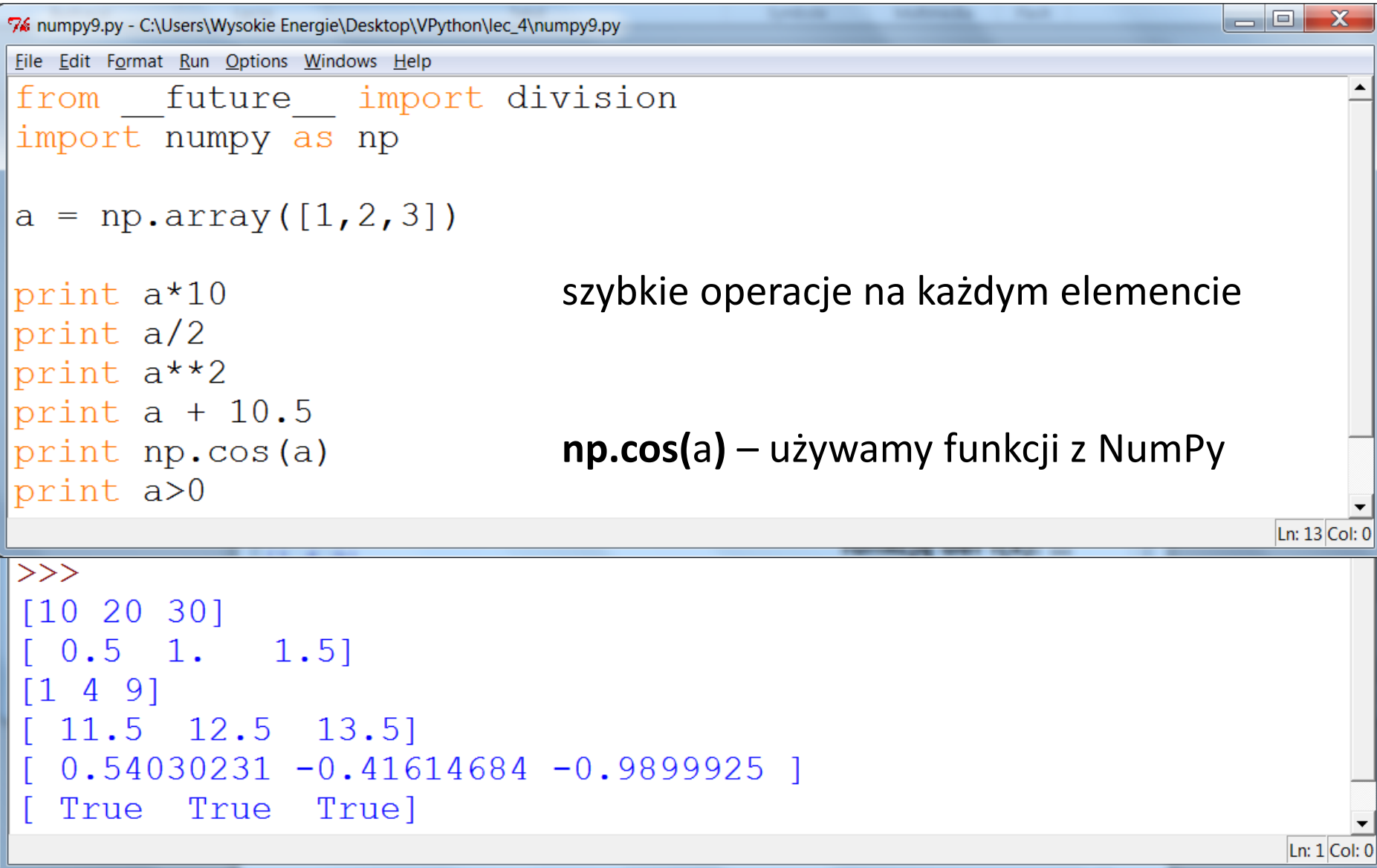


VPython - symulacje fizyczne z grafiką 3D dla każdego

wykład 5

Dr hab. Adam Bzdak

Dlaczego NumPy ?



The image shows a screenshot of a Python IDE window titled 'numpy9.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_4\numpy9.py'. The window contains two panes. The top pane shows the source code for a Python script that imports NumPy and performs various operations on an array. The bottom pane shows the output of the script when executed in a REPL.

```
from __future__ import division
import numpy as np

a = np.array([1,2,3])

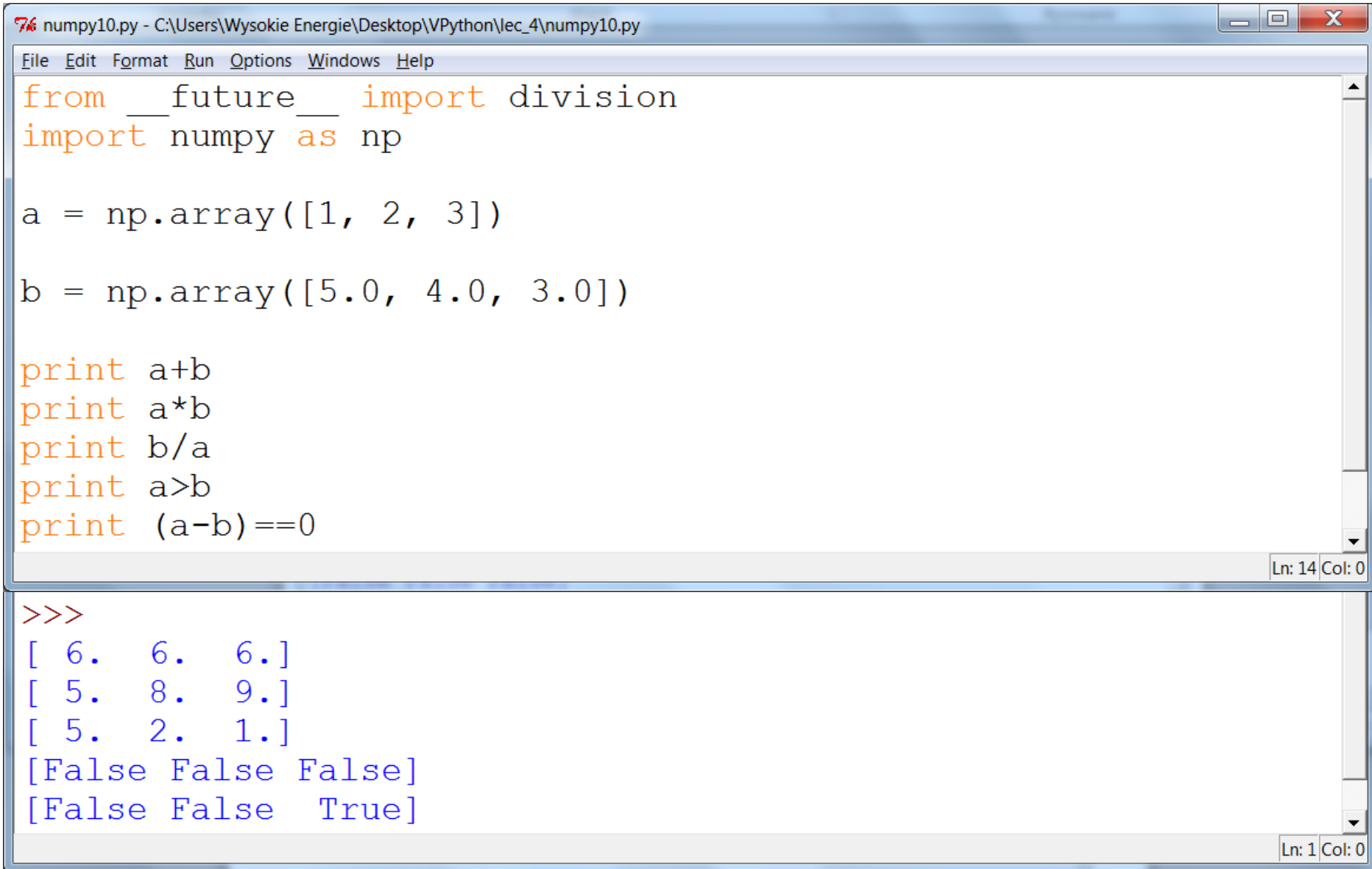
print a*10
print a/2
print a**2
print a + 10.5
print np.cos(a)
print a>0
```

