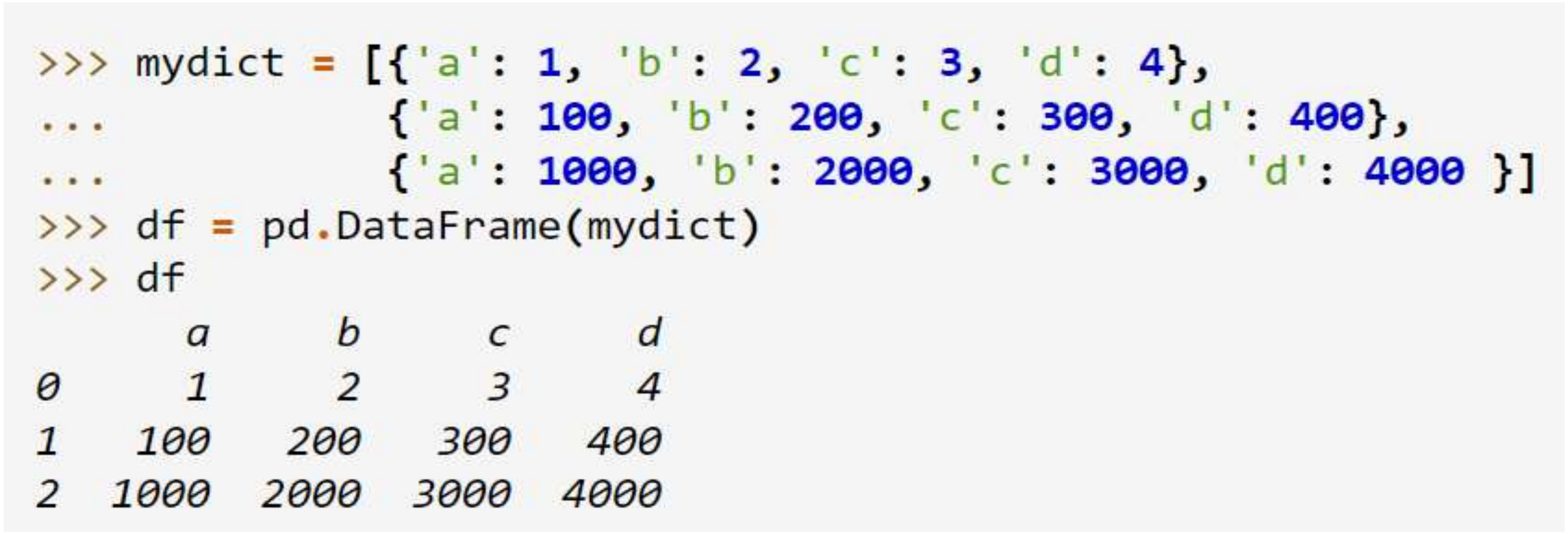
Pandas.DataFrame.iloc and Pandas.DataFrame.loc

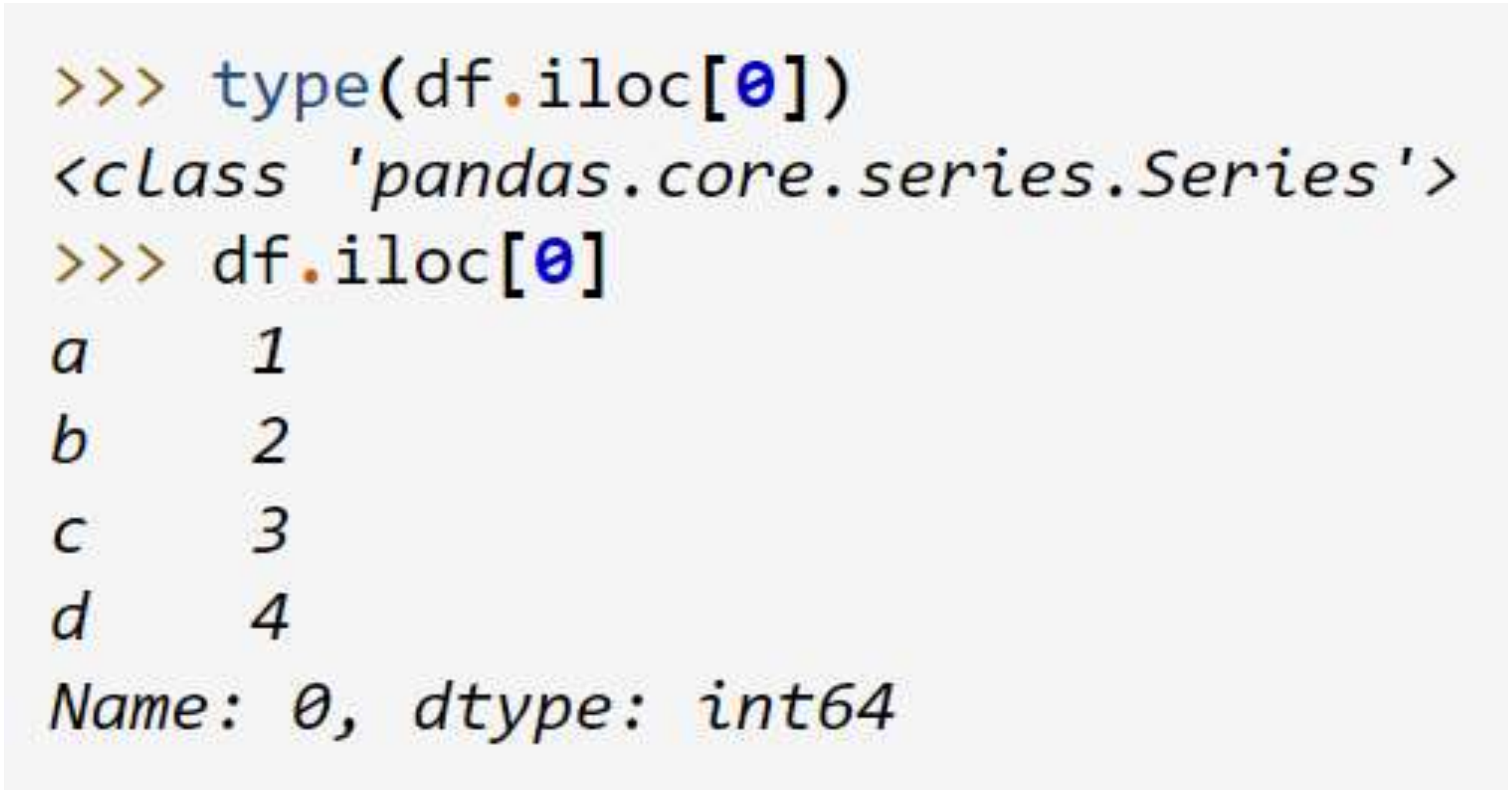
Pandas.DataFrame.iloc

Purely integer-location based indexing for selection by position.



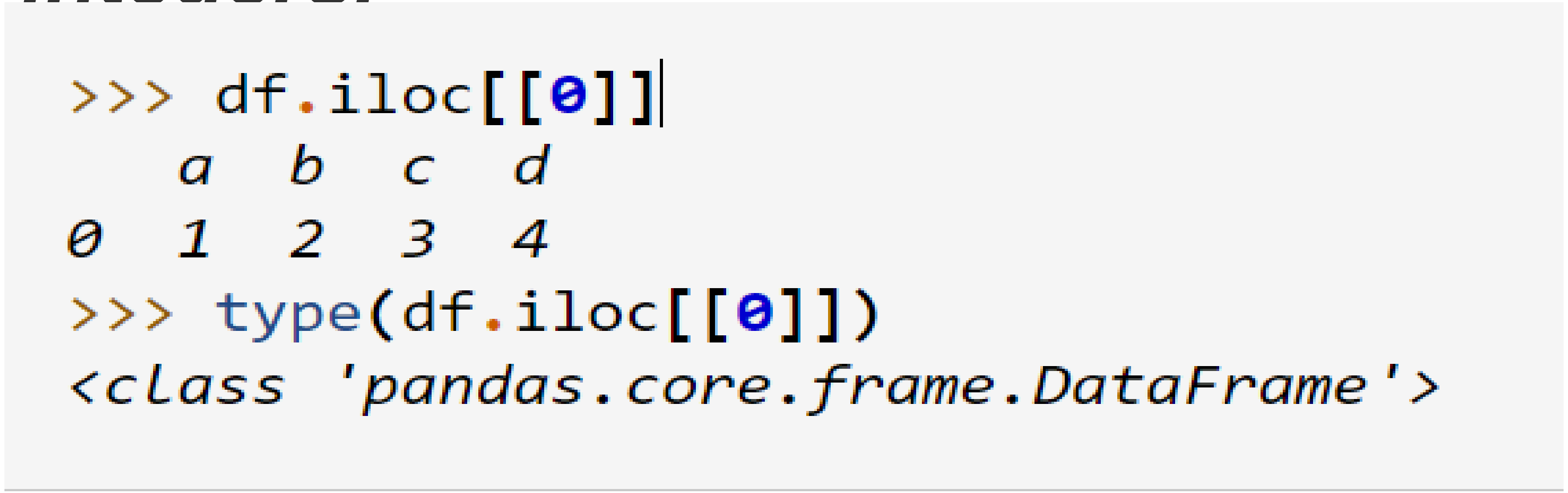
```
>>> mydict = [{'a': 1, 'b': 2, 'c': 3, 'd': 4},
...           {'a': 100, 'b': 200, 'c': 300, 'd': 400},
...           {'a': 1000, 'b': 2000, 'c': 3000, 'd': 4000 }]
>>> df = pd.DataFrame(mydict)
>>> df
      a     b     c     d
0     1     2     3     4
1   100   200   300   400
2  1000  2000  3000  4000
```

