ΕΙΣΑΓΩΓΗ ΣΤΟΥΣ Η/Υ & ΕΦΑΡΜΟΓΕΣ

Σημειώσεις Εργαστηρίου: Εργαστηριακή Άσκηση 8

Βικτωρία Δασκάλου, Εμμανουήλ Τζαγκαράκης
daskalou@upatras.gr, tzagara@upatras.gr

Περιεχόμενα

Στόχος

Ο στόχος της Εργαστηριακής Άσκησης ΕΡΓΑΣΤΗΡΙΟ 8: Εισαγωγή σε python pandas είναι η εξοικείωση με χειρισμό δεδομένων δύο διαστάσεων (γραμμές και στήλες) που περιέχονται σε αρχεία και η απλή στατιστική επεξεργασία τους στη γλώσσα προγραμματισμού Python.

Θα πρέπει να υπάρχει εγκατεστημένη η γλώσσα Python, με ενσωματωμένο το περιβάλλον προγραμματισμού IDLE, καθώς και τη βιβλιοθήκη pandas. Μελετήστε την Εργαστηριακή Άσκηση 7 για την εγκατάστασή τους.

Εκφώνηση

Η εργασία ΕΡΓΑΣΤΗΡΙΟ 8: Εισαγωγή σε python pandas έχει την ακόλουθη εκφώνηση:

Έστω ότι έχει δημιουργηθεί μία δομή pandas dataframe με το πρόγραμμα `test.py` ως ακολούθως:

```
import pandas as pd

data = {'name': ['John', 'Mike', 'Sara', 'Emma'],
'age': [25, 32, 45, 18],
'salary': [50000, 60000, 75000, 30000]}

data = pd.DataFrame(data)
print(data)
```

που η παραπάνω εντολή print δίνει το ακόλουθο:

```
  name age salary
0 John 25 50000
1 Mike 32 60000
2 Sara 45 75000
3 Emma 18 30000
```

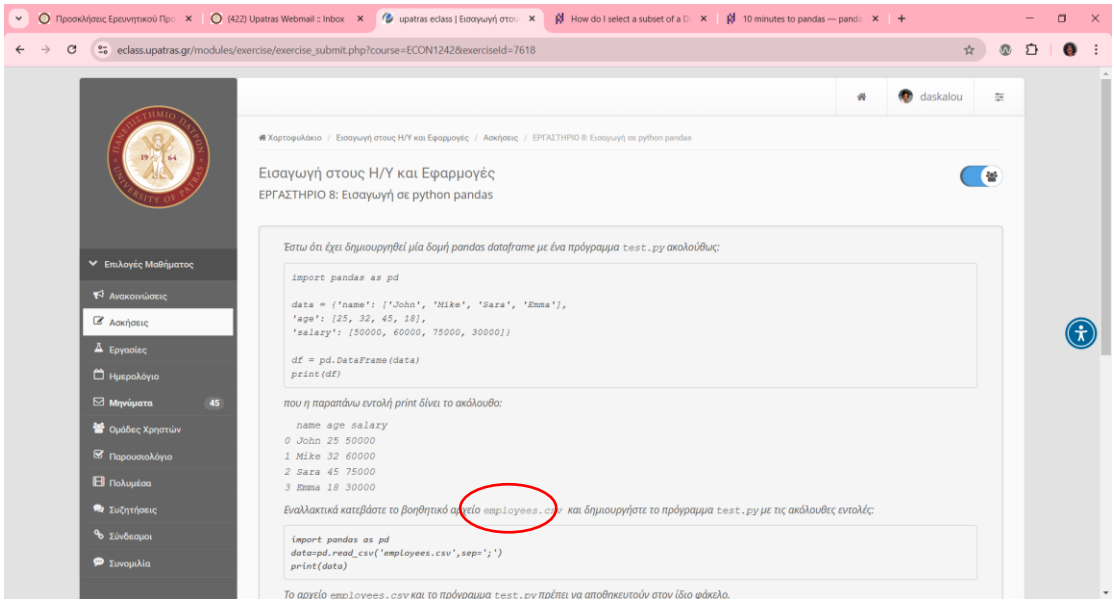
Εναλλακτικά κατεβάστε το βοηθητικό αρχείο `employees.csv` και δημιουργήστε το πρόγραμμα `test.py` με τις ακόλουθες εντολές:

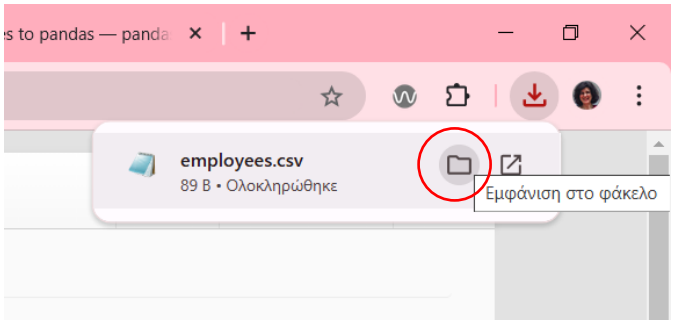
```
import pandas as pd
data=pd.read_csv('employees.csv',sep=';')
print(data)
```

Το αρχείο `employees.csv` και το πρόγραμμα `test.py` πρέπει να αποθηκευτούν στον ίδιο φάκελο.

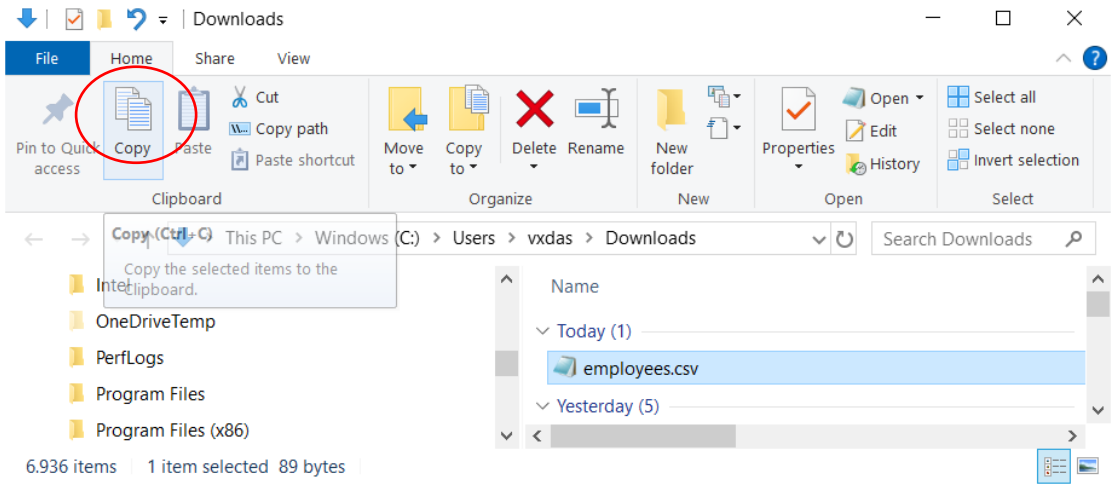
Καταβίβαση αρχείου δεδομένων

Στην εκφώνηση της εργασίας, πατώ πάνω στο link `employees.csv` της εκφώνησης και καταβιβάζεται το αρχείο στο φάκελο «Λήψεις».