szybkie operacje na każdym elemencie

np.cos(a) – używamy funkcji z NumPy

```
>>>
[10 20 30]
[ 0.5  1.   1.5]
[ 1  4  9]
[ 11.5  12.5  13.5]
[ 0.54030231 -0.41614684 -0.9899925 ]
[ True  True  True]
```

Dlaczego NumPy ?



The image shows a screenshot of a Python IDE window titled 'numpy10.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_4\numpy10.py'. The window contains a Python script that uses NumPy to create two arrays, 'a' and 'b', and perform various operations on them. The script is as follows:

```
from __future__ import division
import numpy as np

a = np.array([1, 2, 3])
b = np.array([5.0, 4.0, 3.0])

print a+b
print a*b
print b/a
print a>b
print (a-b)==0
```

The output of the script is displayed in the console window at the bottom of the IDE. It shows the results of the arithmetic operations and the comparison:

```
>>>
[ 6.  6.  6.]
[ 5.  8.  9.]
[ 5.  2.  1.]
[False False False]
[False False  True]
```

The IDE window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Windows', and 'Help'. The status bar at the bottom right of the editor shows 'Ln: 14 Col: 0' and the status bar at the bottom right of the console shows 'Ln: 1 Col: 0'.

Liczby pseudolosowe

random_1.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\random_1.py

File Edit Format Run Options Windows Help

```
import numpy as np
```

```
print np.random.normal(0, 5, (2, 3))  
print
```

$$\exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

```
print np.random.exponential(0.5, (2, 4))  
print
```

$$\exp(-x/\beta), x > 0$$

```
print np.random.poisson(5, 10)  
print
```

$$\frac{\lambda^n}{n!} e^{-\lambda} \quad \lambda = \langle n \rangle$$

Ln: 14 Col: 0

```
>>>
```

```
[[ -0.53985692 -1.19592883  5.93510607]  
 [ -2.34318102 -1.10001634 -4.00327543]]
```

```
[[ 0.31203411  0.06056116  0.01186506  0.87519643]  
 [ 0.22901289  0.30740062  0.56839502  0.76574675]]
```

```
[ 4  4  3  5 10  4  2  9  7  5]
```

Ln: 1 Col: 0

permutation, choice

random_2.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\random_2.py

File Edit Format Run Options Windows Help

```
import numpy as np

print np.random.permutation([1,10,100,1000])
print np.random.permutation(np.arange(1,11))
print np.random.permutation(['a','b','c'])
print ''

print np.random.choice(['a','b','c','d'], 2)
print np.random.choice(['a','b','c','d'], (2,10))
```

Ln: 11 Col: 0

```
>>>
[ 10   1 100 1000]
[ 5  7  3  6  9  2  1  4 10  8]
['c' 'b' 'a']

['d' 'd']
[['b' 'c' 'a' 'd' 'd' 'b' 'a' 'd' 'a' 'a']
 ['a' 'd' 'b' 'd' 'b' 'd' 'a' 'd' 'd' 'a']]
```

Ln: 1 Col: 0

Arrays z różnymi typami

76 numpy42.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy42.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.array([1, 2, 3])
print a.dtype.name

b = np.array([1.0, 2.0, 3.0])
print b.dtype.name

c = a + b
print c
print c.dtype.name
```

końcowy obiekt jest bardziej
ogólny (upcasting).

ostrożnie z +=, *=, itd

Ln: 13 Col: 0

```
>>>
int32
float64
[ 2.  4.  6.]
float64
>>>
```

Ln: 1 Col: 0

sum(axis=...)

numpy44.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy44.py

File Edit Format Run Options Windows Help

```
import numpy as np
```

```
a = np.arange(0,12).reshape(3,4)
```

3 wiersze, 4 kolumny

```
print a, '\n'
```

```
print a.sum()
```

suma wszystkich elementów

```
print a.sum(axis=0)
```

suma z każdej kolumny


```
print a.sum(axis=1)
```

suma z każdego wiersza

Ln: 13 Col: 0

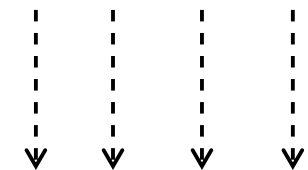
```
>>>
```

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
```



```
66
[12 15 18 21]
[ 6 22 38]
```

axis = 0



[]