Indexing the rows
With a scalar integer

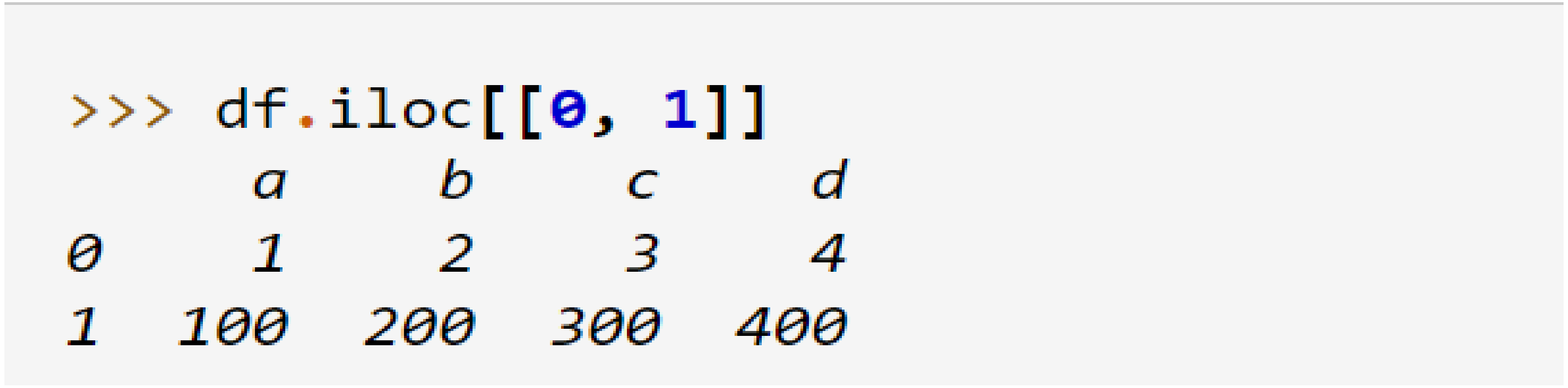


```
>>> type(df.iloc[0])
<class 'pandas.core.series.Series'>
>>> df.iloc[0]
a    1
b    2
c    3
d    4
Name: 0, dtype: int64
```

With a list of integers.

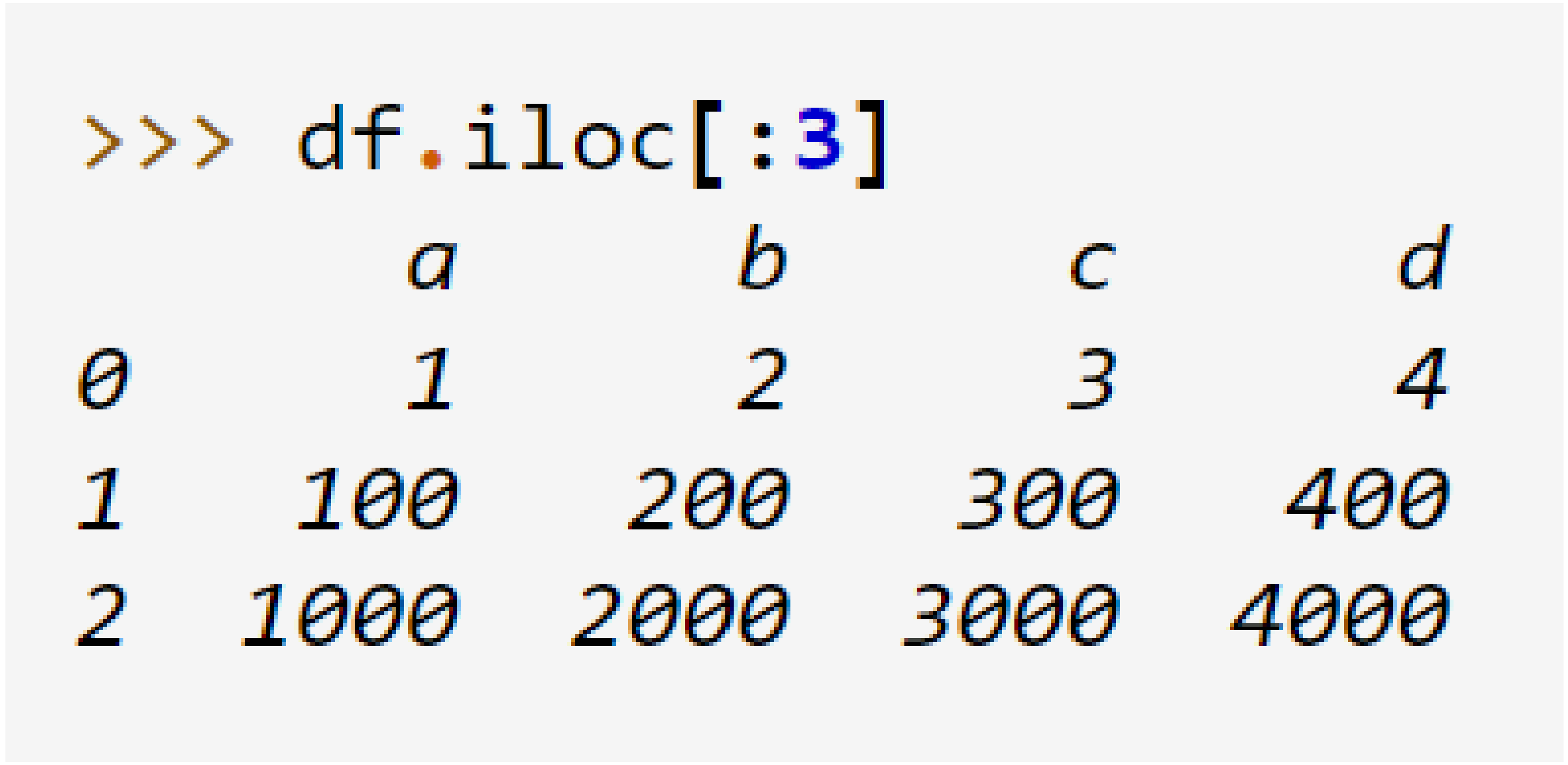


```
>>> df.iloc[[0]]
   a  b  c  d
0  1  2  3  4
>>> type(df.iloc[[0]])
<class 'pandas.core.frame.DataFrame'>
```



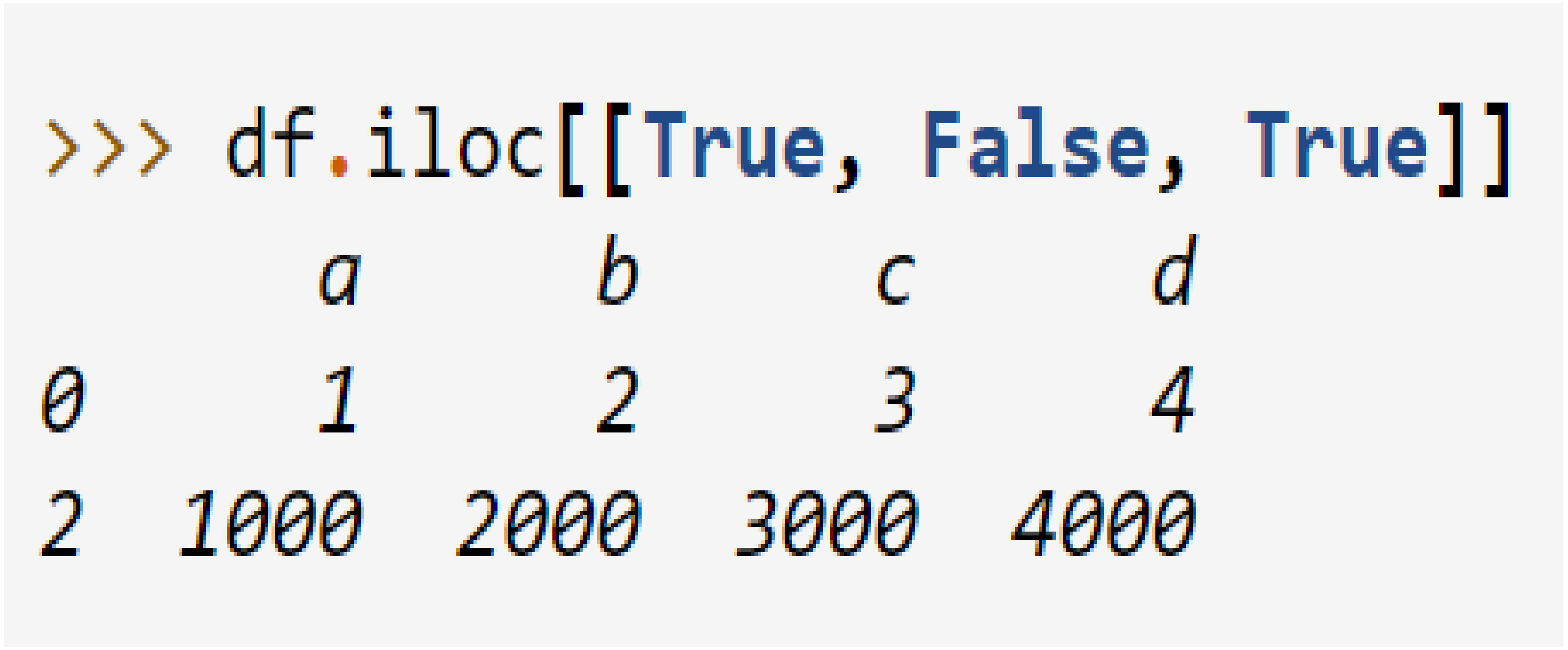
```
>>> df.iloc[[0, 1]]
     a    b    c    d
0    1    2    3    4
1  100  200  300  400
```