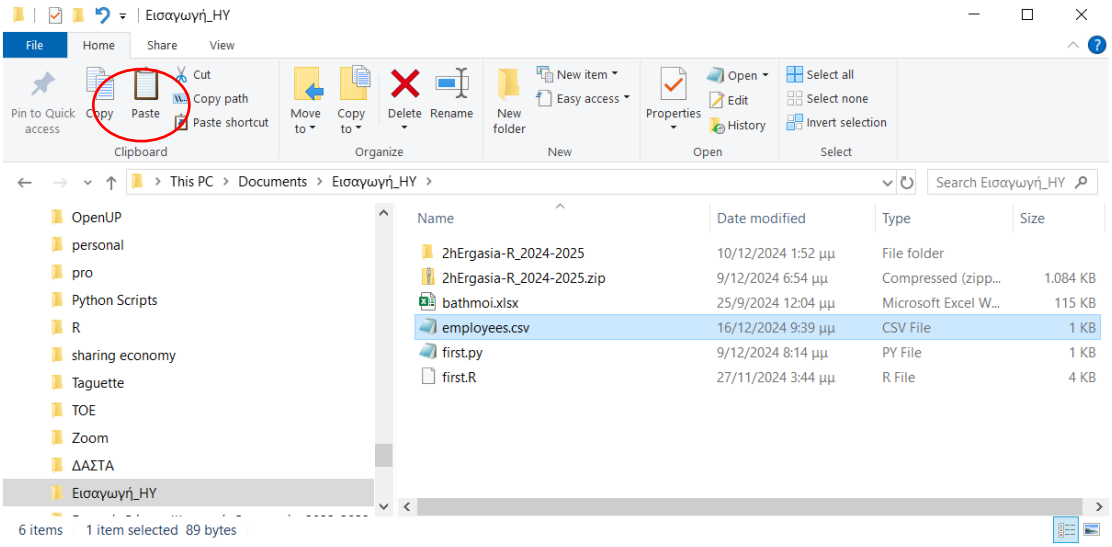




Πατώ πάνω στο όνομα του αρχείο employees.csv και επιλέγω Copy ώστε να αντιγράψω το αρχείο δεδομένων στο φάκελο που θα εργαστώ, για παράδειγμα στο φάκελο «Εισαγωγή_ΗΥ» μέσα στο φάκελο «Documents».



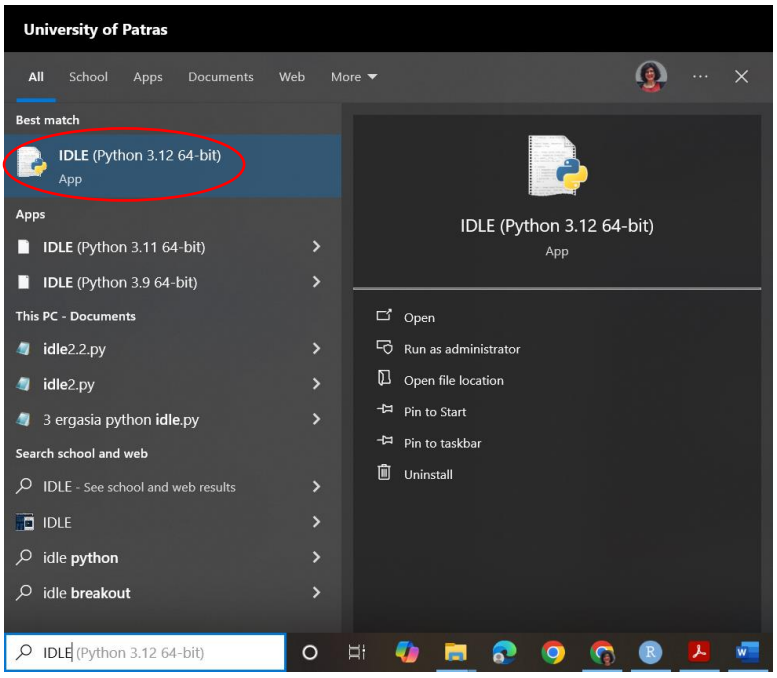
Πηγαίνω στο φάκελο αυτό και πατώ Paste ώστε να επικολλήσω το αρχείο employees.csv στο φάκελο «Εισαγωγή_ΗΥ».