Ln: 12 Col: 4

axis=0, axis=1

Python 2.7.5 Shell

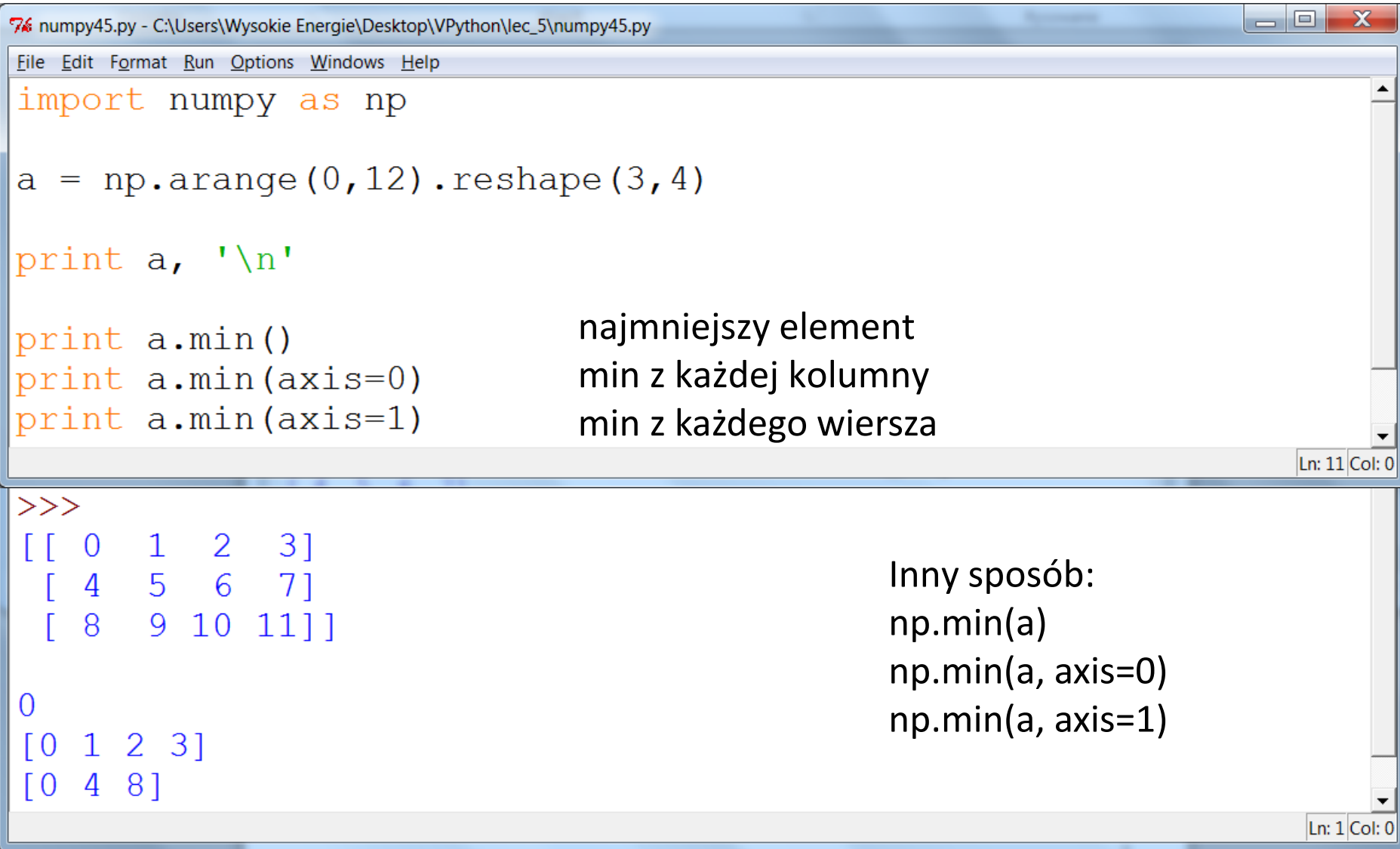
File Edit Shell Debug Options Windows Help

[[0 1 2 3] ->
[4 5 6 7] ->
[8 9 10 11] ->] axis = 1

[
axis = 0
]

Ln: 79 Col: 0

min(axis=...), max(axis=...)



The image shows a screenshot of a Python IDE window titled 'numpy45.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy45.py'. The window contains two panels. The top panel shows the source code for a script, and the bottom panel shows the output of the script.

```
File Edit Format Run Options Windows Help

import numpy as np

a = np.arange(0,12).reshape(3,4)

print a, '\n'

print a.min()
print a.min(axis=0)
print a.min(axis=1)
```

najmniejszy element
min z każdej kolumny
min z każdego wiersza

Ln: 11 Col: 0

```
>>>
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]

0
[0 1 2 3]
[0 4 8]
```

Inny sposób:
np.min(a)
np.min(a, axis=0)
np.min(a, axis=1)

Ln: 1 Col: 0

np.sort(axis=...)

numpy46_b.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy46_b.py

File Edit Format Run Options Windows Help

```
import numpy as np
```

```
a = np.array([[8,4,1], [2,1,0], [3,13,2]])
```

```
print a, '\n'
```

```
print np.sort(a, axis=0)
```

sortuje każdą kolumnę

```
print ''
```

```
print np.sort(a, axis=1)
```

sortuje każdy wiersz

Ln: 11 Col: 0

```
[[ 8  4  1]
 [ 2  1  0]
 [ 3 13  2]]
```

```
[[ 2  1  0]
 [ 3  4  1]
 [ 8 13  2]]
```

a.sort(axis=...)
zmienia sam obiekt

```
[[ 1  4  8]
 [ 0  1  2]
 [ 2  3 13]]
```

Ln: 1 Col: 0

identity()

```
numpy47.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy47.py
File Edit Format Run Options Windows Help

import numpy as np

a = np.identity(4)
print a, '\n'

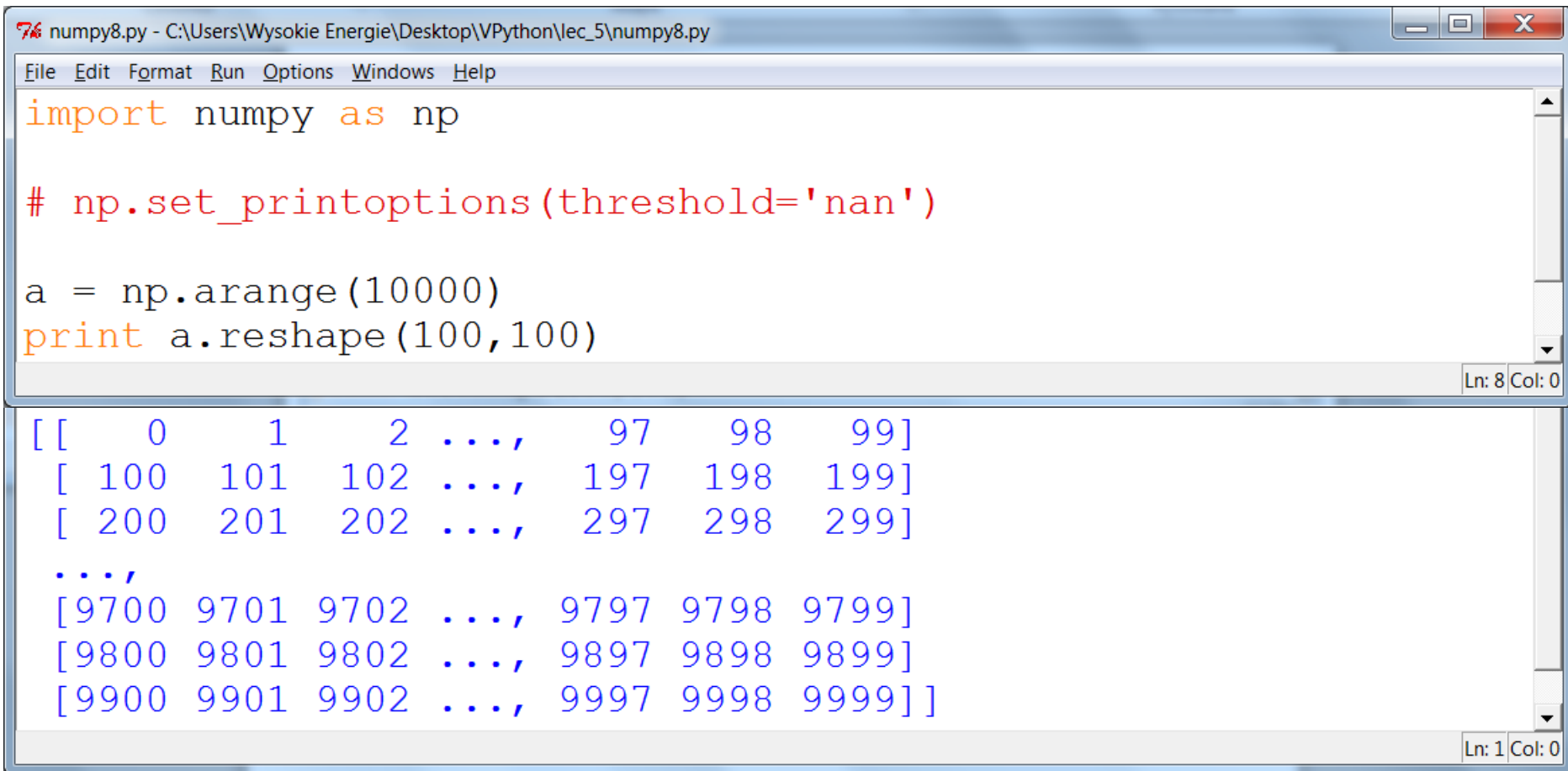
a = np.identity(7, dtype=np.int)
print a

[[ 1.  0.  0.  0.]
 [ 0.  1.  0.  0.]
 [ 0.  0.  1.  0.]
 [ 0.  0.  0.  1.]]

[[1 0 0 0 0 0 0]
 [0 1 0 0 0 0 0]
 [0 0 1 0 0 0 0]
 [0 0 0 1 0 0 0]
 [0 0 0 0 1 0 0]
 [0 0 0 0 0 1 0]
 [0 0 0 0 0 0 1]]

>>>
```

