Elements from NumPy arrays can be selected using four methods: scalar selection, slicing, numerical (or list-of-locations) indexing and logical (or Boolean) indexing.

Numerical indexing uses lists or arrays of locations to select elements while logical indexing uses arrays containing Boolean values to select elements.

```
>>> x = 10 * arange(5.0)
>>> x[[0]] # List with 1 element
array([ 0.])
>>> x[[0,2,1]] # List
array([ 0., 20., 10.])
>>> sel = array([4,2,3,1,4,4]) # Array with repetition
>>> x[sel]
array([ 40., 20., 30., 10., 40., 40.])
>>> sel = array([[4,2],[3,1]]) # 2 by 2 array
>>> x[sel] # Selection has same size as sel
array([[ 40., 20.],
[ 30., 10.]])
>>> sel = array([0.0,1]) # Floating point data
>>> x[sel] # Error
IndexError: arrays used as indices must be of integer (or boolean) type
>>> x[sel.astype(int)] # No error
array([ 10., 20.])
>>> x[0] # Scalar selection, not numerical indexing
1.0
```

```
>>> x = reshape(arange(10.0), (2,5))
>>> x
array([[ 0., 1., 2., 3., 4.],
[ 5., 6., 7., 8., 9.]])
>>> sel = array([0,1])
>>> x[sel,sel] # 1-dim arrays, no broadcasting
array([ 0., 6.])
>>> x[sel, sel+1]
array([ 1., 7.])
>>> sel_row = array([[0,0],[1,1]])
>>> sel_col = array([[0,1],[0,1]])
>>> x[sel_row,sel_col] # 2 by 2, no broadcasting
array([[ 0., 1.],
[ 5., 6.]])
>>> sel_row = array([[0],[1]])
>>> sel_col = array([[0,1]])
>>> x[sel_row,sel_col] # 2 by 1 and 1 by 2 - difference shapes, broadcasted as 2 by 2
array([[ 0., 1.],
[ 5., 6.]])
```

0.1 Mixing Numerical Indexing with Scalar Selection

NumPy permits using difference types of indexing in the same expression. Mixing numerical indexing with scalar selection is trivial since any scalar can be broadcast to any array shape.

```
>>> x = array([[1,2],[3,4]])
>>> sel = x <= 3
>>> indices = nonzero(sel)
>>> indices
(array([0, 0, 1], dtype=int64), array([0, 1, 0], dtype=int64))
```

```
>>> x = randn(3)
>>> x
array([-0.5910316 , 0.51475905, 0.68231135])
>>> argwhere(x<0.6)
array([[0],
[1]], dtype=int64)
>>> argwhere(x<-10.0) # Empty array
array([], shape=(0L, 1L), dtype=int64)
>>> x = randn(3,2)
>>> x
array([[ 0.72945913, 1.2135989 ],
[ 0.74005449, -1.60231553],
[ 0.16862077, 1.0589899 ]])
>>> argwhere(x<0)
array([[1, 1]], dtype=int64)
>>> argwhere(x<1)
array([[0, 0],
[1, 0],
[1, 1],
[2, 0]], dtype=int64)
```