With a Slice object



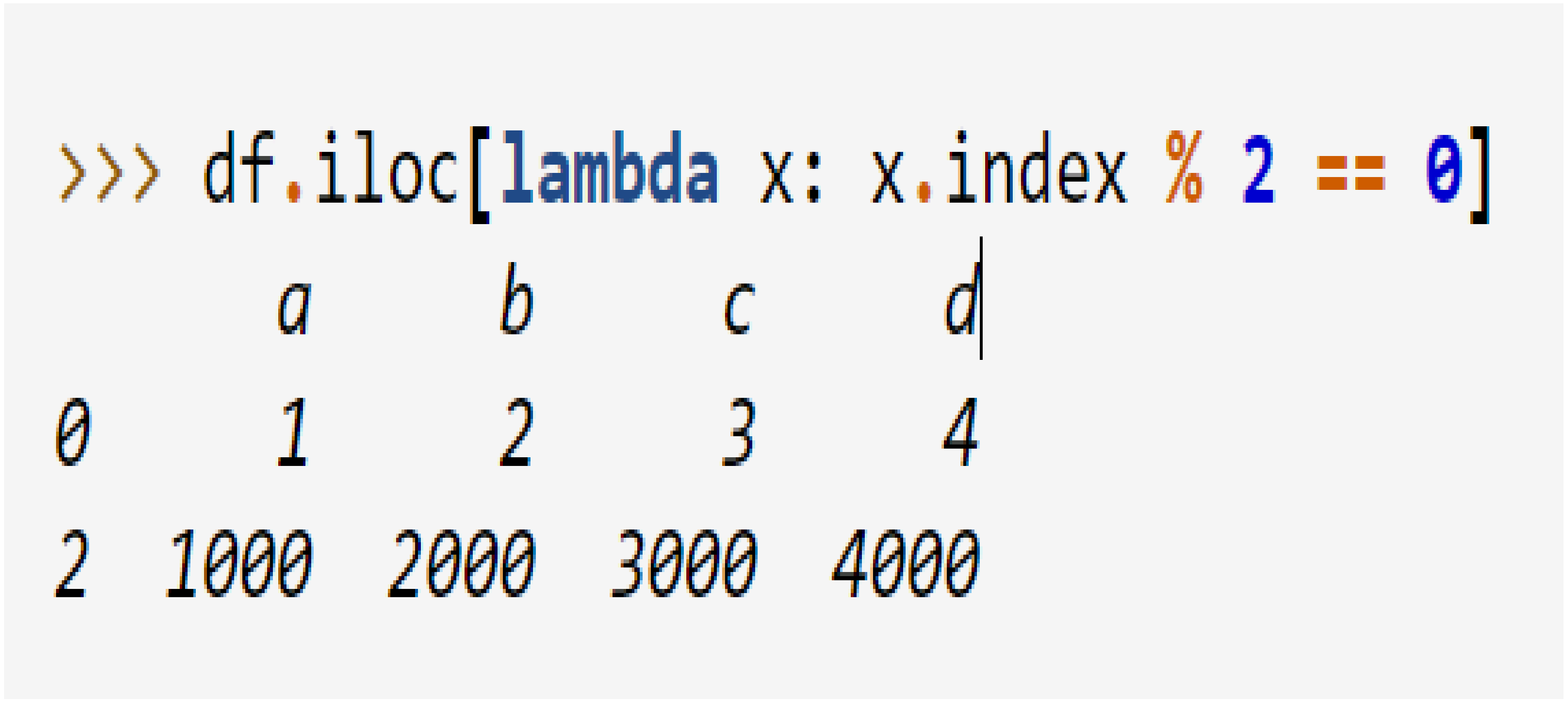
```
>>> df.iloc[:3]
      a     b     c     d
0     1     2     3     4
1   100   200   300   400
2  1000  2000  3000  4000
```

With a boolean mask the same length as the index.



```
>>> df.iloc[[True, False, True]]
      a     b     c     d
0     1     2     3     4
2  1000  2000  3000  4000
```

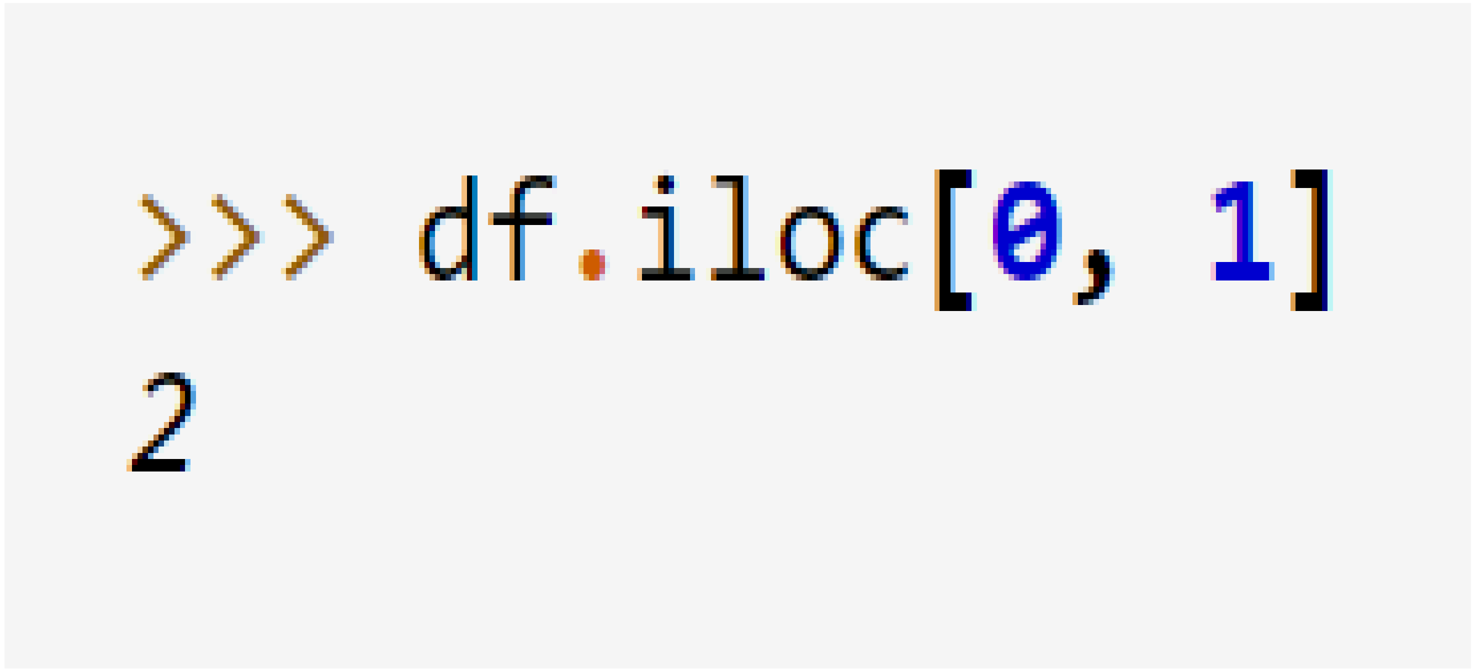
With a callable, useful in method chains. The x passed to the **lambda** ***is the DataFrame being sliced. This selects the rows whose index label even.***



```
>>> df.iloc[lambda x: x.index % 2 == 0]
      a     b     c     d
0     1     2     3     4
2  1000  2000  3000  4000
```

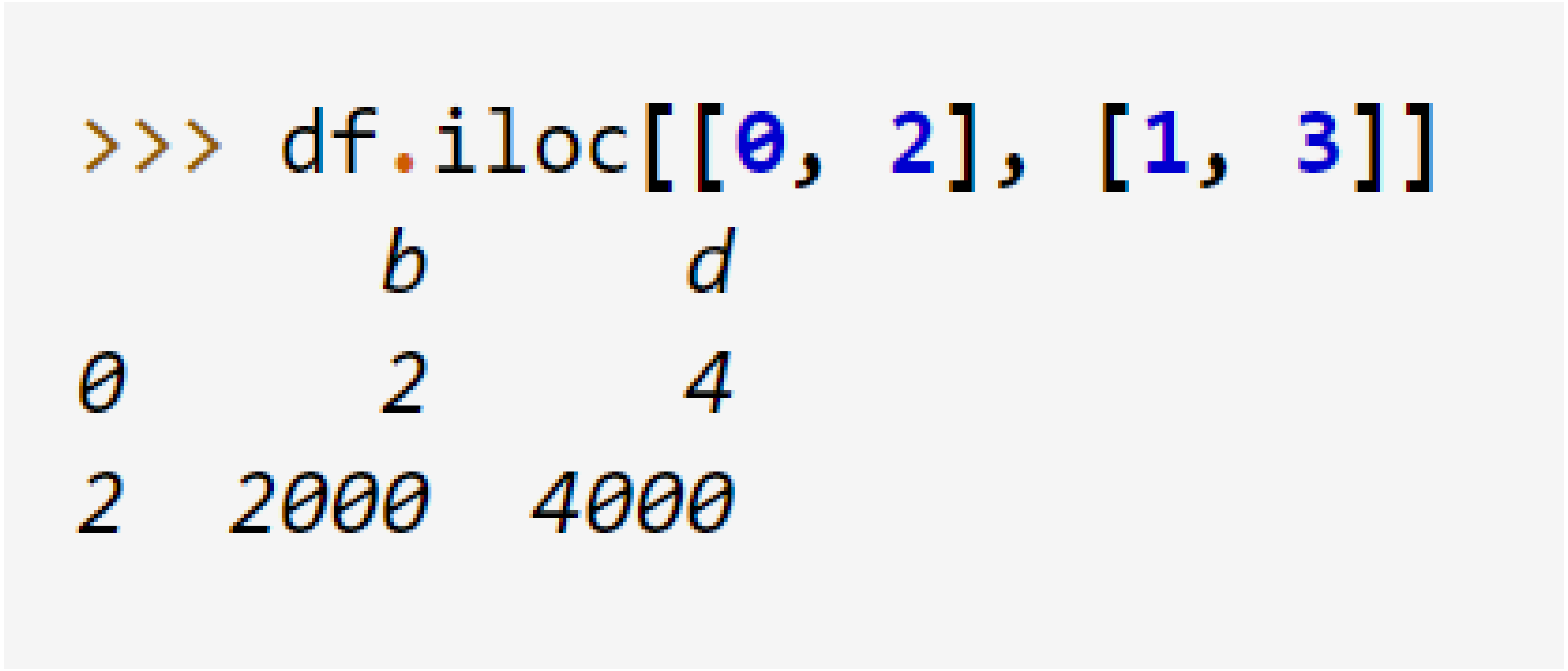
Indexing both axes

You can mix the indexer types for the index and columns.Use : to select the entire axis.
With Scalar integers.