za duży array

The image shows a screenshot of a Python IDE window titled 'numpy8.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy8.py'. The menu bar includes File, Edit, Format, Run, Options, Windows, and Help. The code editor contains the following Python code:

```
import numpy as np

# np.set_printoptions(threshold='nan')

a = np.arange(10000)
print a.reshape(100,100)
```

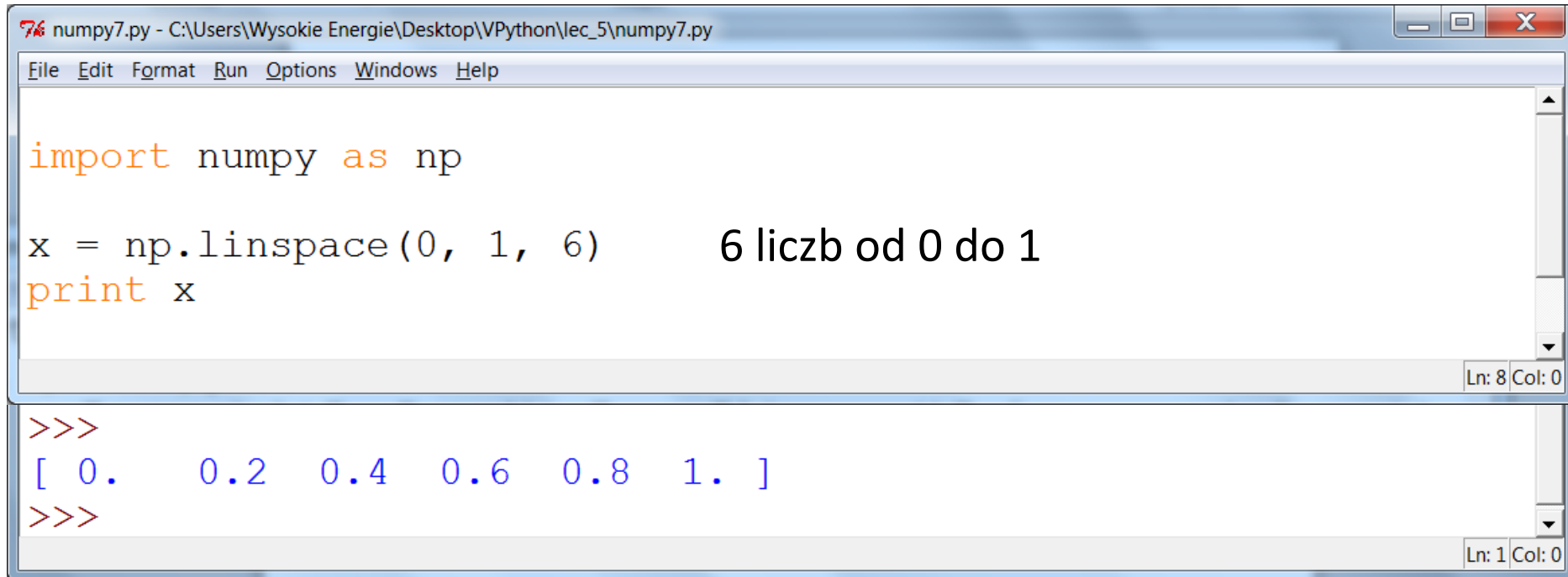
The status bar at the bottom right of the editor shows 'Ln: 8 Col: 0'. Below the code editor, the output of the program is displayed in a separate window. It shows a 100x100 array of integers, with only the first few rows visible due to the default print threshold. The output is:

```
[[ 0  1  2 ..., 97 98 99]
 [100 101 102 ..., 197 198 199]
 [200 201 202 ..., 297 298 299]
 ...,
 [9700 9701 9702 ..., 9797 9798 9799]
 [9800 9801 9802 ..., 9897 9898 9899]
 [9900 9901 9902 ..., 9997 9998 9999]]
```

The status bar at the bottom right of the output window shows 'Ln: 1 Col: 0'.

Jeśli obiekt jest za duży to NumPy pokazuje tylko *rogi*.
Można to wyłączyć poprzez `set_printoptions`.

linspace



The image shows a screenshot of a Python IDE window titled 'numpy7.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy7.py'. The window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Windows', and 'Help'. The main editor area contains the following Python code:

```
import numpy as np  
  
x = np.linspace(0, 1, 6)    6 liczb od 0 do 1  
print x
```

The status bar at the bottom right of the editor shows 'Ln: 8 Col: 0'. Below the editor is a console window showing the output of the code:

```
>>>  
[ 0.   0.2  0.4  0.6  0.8  1. ]  
>>>
```

The console window status bar shows 'Ln: 1 Col: 0'.

Iteracja

```
iterating_1.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\iterating_1.py
File Edit Format Run Options Windows Help

import numpy as np

a = np.arange(20)**2
a = a.reshape(5,4)
print a, '\n'

for i in a:
    print i
```

iteracja po każdym rzędzie

```
[[ 0  1  4  9]
 [16 25 36 49]
 [64 81 100 121]
 [144 169 196 225]
 [256 289 324 361]]

[0 1 4 9]
[16 25 36 49]
[64 81 100 121]
[144 169 196 225]
[256 289 324 361]
```

Iteracja, flat

```
iterating_2.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\iterating_2.py
File Edit Format Run Options Windows Help

import numpy as np

a = np.arange(20)**2
a = a.reshape(5,4)
print a, '\n'

for i in a.flat:      iteracja po każdym elemencie
    print i,
```

to samo z

```
for i in a:
    for el in i:
        print el,
```

```
>>>
[[ 0  1  4  9]
 [16 25 36 49]
 [64 81 100 121]
 [144 169 196 225]
 [256 289 324 361]]

0 1 4 9 16 25 36 49 64 81 100 121 144 169 196 225 256 289 324
361
```

| | | | | | | | | | | | | | | | | | | | | | | | | |
|---|----|---|----|---|----|---|----|---|----|---|----|---|---|---|---|---|---|---|---|---|---|---|---|---|
| + | - | - | - | + | - | - | - | + | - | - | - | + | - | - | - | + | - | - | - | + | - | - | - | + |
| | P | | y | | t | | h | | o | | n | | | | | | | | | | | | | |
| + | - | - | - | + | - | - | - | + | - | - | - | + | - | - | - | + | - | - | - | + | - | - | - | + |
| | 0 | | 1 | | 2 | | 3 | | 4 | | 5 | | | | | | | | | | | | | |
| | -6 | | -5 | | -4 | | -3 | | -2 | | -1 | | | | | | | | | | | | | |

