BIBLIOTEKA NUMPY, CZĘŚĆ 2

1. Podstawowe operatory

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(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
b = np.arange(12).reshape(3,4)
print(b)
# suma całej macierzy
print(b.sum())
```

```
# suma każdej z kolumn
print(b.sum(axis=0))
# minimum każdego rzędu
print(b.min(axis=1) )
# skumulowana suma dla rzędu
print(b.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.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=np.int32)
print(a.dtype.name)
#Macierz zmiennoprzecinkowa
b = np.linspace(0,np.pi,3)
print(b.dtype.name)
#Wynikiem jest macierz zmiennoprzecinkowa
c = a+b
print(c)
print(c.dtype.name)
#Wynikiem jest macierz liczb zespolonych
d = np.exp(c*1j)
```

```
print(d)
print(d.dtype.name)
```

Zadanie 1
Utwórz dwie macierze 1x3 różnych wartości a następnie wykonaj operację mnożenia.

Zadanie 2
Utwórz macierz 3x3 oraz 4x4 i znajdź najniższe wartości dla każdej kolumny i każdego rzędu.

Zadanie 3
Wykorzystaj macierze z zadania pierwszego i wykonaj iloczyn macierzy.

Zadanie 4
Utwórz dwie macierze 1x3 gdzie pierwsza z nich będzie zawierała liczby całkowite, a druga liczby rzeczywiste. Następnie wykonaj na nich operację mnożenia.

2. Funkcje uniwersalne

Funkcje uniwersalne działają na każdym elemencie tablicy oraz tworzą nowa 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))
```

Zadanie 5
Utwórz macierz 2x3 różnych wartości a następnie wylicz sinus dla każdej z jej wartości i zapisz do zmiennej “a”.

Zadanie 6
Utwórz nową macierz 2x3 różnych wartości a następnie wylicz cosinus dla każdej z jej wartości i zapisz do zmiennej “b”.

Zadanie 7
Wykonaj funkcję dodawania obu macierzy zapisanych wcześniej do zmiennych a i b.

3. Iteracja tablic

Tablice wielowymiarowe iterujemy w odniesieniu do pierwszej osi!

Przykład 6:

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

Możemy również iterować wszystkie elementy macierzy jakby była macierzą płaską

Przykład 7:

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

Zadanie 8

Wygeneruj macierz 3x3 i wyświetl w pętli każdy z rzędów.

Zadanie 9

Wygeneruj macierz 3x3 i wyświetl w pętli każdy element korzystając z opcji “spłaszczenia” macierzy.

4. Przekształcenia

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

Przykład 8:

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

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

Przykład 9:

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

Zadanie 10

Wygeneruj macierz 9x9 a następnie zmień jej kształt. Jakie mamy możliwości i dlaczego?

Zadanie 11

Wygeneruj macierz płaską i przekształć ja na 3x4. Wygeneruj w ten sposób jeszcze kombinacje 4x3 i 2x6. Spłaszcz każdą z nich i wyświetl wyniki. Czy są identyczne?