Numpy

Βιβλιοθήκη Python για μαθηματικούς υπολογισμούς

Η Numpy είναι ένα πακέτο (σύνολο) συναρτήσεων ειδικά σχεδιασμένες για μαθηματικούς υπολογισμούς
Κατασκευάζεται μια νέα δομή - κλάση αντικειμένων, η `array` , η οποία δίνει τη δυνατότητα να ορισθούν διανύσματα και πίνακες.
Το `array` είναι μια δομή - κλάση που μοιάζει με τη δομή `list` της Python.
Σε αντίθεση με τη `list` η διάσταση ενός `array` είναι ορισμένη από την αρχή και αποθηκεύει αντικείμενα που είναι μόνο του ίδου τύπου (δηλ. `integer` , `float` , `boolean` , `str` , ...)
Με την εντολή `array([10, 20, 30])` δημιουργούμε ένα αντικείμενο `array` που έχει διάσταση 1. Είναι το διάνυσμα `[10, 20, 30]` . Δοκιμάστε να τρέξετε το παρακάτω πρόγραμμα.

In [ ]:
```
import numpy as np
a=np.array([10, 20, 30])
print(a)
```

Υπάρχουν διάφορες εντολές οι οποίες μας δίνουν πληροφορίες για ένα αντικείμενο `array` . Όπως η `ndim` η οποία δηλώνει τη διάσταση του `array` , η `shape` για τη μορφή και η `dtype` για το είδος των στοιχείων του `array` .

In [ ]:
```
import numpy as np
a=np.array([10, 20, 30])
print(a)
# dimension of a
print(a.ndim)
# shape of a
print(a.shape)
# type of elements of a
print(a.dtype)
```

Άσκηση 1:

Τώρα δοκίμαστε να δώσετε ένα 2-διάστατο (μορφή πίνακα) `array` . Τροποποιήστε τη εντολή `array` σε `array([[1,2],[2,3]])` και τυπώστε το νεό πίνακα. Προσέξτε τη θέση των [και] για τον ορισμό των γραμμών και στηλών του πίνακα.

Επίσης υπάρχουν διάφορες χρήσιμες εντολές. Με την εντολή `arange` μπορούμε να δημιουργήσουμε ένα 1-διάστατο `array` με στοιχεία ακεραίους, όπως και με την `range` στην python. Σε αντίθεση όμως με την `range` μπορούμε να χρησιμοποιήσουμε και πραγματικούς.

In [ ]:
```
import numpy as np
# The integers from 0 to 9.
#It works like range in python
a=np.arange(10)
print(a)
# sequence with step 2
a=np.arange(3,12,2)
print(a)
# use float in arange
a=np.arange(1.2, 2.0, 0.1)
print(a)
```

Με την εντολή `linspace` μπορούμε να δημιουργήσουμε ένα 1-διάστατο `array` με ορισμένο αριθμό ισαπέχοντων πραγματικών αριθμών. Επίσης με την εντολή `zeros` δημιουργούμε ένα `array` με ορισμένο αριθμό μηδενικών και με την `ones` ένα `array` με ορισμένο αριθμό μονάδων.

In [ ]:
```
import numpy as np
# An array with 5 elements equidistant starting from 11
a=np.linspace(11., 12., 5)
print(a)
# An array with 10 zero elements
a=np.zeros(10)
print(a)
# An array with 5 one's elements
a=np.ones(5)
print(a)
```

Άσκηση 2:

Μπορούμε να δηλώσουμε και 2-διάστατα `array` με ορισμένο αριθμό μηδενικών ή μονάδων. Τροποποιήστε την παραπάνω εντολή `zeros` σε `zeros((2,3))`. Προσέξτε τη θέση των (και) για τον ορισμό των γραμμών και στηλών του πίνακα. Κάντε το ίδιο τροποποιώντας την εντολή `ones` σε `ones((2,3))`.

Επίσης μπορούμε να χρησιμοποιήσουμε τις εντολές `eye` και `diag` για να δημιουργήσουμε 2-διάστατα `array`

In [ ]:
```
import numpy as np
a=np.eye(3) # μοναδιαίος πίνακας 3x3
print(a)
v=np.array([1.,2.,3.])
print(np.diag(v)) # διαγώνιος πίνακας που σχηματίζεται από ένα διάνυσμα
```

Μπορούμε να αλλάξουμε τη μορφή ενός `array` , δηλαδή ένα 1-διάστατο `array` μπορεί να γίνει 2-διάστατο. Χρησιμοποιούμε την εντολή reshape και ο μόνος περιορισμός που έχουμε είναι ότι το νεο `array` πρεπει να έχει το ίδιο πλήθος στοιχείων με το παλιό.

In [ ]:
```
import numpy as np
# An 1-dimension array with 12 elements
a=np.arange(12)
print(a)
# Turn a to 2-dimension array with 3 lines 4 columns
a=a.reshape(3,4)
print(a)
# Turn a to 2-dimension array with 2 lines 6 columns
a=a.reshape(2,6)
print(a)
```

Μια συνήθη αλλαγή στη μορφή ενός πίνακα που μπορούμε να κάνουμε είναι να θεωρήσουμε τον ανάστροφο του. Αυτό γίνεται με το σύμβολο `T` . Μπορεί όμως να μην γίνει κάποια αλλαγή σε μονοδιάστατα `array` .

In [ ]:
```
a=np.array([[1,2,3],[5,6,7]])
print(a) # 2x3 πίνακας
print(a.T) # 3x2 πίνακας
# Προσοχή όμως σε μονοδιάστατα array
a=np.array([1, 2, 3])
print(a) # δεν υπολογίζει τον ανάστροφο σε 1x3 διανύσματα
print(a.T)
a=np.array([[1],[2],[3]])
print(a) # ισχύει ο αναστροφος σε διανύσματα 3x1
print(a.T)
```

Η πρόσβαση σε στοιχεία ενός `array` γίνεται με ανάλογο τρόπο όπως και στις λίστες.

In [ ]:
```
import numpy as np
a=np.array([1,2,3])
print(a[1])
print(a[0])
```

Άσκηση 3:

Φτιάξτε μια συνάρτηση που να δημιουργεί ένα πίνακα 3x3 με μηδενικά στοιχεία. Και στη συνέχεια να το "γεμίζει¨ με τους αριθμούς από 1 εως 9.

Επίσης το "κομμάτιασμα" γίνεται με ανάλογο τρόπο όπως και στις λίστες.

In [ ]:
```
import numpy as np
a=np.array([1,2,3,4,5])
print(a[0:4])
```

Σε ένα 2-διάστατο `array` πρέπει να ορίσουμε το κομμάτιασμα σε κάθε διάσταση

In [ ]:
```
import numpy as np
a=np.array([[1,2,5],[3,4,6],[5,6,8]])
print(a[:,1]) # δεύτερη στήλη
print(a[0:2,:]) # δυο πρώτες γραμμές
print(a[1:,1:]) # κάτω δεξια υποπίνακας
```

Άσκηση 4:

Φτιάξτε μια συνάρτηση η οποία να δέχεται ως όρισμα ένα πίνακα Α 3x3 και να επιστρέφει τα τρία διανύσματα στήλες του πίνακα