`s[2:5] = 'tho'`

włączony wyłączony

Cięcia (slicing)

76 slicing_1.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\slicing_1.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.arange(5)**2
print a, '\n'

print a[1], a[-2]

print a[2:4]

print a[:3]

print a[3:]

print a[:]
```

$a[:3] = a[0:3]$

$a[3:] = a[3:\text{len}(a)]$ (do ostatniego elementu)

$a[:] = a[0:\text{len}(a)]$

Ln: 16 Col: 0

```
[ 0  1  4  9 16]
```

```
1 9
```

```
[4 9]
```

```
[0 1 4]
```

```
[ 9 16]
```

```
[ 0  1  4  9 16]
```

17

Ln: 1 Col: 0

slicing

```
slicing_2.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\slicing_2.py
File Edit Format Run Options Windows Help

import numpy as np

a = np.arange(7)**2
print a, '\n'

print a[2:6:2]          co drugi element
print a[ : :3]          a[ : :3] = a[0 : len(a) : 3]  (co trzeci element)
print a[ : :-1]         do tyłu

a[2:6] = 0
print a

[ 0  1  4  9 16 25 36]

[ 4 16]
[ 0  9 36]
[36 25 16  9  4  1  0]
[ 0  1  0  0  0  0 36]
```

slicing

```
slicing_3.py - C:\Users\Wysokie Energie\Desktop\VPytha...
File Edit Format Run Options Windows Help

import numpy as np

a = np.arange(20)**2
a = a.reshape(5,4)

print a, '\n'

2 wiersz, 3 kolumna
print a[2, 3], '\n'

wiersze, kolumny
print a[0:3, 0:2], '\n'

wiersze, wszystkie kolumny
print a[0:3, :]
```

Ln: 17 Col: 0

```
Python 2.7.5 Shell
File Edit Shell Debug Options Windows Help

>>>
[[ 0  1  4  9]
 [16 25 36 49]
 [64 81 100 121]
 [144 169 196 225]
 [256 289 324 361]]

121

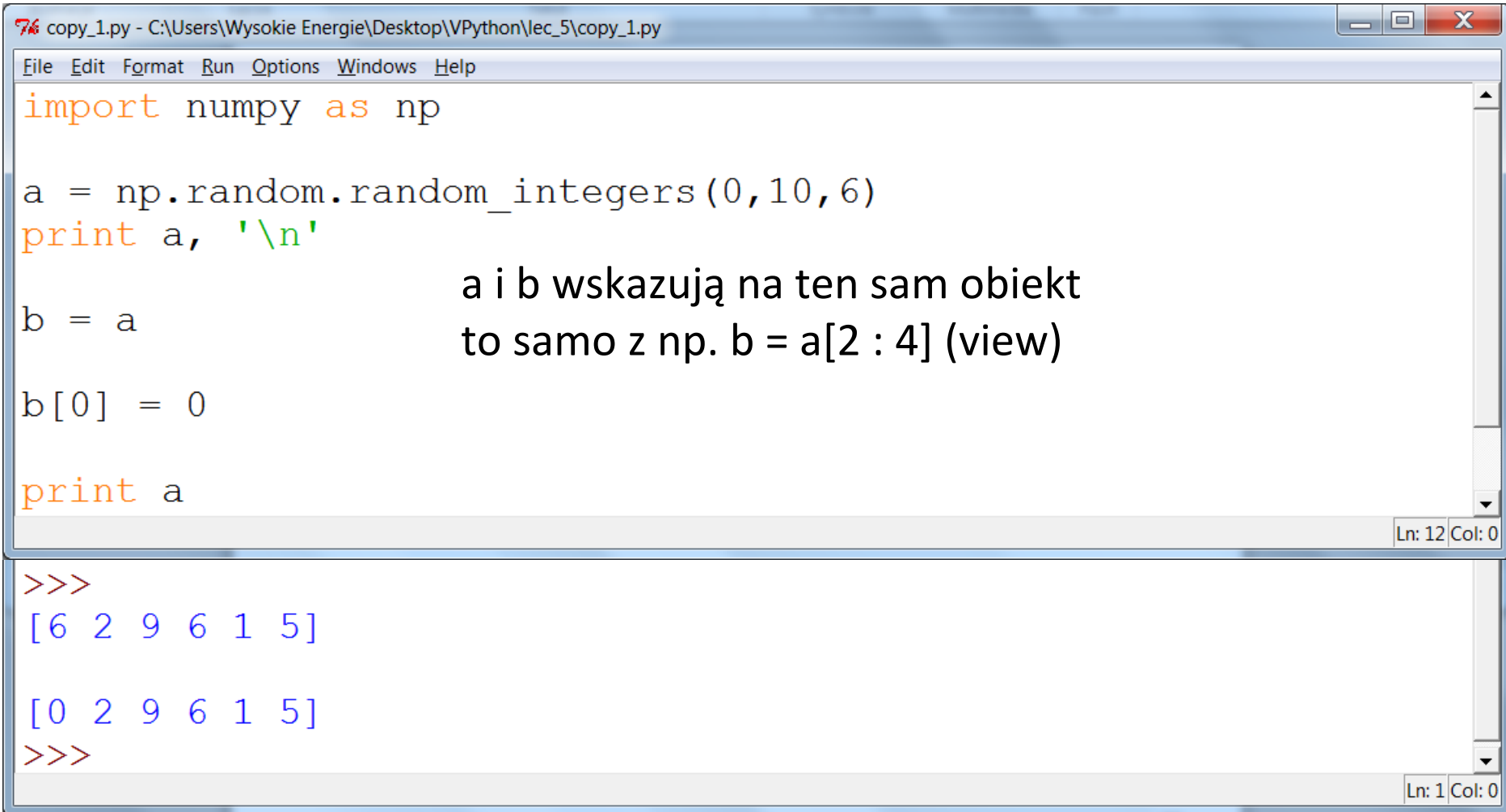
[[ 0  1]
 [16 25]
 [64 81]]

[[ 0  1  4  9]
 [16 25 36 49]
 [64 81 100 121]]

>>>
```

Ln: 20 Col: 4

copy



The image shows a screenshot of a Python IDE window titled "copy_1.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\copy_1.py". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Windows", and "Help". The main editor area contains the following Python code:

```
import numpy as np

a = np.random.random_integers(0,10,6)
print a, '\n'

b = a

b[0] = 0

print a
```

To the right of the code, there is a text annotation in black: "a i b wskazują na ten sam obiekt to samo z np. b = a[2 : 4] (view)".

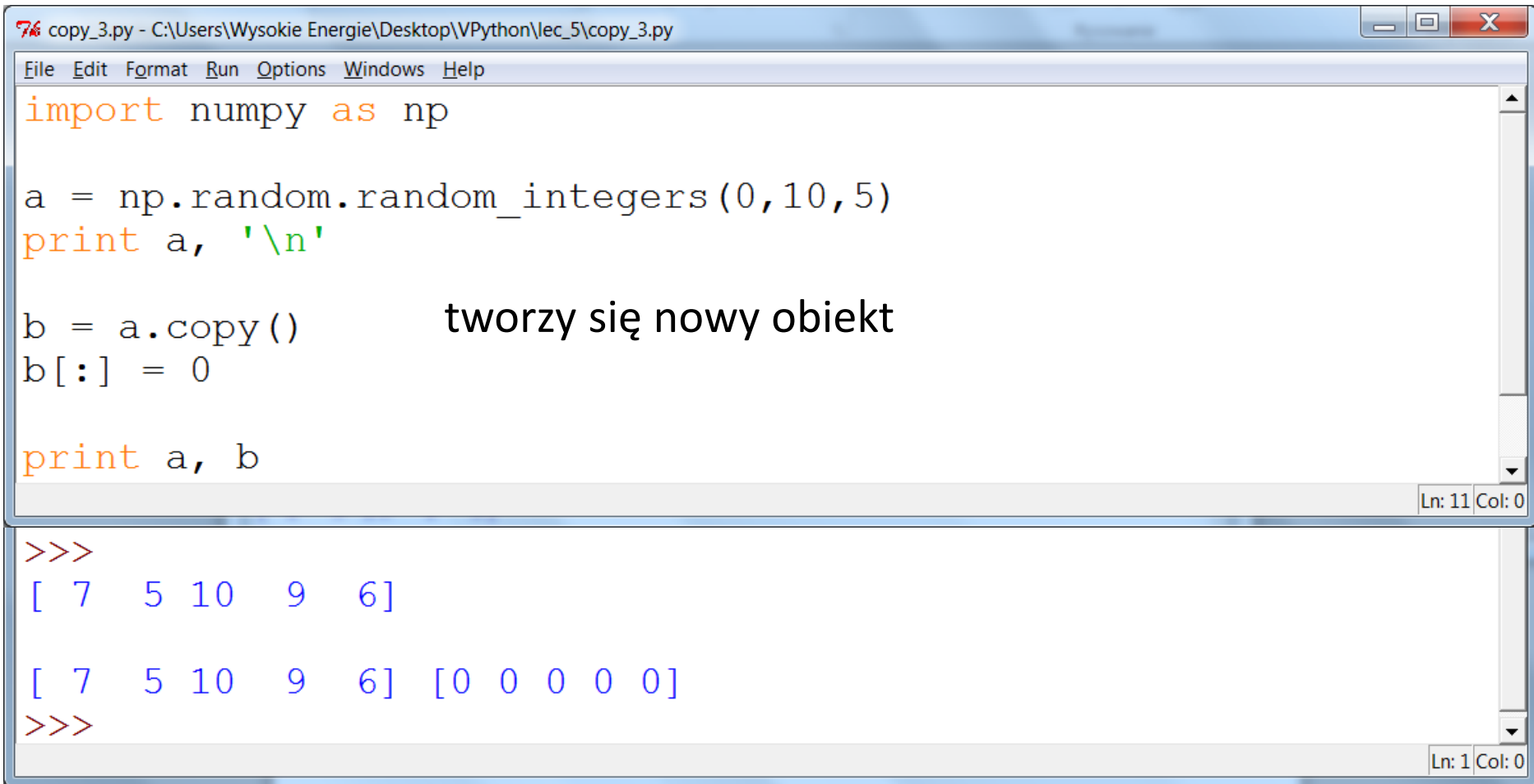
At the bottom of the window is a console area showing the execution output:

```
>>>
[6 2 9 6 1 5]

[0 2 9 6 1 5]
>>>
```

The status bar at the bottom right of the editor shows "Ln: 12 Col: 0", and the status bar at the bottom right of the console shows "Ln: 1 Col: 0".

copy



The image shows a screenshot of a Python IDE window titled 'copy_3.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\copy_3.py'. The window contains a Python script and its execution output.

The script in the editor is as follows:

```
import numpy as np

a = np.random.random_integers(0,10,5)
print a, '\n'

b = a.copy()          tworzy się nowy obiekt
b[:] = 0

print a, b
```

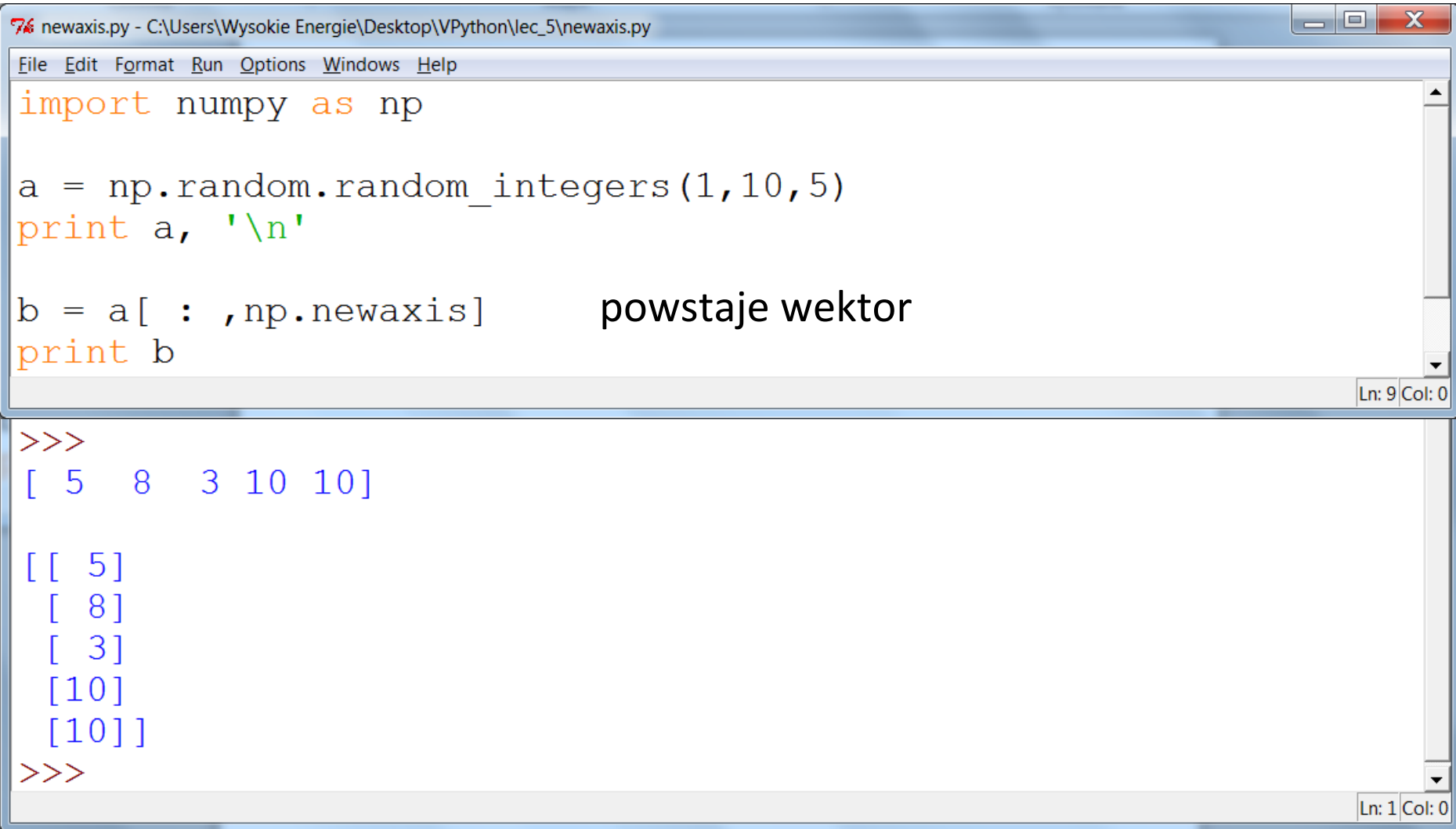
The output of the script, shown in the console, is:

```
>>>
[ 7  5 10  9  6]

[ 7  5 10  9  6] [0 0 0 0 0]
>>>
```

The IDE window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Windows', and 'Help'. The status bar at the bottom right of the editor shows 'Ln: 11 Col: 0'.

newaxis



The image shows a screenshot of a Python IDE window titled 'newaxis.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\newaxis.py'. The window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Windows', and 'Help'. The script content is as follows:

```
import numpy as np

a = np.random.random_integers(1,10,5)
print a, '\n'

b = a[ : ,np.newaxis]      powstaje wektor
print b
```

The status bar at the bottom right of the script editor shows 'Ln: 9 Col: 0'. Below the script editor is a console window showing the execution output:

```
>>>
[ 5  8  3 10 10]

[[ 5]
 [ 8]
 [ 3]
 [10]
 [10]]
>>>
```

The status bar at the bottom right of the console window shows 'Ln: 1 Col: 0'.

trik z boolean

```
trick_boolean.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\trick_boolean.py
File Edit Format Run Options Windows Help
import numpy as np

a = np.random.random_integers(0,10, (2,8))
print a, '\n'

b = (a >= 5)
print b, '\n'

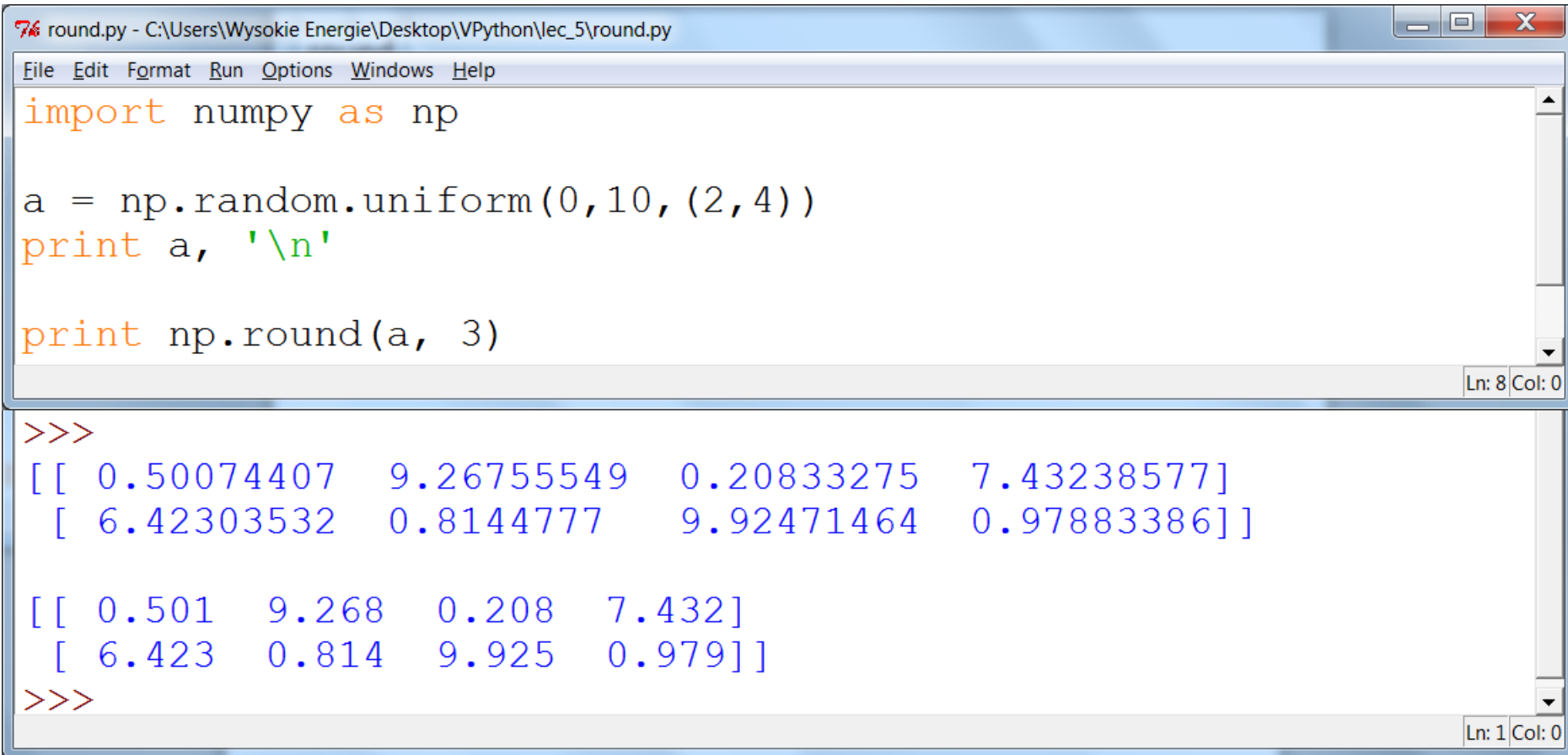
a[b] = 0
print a

>>>
[[ 3  4  5  0  7 10 10  7]
 [ 5  6  6  0  6  5  5  1]]

[[False False  True False  True  True  True  True]
 [ True  True  True False  True  True  True False]]

[[3 4 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]]
```

zaokrąglanie, round



The image shows a screenshot of a Python IDE window titled "round.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\round.py". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Windows", and "Help". The main text area contains the following Python code:

```
import numpy as np

a = np.random.uniform(0,10, (2,4))
print a, '\n'

print np.round(a, 3)
```

The status bar at the bottom right of the editor shows "Ln: 8 Col: 0". Below the editor is a console window showing the output of the script:

```
>>>
[[ 0.50074407  9.26755549  0.20833275  7.43238577]
 [ 6.42303532  0.8144777   9.92471464  0.97883386]]