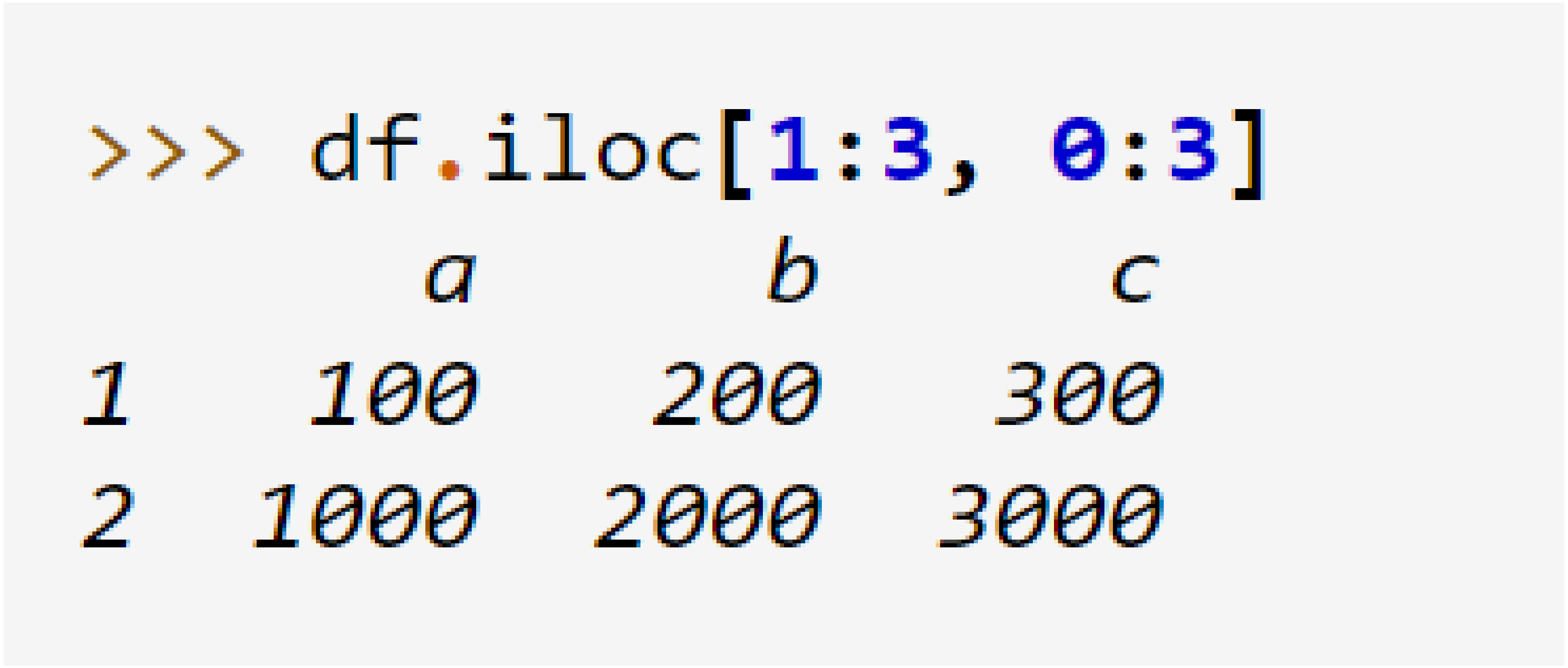
```
>>> df.iloc[0, 1]
2
```

With lists of integers.



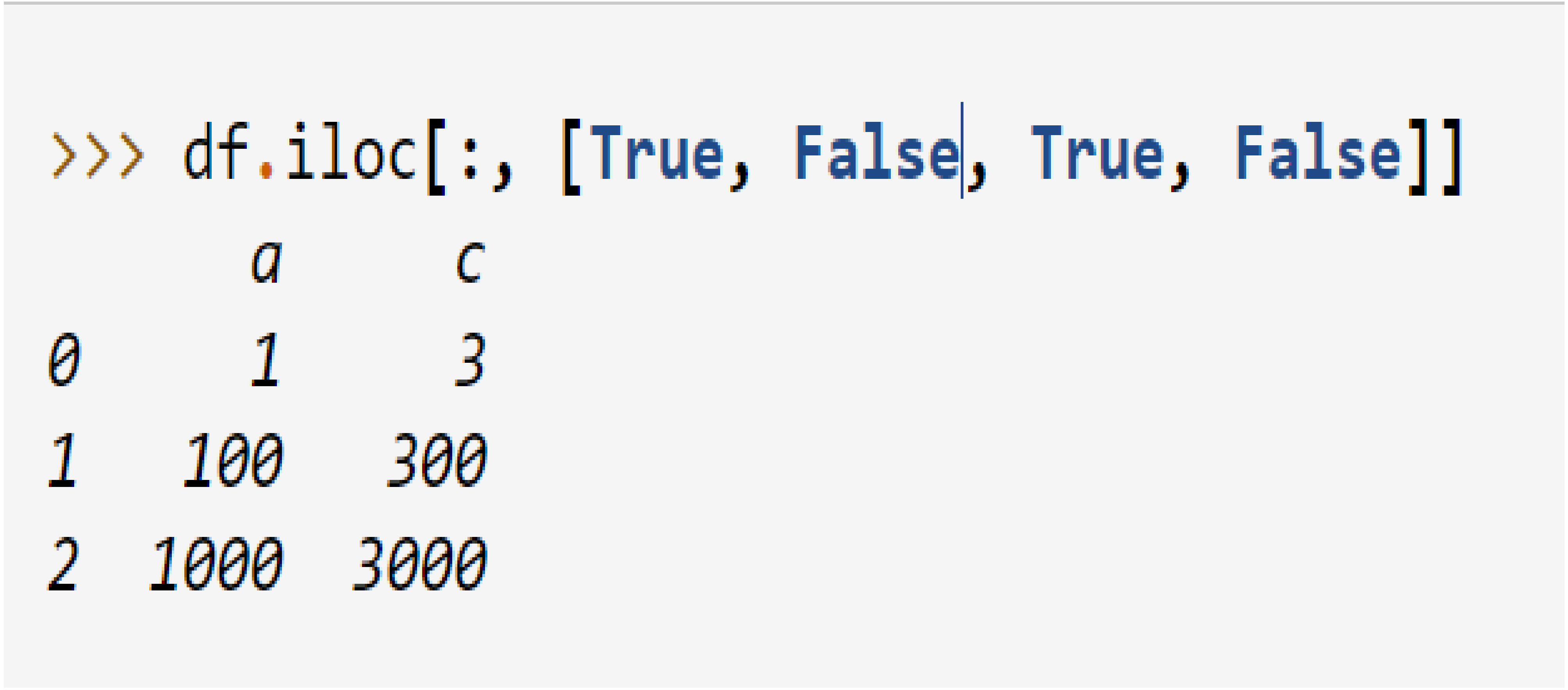
```
>>> df.iloc[[0, 2], [1, 3]]
      b     d
0     2     4
2  2000  4000
```

With slice objects.



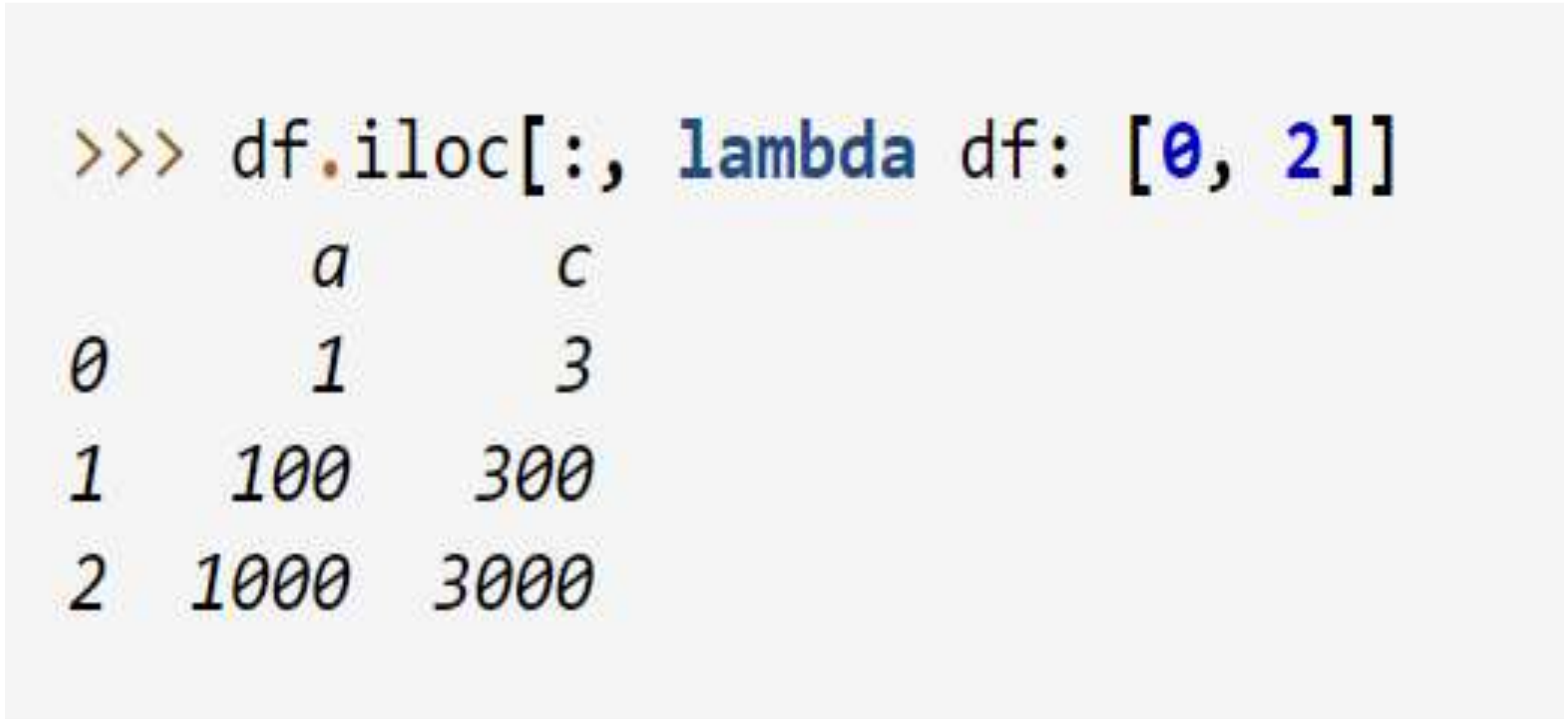
```
>>> df.iloc[1:3, 0:3]
      a     b     c
1   100   200   300
2  1000  2000  3000
```