Δημιουργία προγράμματος test.py και διάβασμα δεδομένων σε dataframe

Η γλώσσα Python με την εγκατάστασή της έχει ενσωματωμένο ένα περιβάλλον ανάπτυξης που ονομάζεται IDLE.

Από το Start (Έναρξη) αρχίζουμε να αναζητούμε το IDLE και το εκτελούμε.

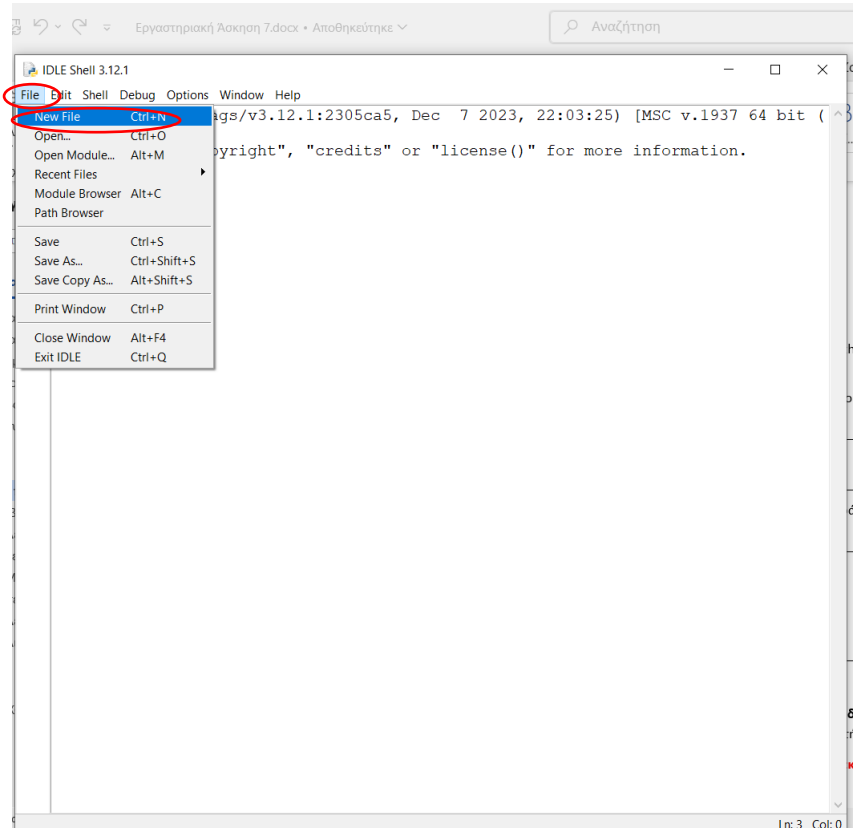


Εμφανίζεται τότε το παράθυρο του IDLE Shell όπου μπορούμε να δίνουμε απευθείας εντολές στη γλώσσα python είτε να δημιουργούμε προγράμματα.

Εναλλακτικά, μπορούμε να εγκαταστήσουμε ένα άλλο περιβάλλον ανάπτυξης προγραμμάτων python, όπως το πρόγραμμα Spyder.