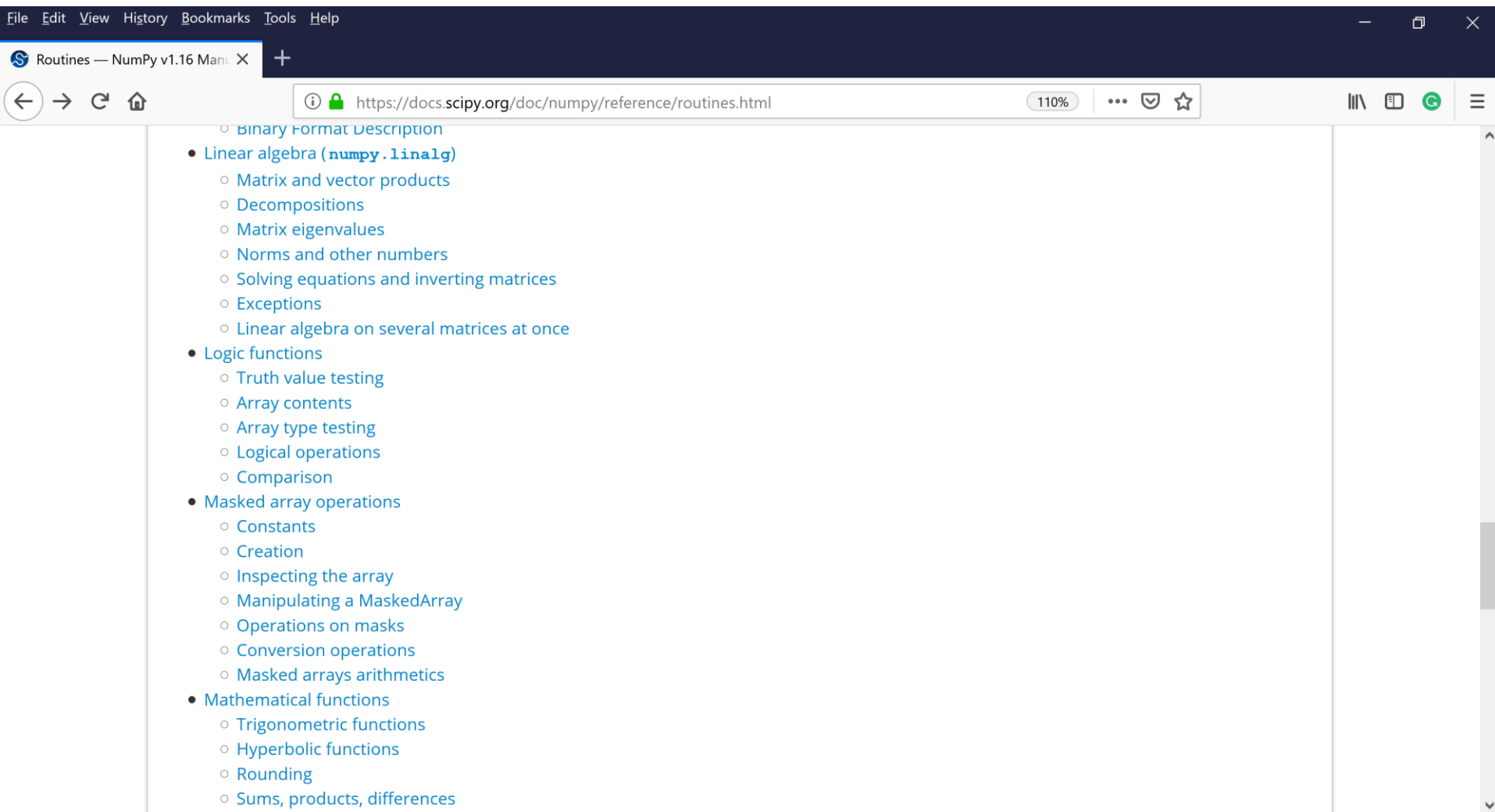
[[ 0.501  9.268  0.208  7.432]
 [ 6.423  0.814  9.925  0.979]]
>>>
```

The console window status bar at the bottom right shows "Ln: 1 Col: 0".

Proszę zobaczyć

<https://docs.scipy.org/doc/numpy/reference/routines.html>

funkcje NumPy kategoriami



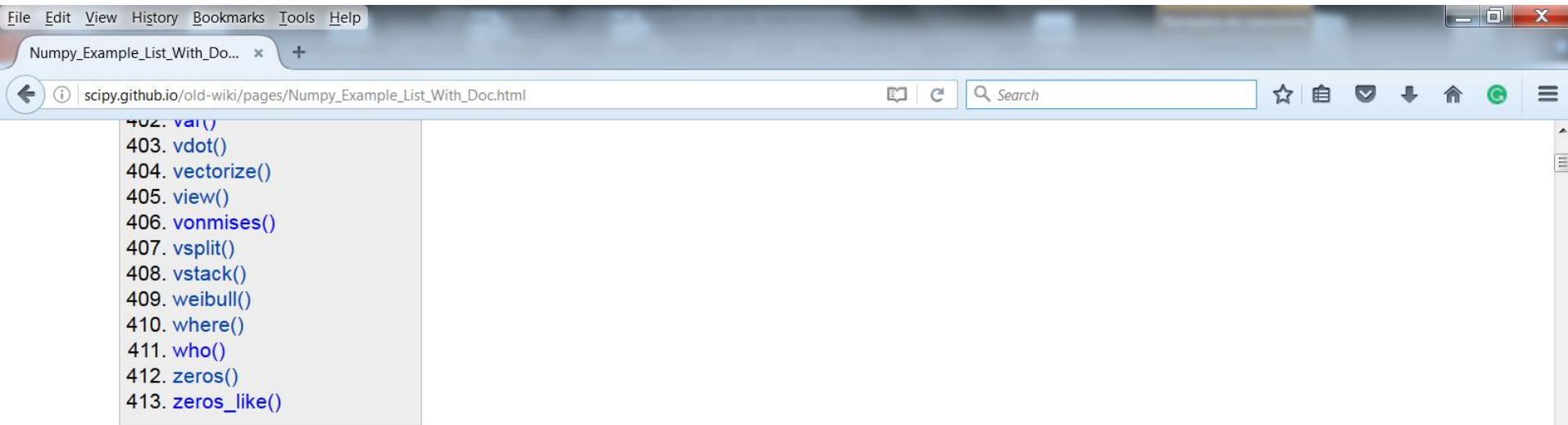
The screenshot shows a web browser window with the address bar displaying <https://docs.scipy.org/doc/numpy/reference/routines.html>. The page content is a list of NumPy routines categorized by function type. The categories are: Binary Format Description, Linear algebra (numpy.linalg), Logic functions, Masked array operations, and Mathematical functions. Each category has a list of sub-routines.

- Binary Format Description
- Linear algebra (**numpy.linalg**)
 - Matrix and vector products
 - Decompositions
 - Matrix eigenvalues
 - Norms and other numbers
 - Solving equations and inverting matrices
 - Exceptions
 - Linear algebra on several matrices at once
- Logic functions
 - Truth value testing
 - Array contents
 - Array type testing
 - Logical operations
 - Comparison
- Masked array operations
 - Constants
 - Creation
 - Inspecting the array
 - Manipulating a MaskedArray
 - Operations on masks
 - Conversion operations
 - Masked arrays arithmetics
- Mathematical functions
 - Trigonometric functions
 - Hyperbolic functions
 - Rounding
 - Sums, products, differences

Proszę zobaczyć:

https://scipy.github.io/old-wiki/pages/Numpy_Example_List_With_Doc.html

Bardzo dużo różnych funkcji



...

```
>>> from numpy import *
>>> a = arange(12)
>>> a = a.reshape(3,2,2)
>>> print a
[[[ 0  1]
 [ 2  3]]
 [[ 4  5]
 [ 6  7]]
 [[ 8  9]]]
```