With a boolean array whose length matches the columns.



```
>>> df.iloc[:, [True, False, True, False]]
      a     c
0     1     3
1   100   300
2  1000  3000
```

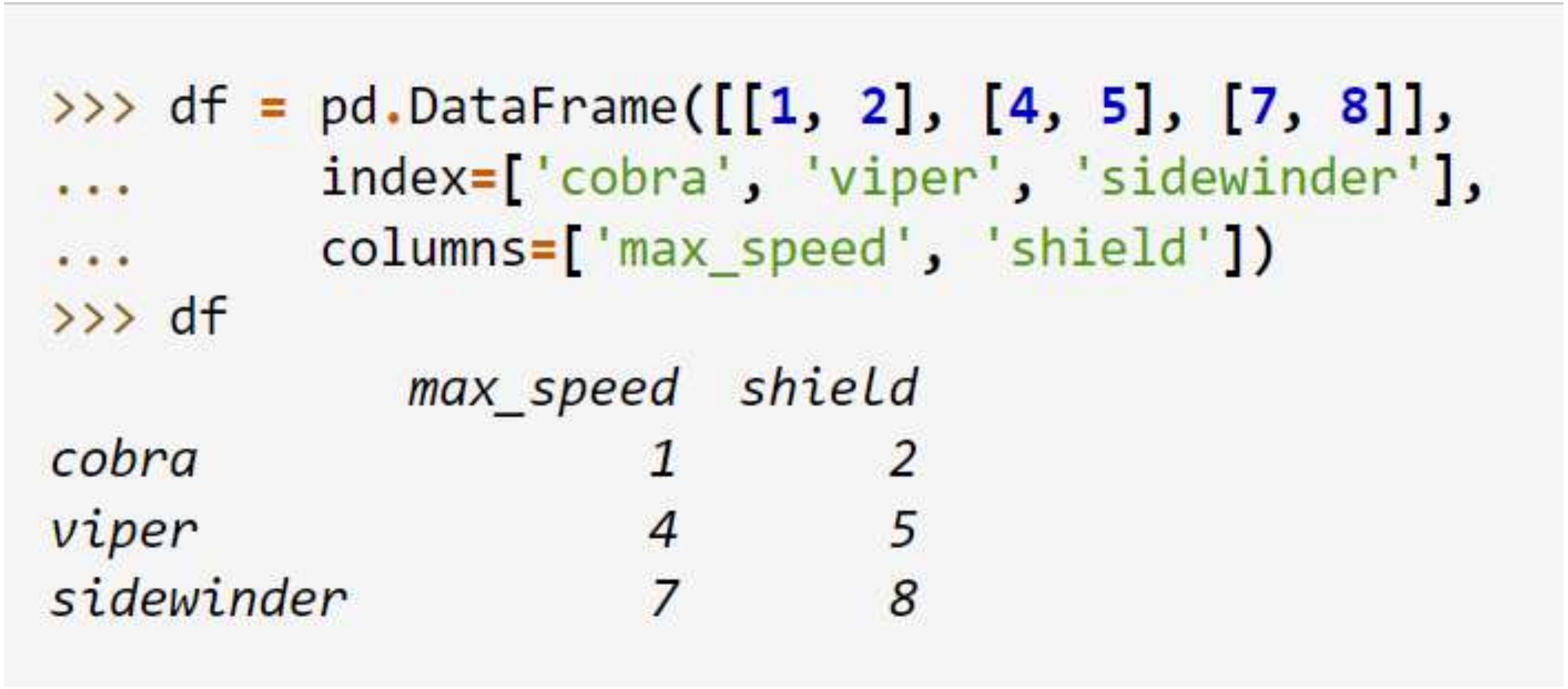
With a callable function that expects the Series or DataFrame.



```
>>> df.iloc[:, lambda df: [0, 2]]
      a     c
0     1     3
1   100   300
2  1000  3000
```

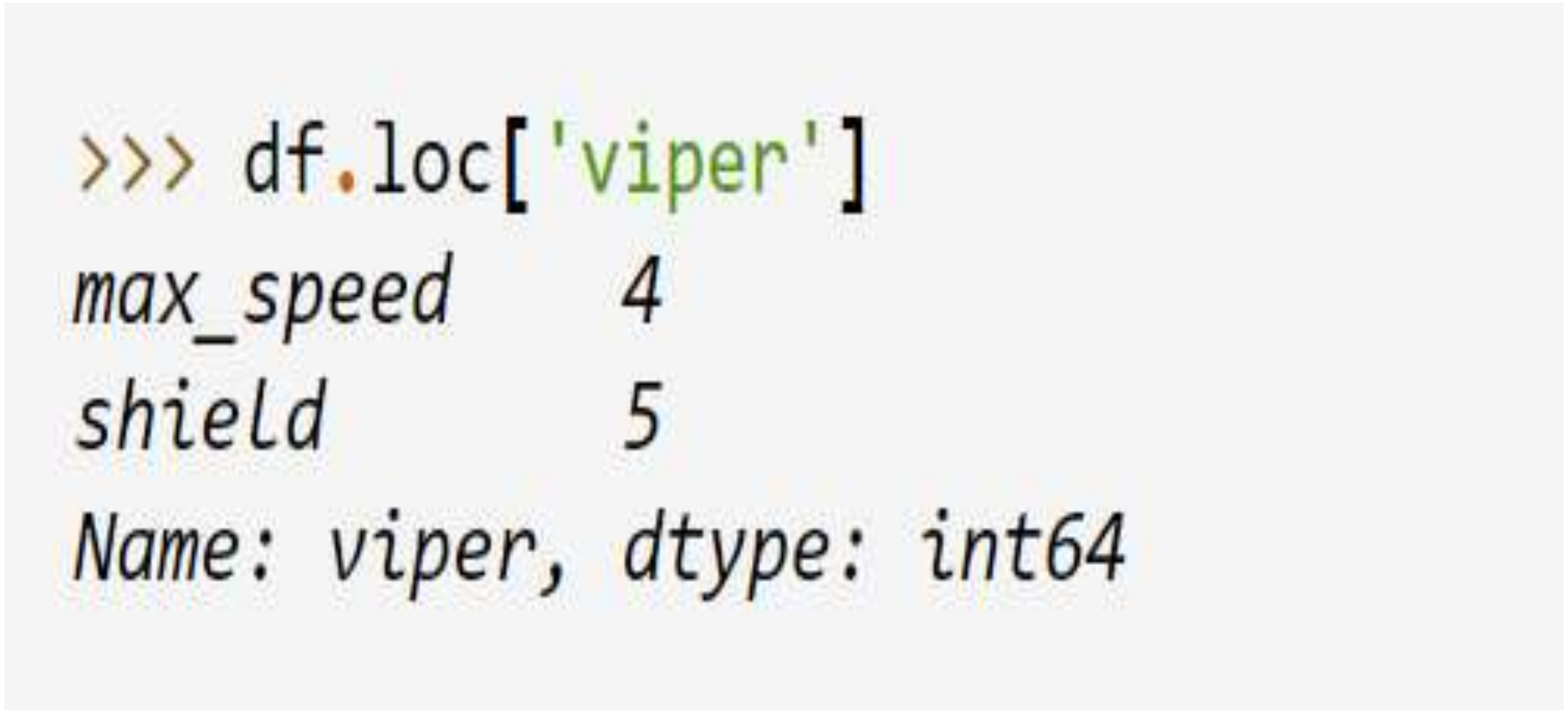
Pandas.DataFrame.loc

Access a group of rows and columns by label(s) or a boolean array.



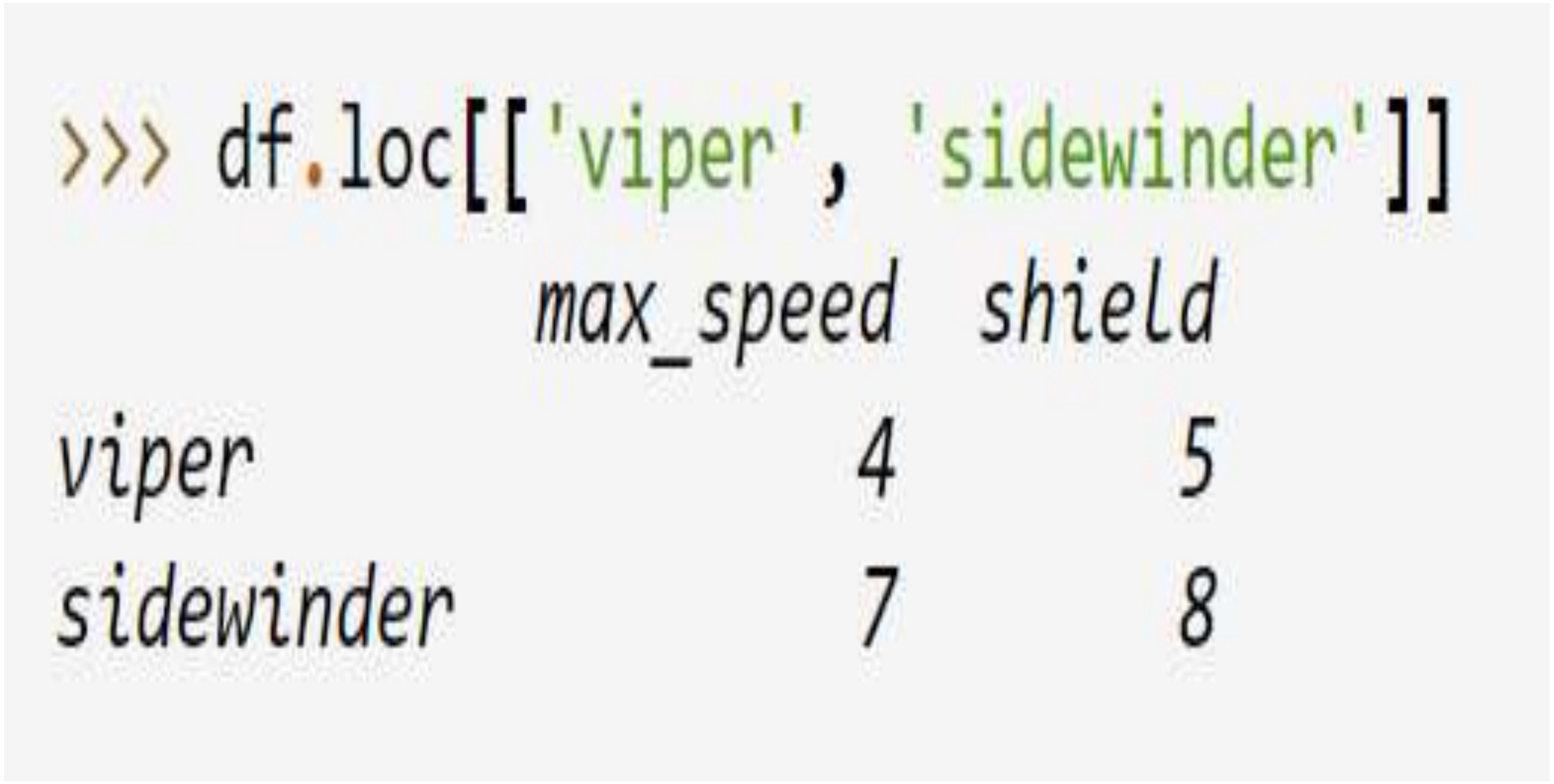
```
>>> df = pd.DataFrame([[1, 2], [4, 5], [7, 8]],
...      index=['cobra', 'viper', 'sidewinder'],
...      columns=['max_speed', 'shield'])
>>> df
            max_speed  shield
cobra               1       2
viper               4       5
sidewinder          7       8
```