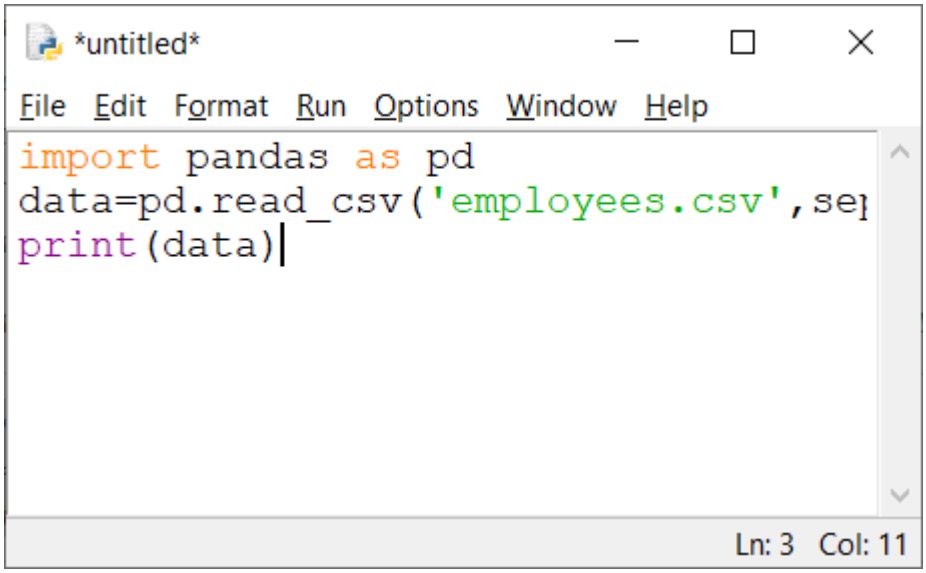
Διάβασμα αρχείου και δημιουργία πλαισίου δεδομένων (pandas DataFrame)

Για να δημιουργήσουμε νέο πρόγραμμα επιλέγουμε από το μενού *File>New File*

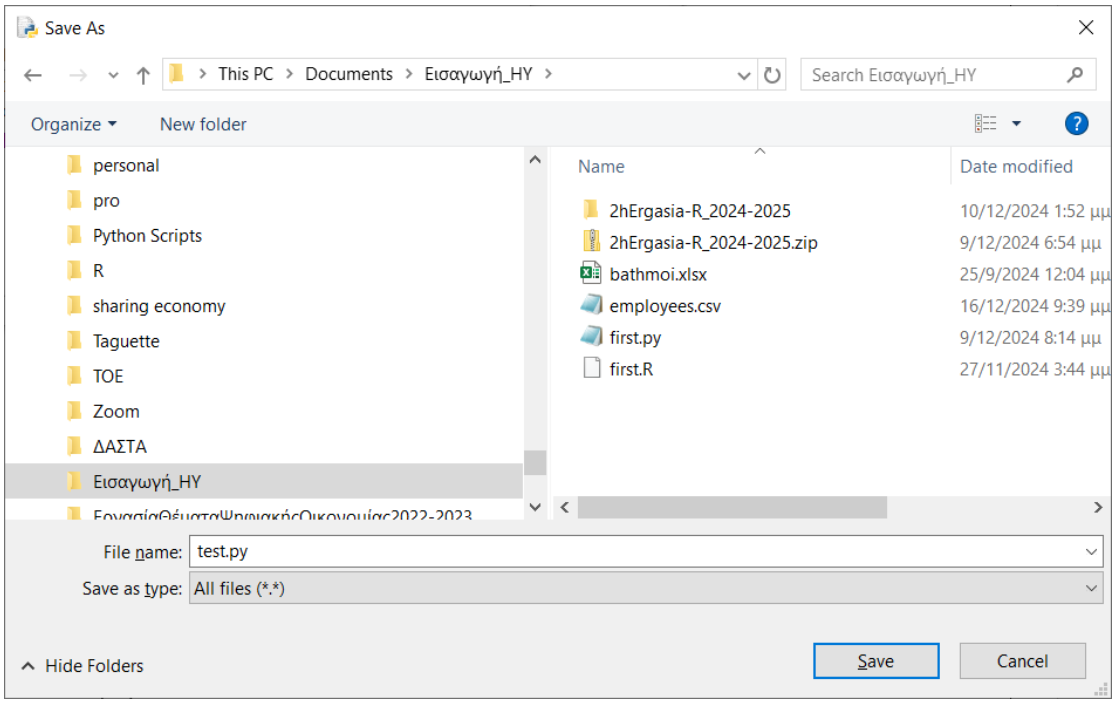


Τότε δημιουργείται ένα νέο αρχείο με όνομα *untitled*, στο οποίο θα αντιγράψουμε τις εντολές από την εκφώνηση:

