1. Operacje na tablicach.

Operacje arytmetyczne wykonywane są na każdym elemencie tablicy

1.1. Operacje arytmetyczne

Przykład 1

```
import numpy as np
#inicjujemy dane
a = np.array([20, 30, 40, 50])
b = np.arange(4)
print(a)
print(b)
#wykonujemy operację i zapisujemy do nowej zmiennej
c = a - b
print(c)
#wykonujemy operację: kwadrat zawartości
print(b**2)
#możemy również zmodyfikować obecne zmienne
print(a)
a += b
print(a)
a -= b
print(a)
```

1.2. Operacje na osiach

Podczas wykonywania operacji na macierzy można podać parametr axis który pozwoli określić po której osi ma zostać wykonana operacja.

Przykład 2

```
import numpy as np
a = np.arange(12).reshape(3,4)
print(a)
#suma całej macierzy
print(a.sum())
#suma każdej z kolumn
print(a.sum(axis=0))
#minimum każdego rzędu
print(a.min(axis=1))
#skummulowana suma dla rzędu
print(a.cumsum(axis=1))
```

1.3. Mnożenie macierzy

Jeżeli chcemy wykonać operację mnożenia macierzy mamy dwie możliwości:

Przykład 3

```
import numpy as np
#inicjujemy dane
a = np.arange(3)
b = np.arange(3)
print(a)
print(b)
print(a.dot(b)) # iloczyn macierzy
print(np.dot(a,b)) #inny sposób
```

1.4. Operacje na tablicach różnej dokładności (upcasting)

Należy pamiętać, że przy operacjach na tablicach różnej dokładności wynik zawsze będzie podnoszony do wyższego.

Przykład 4

```
import numpy as np
#macierz całkowita
a = np.ones(3, dtype='int32')
print(a.dtype)
#macierz zmiennoprzecinkowa
b = np.linspace(0,np.pi,3)
print(b.dtype)
#wynikiem jest macierz zmiennoprzecinkowa
c = a+b
print(c)
print(c.dtype)
#wynikiem jest macierz liczb zespolonych
d = np.exp(c*1j)
print(d)
print(d.dtype)
```

2. Funkcje uniwersalne

Funkcje uniwersalne działają na każdym elemencie tablicy oraz tworzą nową tablicę wynikową.

Przykład 5

```
import numpy as np
b = np.arange(3)
print(b)
print(np.exp(b))
print(np.sqrt(b))
c = np.array([2., -1., 4.])
print(np.add(b,c))
```

3. Iteracja tablic

Tablice wielowymiarowe iterujemy w odniesieniu do pierwszej osi!

Przykład 6

```
import numpy as np
#generujemy macierz 3x2
a = np.arange(6).reshape(3,2)
print(a)
for b in a:
    #iterujemy wzdłuż pierwszej osi
    print(b)
    print("")
```

Możemy również iterować wszystkie elementy macierzy jakby była macierzą płaską.

Przykład 7

```
import numpy as np
#generujemy macierz 3x2
a = np.arange(6).reshape(3,2)
print(a)
for b in a.flat:
    #iterujemy jak by to była macierz płaska
    print(b)
    print("")
```

4. Przekształcenia

Kształt tablicy jest określany po ilości wymiarów na każdej z osi.

Przykład 8

```
import numpy as np
#generujemy macierz 1x6
a = np.arange(6)
print(a)
print(a.shape)
#generujemy macierz 3x3
b = np.array([np.arange(3), np.arange(3), np.arange(3)])
print(b)
print(b.shape)
```

Macierz jednego kształtu możemy zmodyfikować do zupełnie nowego na kilka różnych sposobów.

Przykład 9

```
import numpy as np
#generujemy macierz 1x6
a = np.arange(6)
print(a)
print(a.shape)
print("")
#przekształcamy ją na macierz 2x3
b = a.reshape((2,3))
print(b)
print(b.shape)
print("")
#przekształcamy ją na macierz 3x2
c = b.reshape((3,2))
print(c)
print(c.shape)
print("")
#spłaszczamy macierz zyskując pierwotny kształt ze zmiennej a
d = c.ravel()
print(d)
print(d.shape)
print("")
#transpozycja macierzy
e = b.T
print(e)
print(e.shape)
```