Single label. Note this returns the row as a Series.



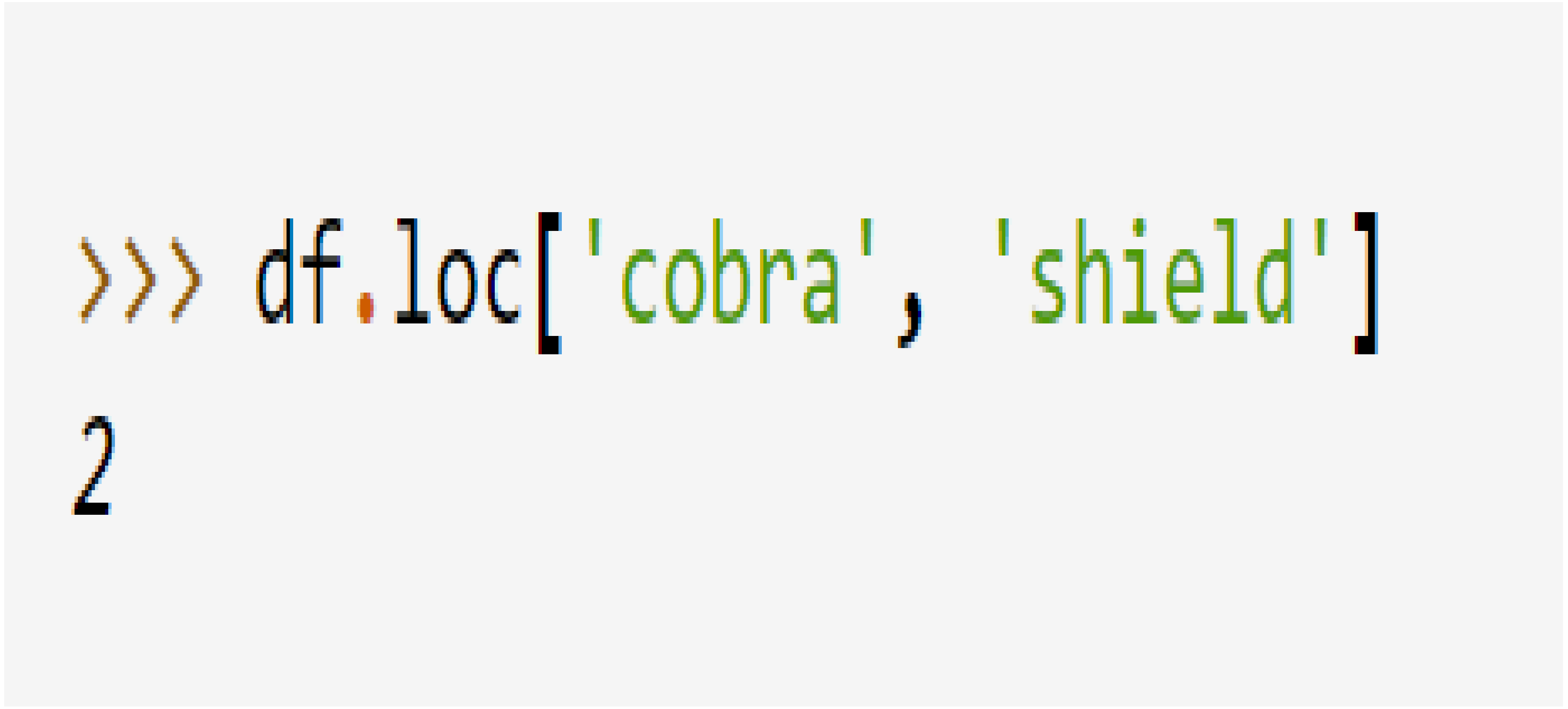
```
>>> df.loc['viper']
max_speed    4
shield       5
Name: viper, dtype: int64
```

List of labels. Note using **[[]]** *returns a DataFrame.*



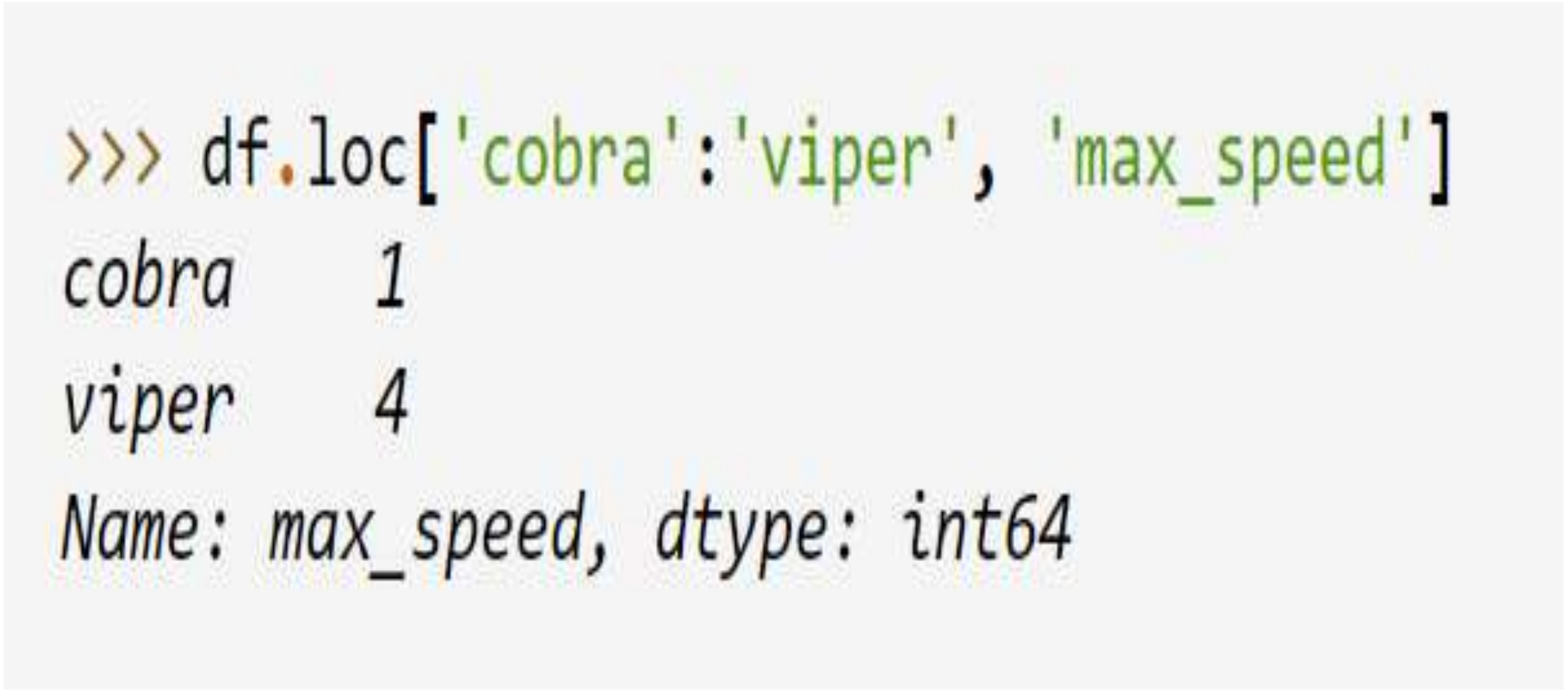
```
>>> df.loc[['viper', 'sidewinder']]
            max_speed  shield
viper               4       5
sidewinder          7       8
```