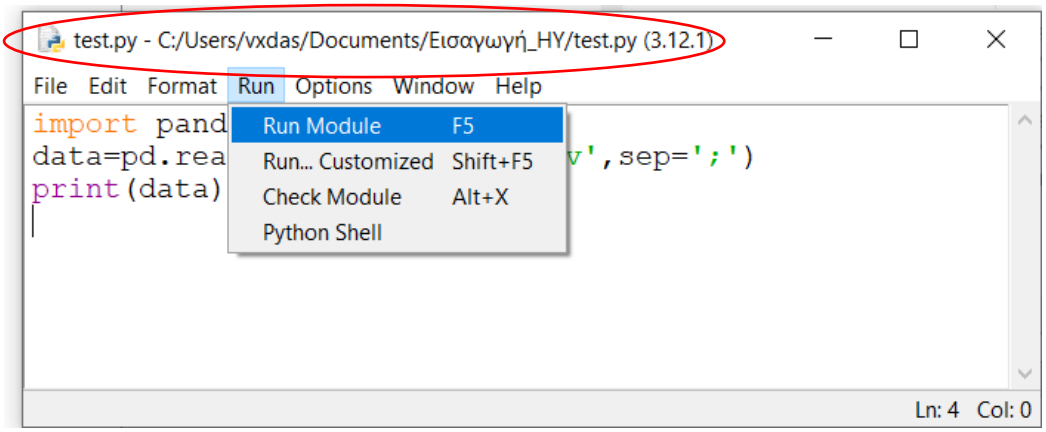
```
import pandas as pd
data=pd.read_csv('employees.csv',sep=';')
print(data)
```



Στη συνέχεια θα το αποθηκεύσουμε με Save as **στον ίδιο φάκελο που βρίσκεται το αρχείο employees.csv** και θα του δώσουμε το όνομα test.py. Προσοχή: αν το πρόγραμμα .py και το αρχείο CSV δεν βρίσκονται στον ίδιο φάκελο, η γλώσσα python δεν θα μπορέσει να το διαβάσει. Η πλήρης τοποθεσία πλέον εμφανίζεται στο πάνω μέρος του παραθύρου του αρχείου.



Στη συνέχεια θα εκτελέσουμε το πρόγραμμα test.py από την επιλογή *Run>Run Module F5*



Η εντολή

```
import pandas as pd
```

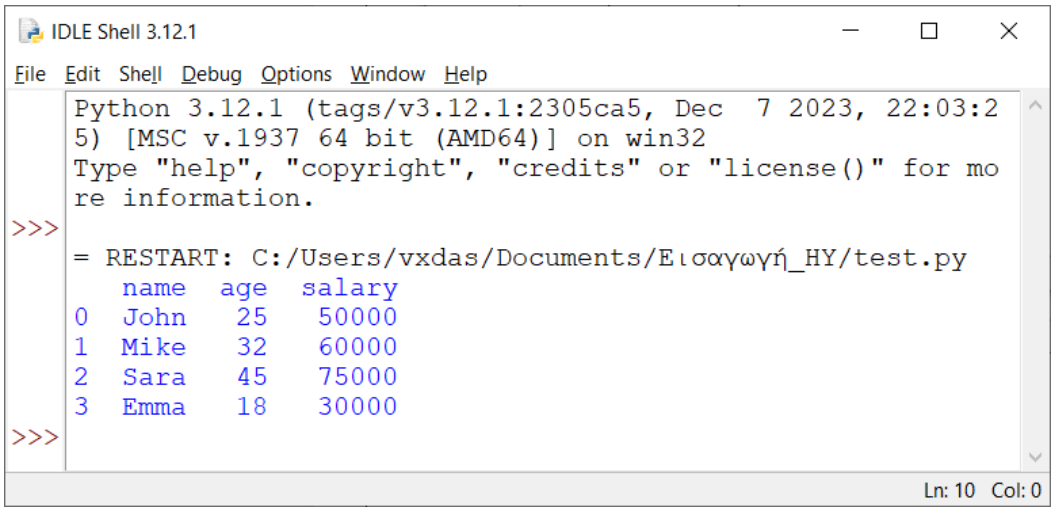
χρειάζεται για τη χρήση της βιβλιοθήκης `pandas`, η οποία θα αναφέρεται στον κώδικα ως `pd`.

