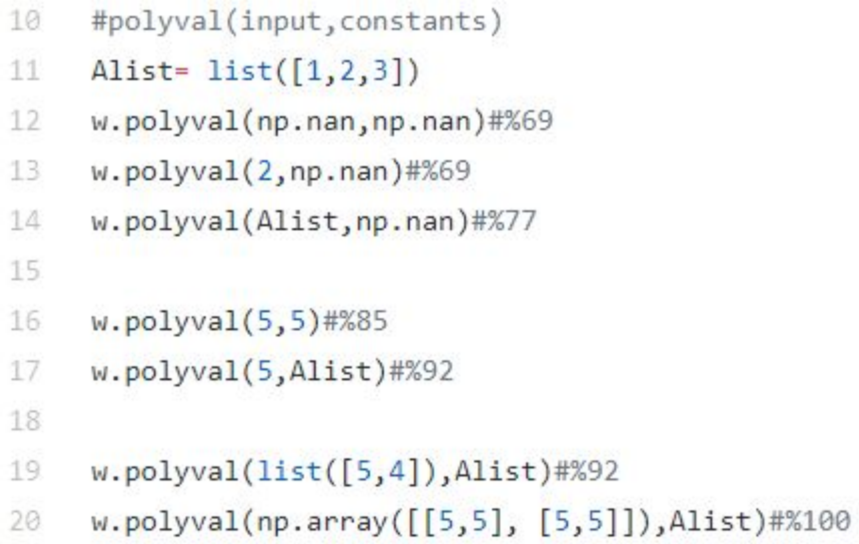


Coverage for **whitebox.py** : 100%

13 statements | 13 run | 0 missing | 0 excluded

```python
import numpy as np
def polyval(x, c):
    c = np.array(c, ndmin=1, copy=0)
    if c.dtype.char in '?bBhHiIlLqQpP':
        # astype fails with NA
        c = c + 0.0
    if isinstance(x, (tuple, list)):
        x = np.asarray(x)
    elif isinstance(x, np.ndarray) :
        c = c.reshape(c.shape + (1,)*x.ndim)

    c0 = c[-1] + x*0
    for i in range(2, len(c) + 1):
        c0 = c[-i] + c0*x
    return c0
```



```python
#polyval(input,constants)
Alist= list([1,2,3])
w.polyval(np.nan,np.nan)#%69
w.polyval(2,np.nan)#%69
w.polyval(Alist,np.nan)#%77

w.polyval(5,5)#%85
w.polyval(5,Alist)#%92

w.polyval(list([5,4]),Alist)#%92
w.polyval(np.array([[5,5], [5,5]]),Alist)#%100
```

CONCLUSION

- NumPy: Explore the capabilities of NumPy through software testing
- Black box testing: Opportunity to expand our knowledge of Unit testing framework
- White box testing: Opportunity to deal with Coverage API

FUTURE WORK

- Implement white box testing:
  - **Polyval** function with expansion of other polyval related functions
  - A more complicated function
- Construct a control flow graph using methods of the white box testing field

REFERENCE

[1] N. developers, "NumPy," 2019. [Online]. Available: https://numpy.org/. [Accessed 4 November 2019].

THANK YOU FOR LISTENING

Any questions?