Single label for row and column



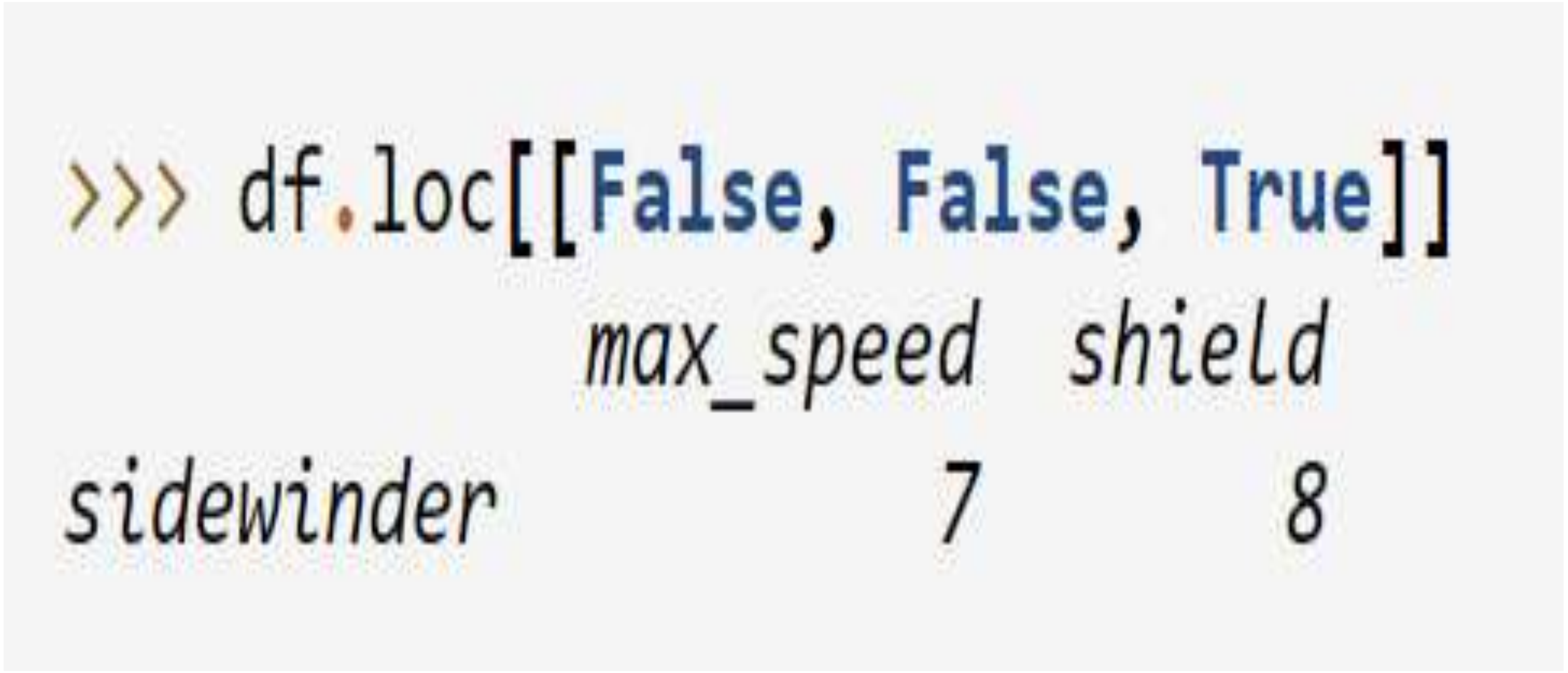
```
>>> df.loc['cobra', 'shield']
2
```

Slice with labels for row and single label for column. As mentioned above, note that both the start and stop of the slice are included.



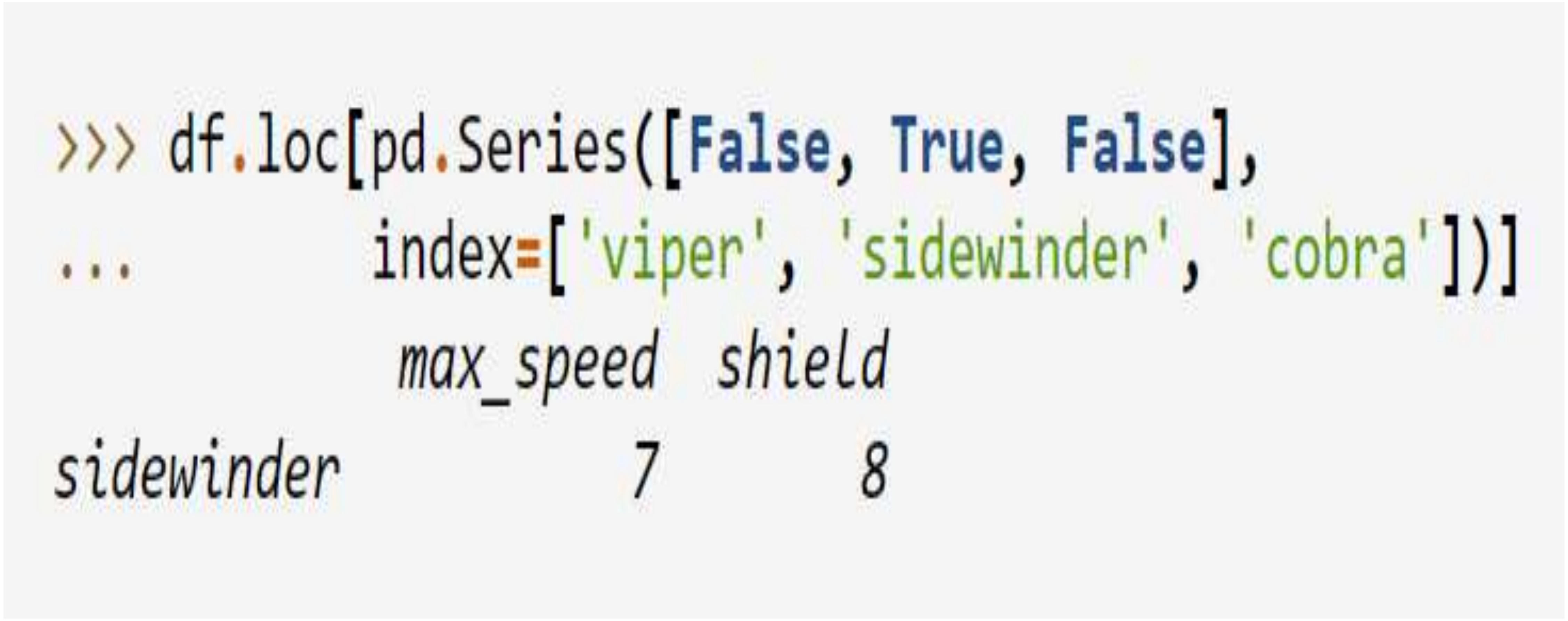
```
>>> df.loc['cobra':'viper', 'max_speed']
cobra     1
viper     4
Name: max_speed, dtype: int64
```

Boolean list with the same length as the row axis



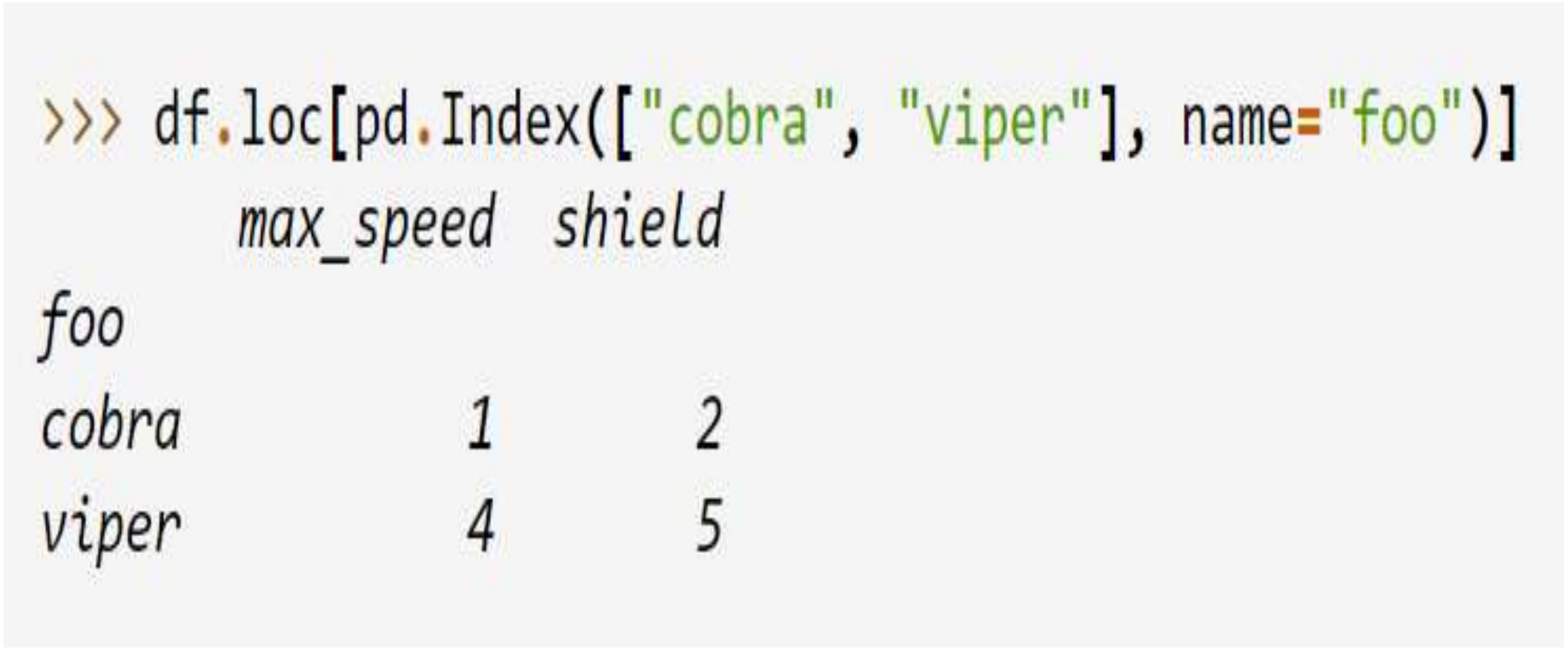
```
>>> df.loc[[False, False, True]]
            max_speed  shield
sidewinder          7       8
```