Η εντολή της βιβλιοθήκης pandas με όνομα `read_csv()` ως εξής:

```
data=pd.read_csv('employees.csv',sep=';')
```

χρησιμοποιείται για να διαβάζει τα δεδομένα από το αρχείο τύπου CSV όπου έχει **χαρακτήρα διαχωρισμού στηλών** `sep=';'`. Η προκαθορισμένη τιμή είναι ότι το αρχείο **έχει επικεφαλίδα τα ονόματα των στηλών**. Αν δεν έχει επικεφαλίδα πρέπει να γράψουμε `header=None`.

Με τη χρήση της print() εκτυπώνουμε τη μεταβλητή data τύπου DataFrame (πλαίσιο δεδομένων), όπου έχουν διαβαστεί στη μνήμη τα δεδομένα του αρχείου. Αυτή έχει ως αποτέλεσμα την εκτύπωση των περιεχομένων του DataFrame στο Shell.



Η βιβλιοθήκη pandas μάς παρέχει δύο είδη τύπων μεταβλητών ή δομών δεδομένων:

Διαστάσεις	Όνομα	Περιγραφή
1	Series	1D πίνακα με ετικέτες με στοιχεία ιδίου τύπου
2	DataFrame	2D πίνακα με ετικέτες που κάθε στήλη είναι δυνητικά διαφορετικού τύπου

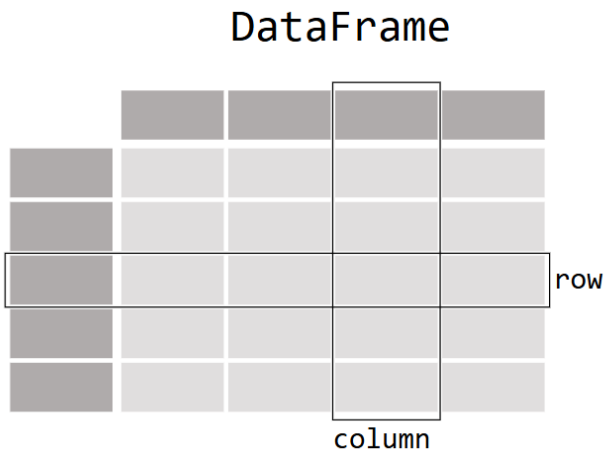
Δεικτοδότηση

Για να κατανοήσουμε καλύτερα τη **μεταβλητή data** τύπου pandas DataFrame γράφουμε τα ακόλουθα στο Shell:

```
>>> type(data)
<class 'pandas.core.frame.DataFrame'>
>>> data.index
RangeIndex(start=0, stop=4, step=1)
>>> data.columns
Index(['name', 'age', 'salary'], dtype='object')
>>> data.dtypes
name      object
age        int64
salary     int64
dtype: object
```

Ένα pandas DataFrame είναι μία δομή δύο διαστάσεων που αποτελείται από γραμμές και στήλες, όπου οι στήλες μπορεί να είναι διαφορετικού τύπου. Κάθε γραμμή και κάθε στήλη έχουν ετικέτα για το χαρακτηρισμό τους.