Alignable boolean Series



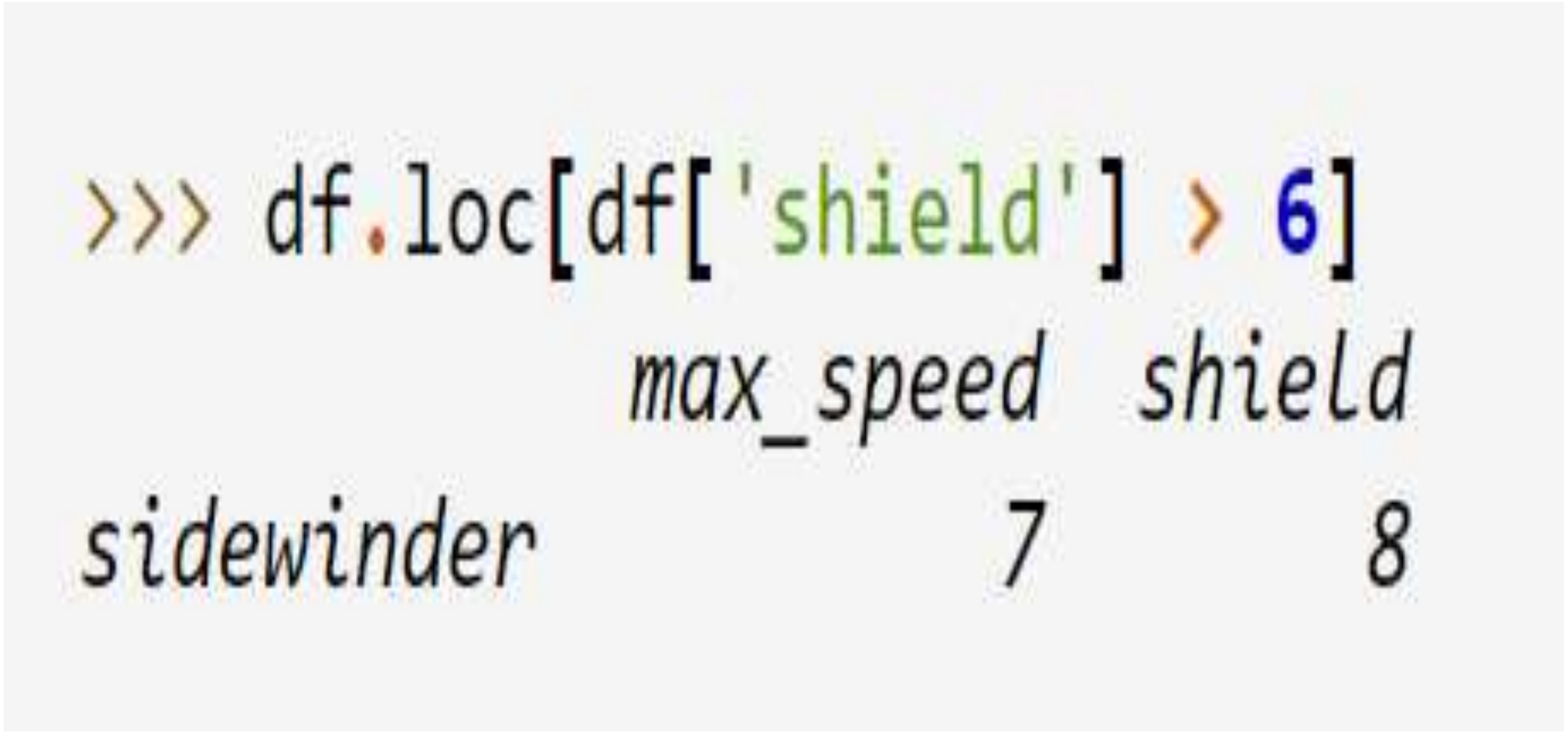
```
>>> df.loc[pd.Series([False, True, False],
...        index=['viper', 'sidewinder', 'cobra'])]
            max_speed  shield
sidewinder          7       8
```

*Index (same behavior as **df.reindex**)*



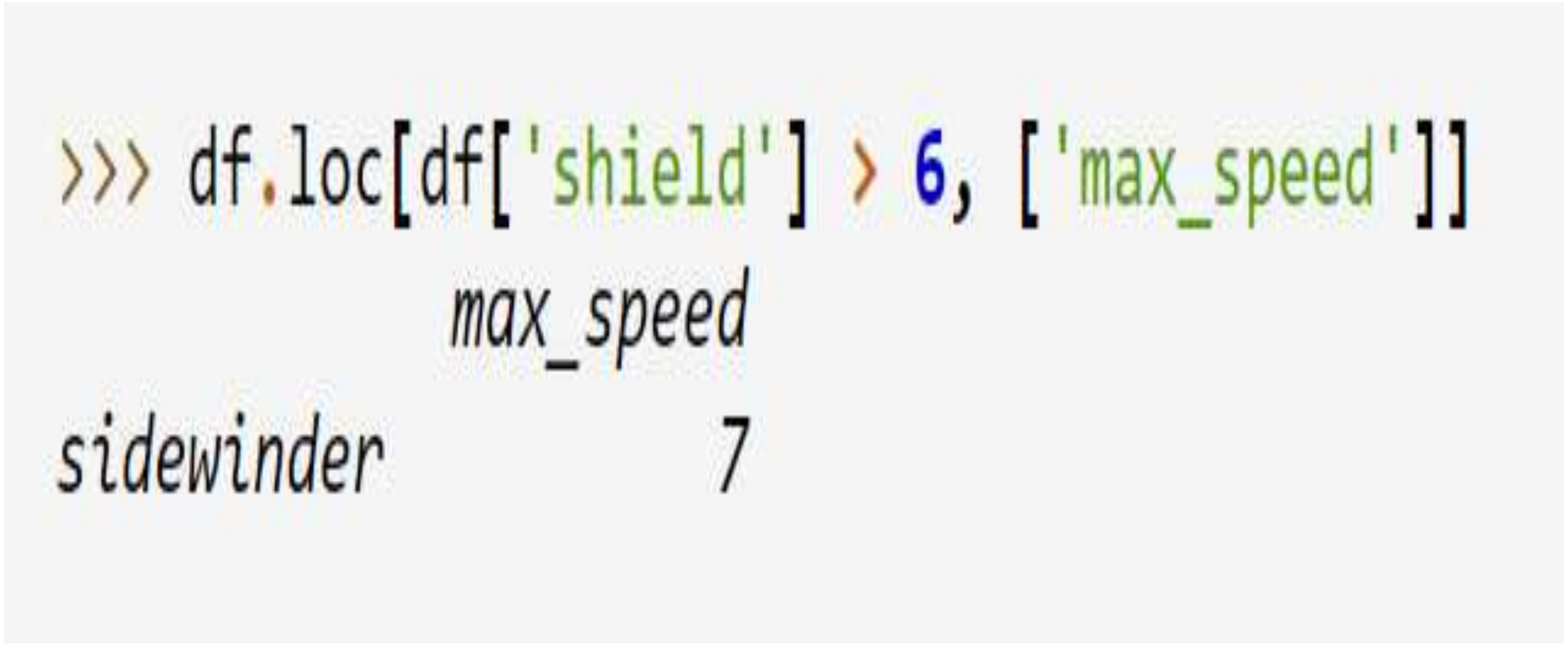
```
>>> df.loc[pd.Index(["cobra", "viper"], name="foo")]
       max_speed  shield
foo
cobra          1       2
viper          4       5
```

Conditional that returns a boolean Series



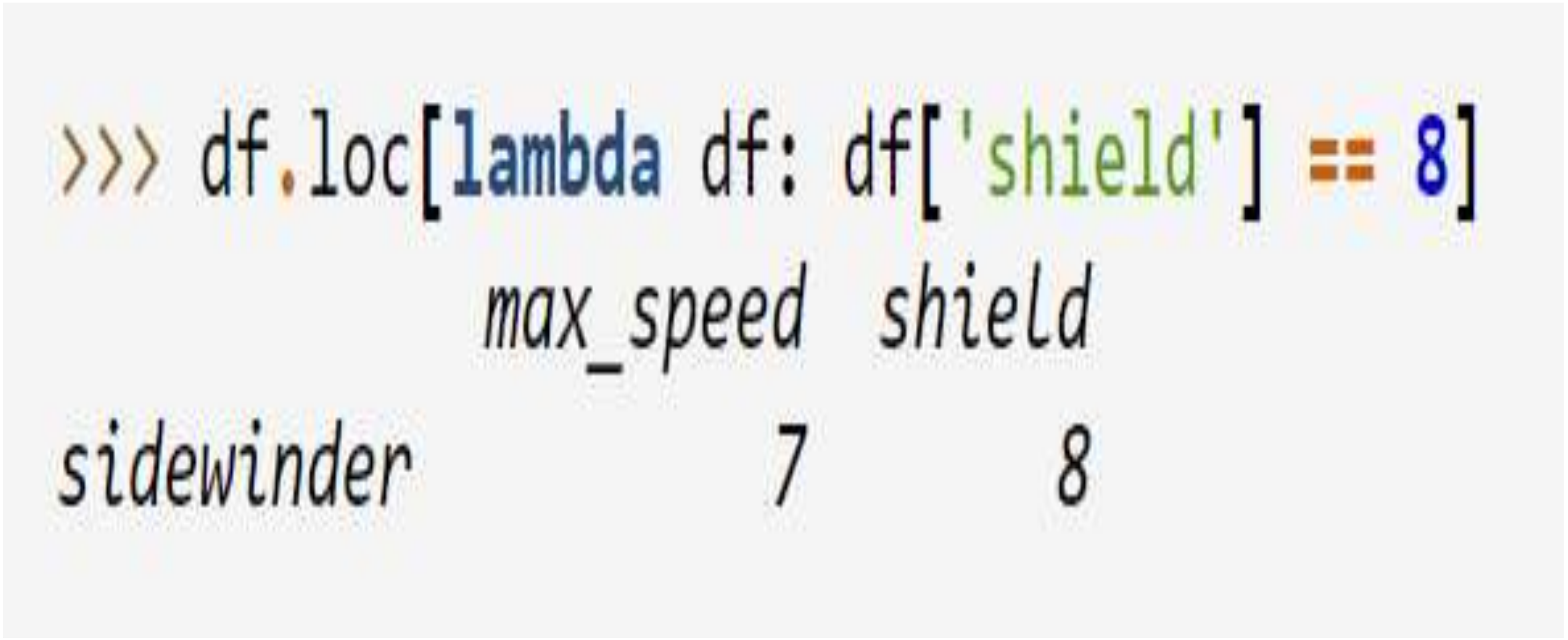
```
>>> df.loc[df['shield'] > 6]
             max_speed  shield
sidewinder           7       8
```

Conditional that returns a boolean Series with column labels specified



```
>>> df.loc[df['shield'] > 6, ['max_speed']]
            max_speed
sidewinder          7
```

Callable that returns a boolean Series



```
>>> df.loc[lambda df: df['shield'] == 8]
            max_speed  shield
sidewinder          7       8
```