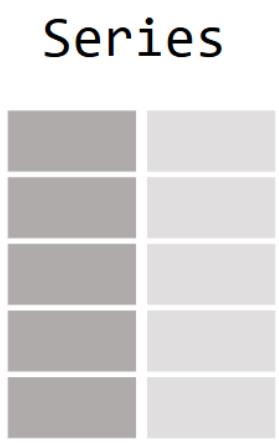
Η γενική μορφή του είναι η ακόλουθη, όπου **για κάθε γραμμή και για κάθε στήλη υπάρχει μία ετικέτα**.




```
>>> data
...
   name  age  salary
0  John   25   50000
1  Mike   32   60000
```

```
2  Sara   45   75000
```

Κάθε στήλη ενός DataFrame είναι ένα αντικείμενο τύπου **Series**, που **δεν έχει ετικέτα στήλης, αλλά μόνο ετικέτα γραμμών**.




```
>>> data['name']
...
0    John
1    Mike
2    Sara
3    Emma
Name: name, dtype: object
```

Αναλυτικότερα, η **δεικτοδότηση** ενός πλαισίου δεδομένων panda σημαίνει την επιλογή συγκεκριμένων υποσυνόλων δεδομένων (όπως γραμμές, στήλες, μεμονωμένα κελιά) από αυτό το πλαίσιο δεδομένων. Τα πλαίσια δεδομένων Pandas έχουν μια εγγενή δομή πίνακα που αντιπροσωπεύεται από σειρές και στήλες όπου κάθε γραμμή και στήλη έχει μια μοναδική **ετικέτα (label)** καθώς και έναν **αριθμό θέσης (position) (όπως μια συντεταγμένη)** μέσα στο πλαίσιο δεδομένων, και κάθε **κελί δεδομένων** χαρακτηρίζεται από τη θέση του στη διασταύρωση μίας συγκεκριμένης γραμμή και στήλης.

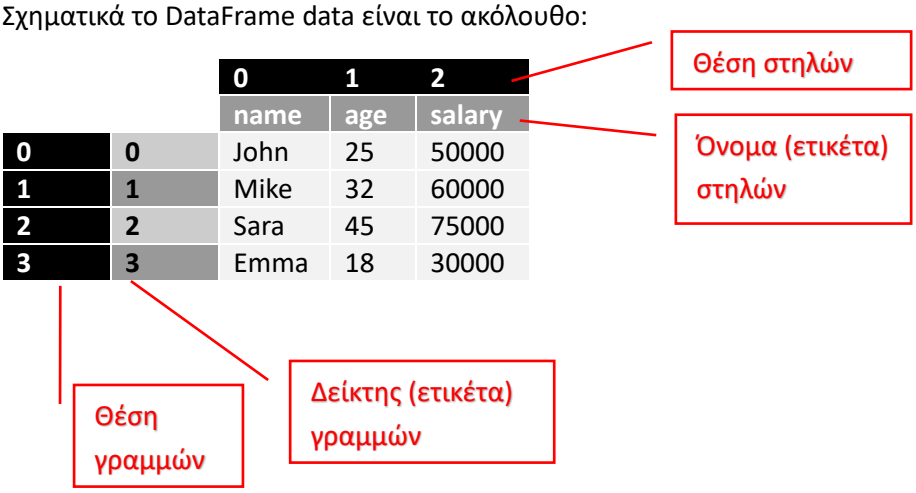
Για τον χαρακτηρισμό των **γραμμών** υπάρχουν **ετικέτες γραμμών** που ονομάζονται **δείκτης (index)**, που τον βλέπουμε με την εντολή **`data.index`**. που στη συγκεκριμένη περίπτωση είναι ακέραιος από το 0 έως το 3 ή (start=0, stop=4, step=1). Ο **δείκτης** μπορεί να είναι **ακέραιοι αριθμοί ή τιμές συμβολοσειρών.**

Για τις **στήλες** συνήθως υπάρχουν **ετικέτες στηλών** που συνήθως μοναδικά **ονόματα στηλών** που βλέπουμε με την εντολή **`data.columns`** που εδώ είναι `'name', 'age', 'salary'.`

Δεδομένου ότι τόσο ο δείκτης του πλαισίου δεδομένων όσο και τα ονόματα στηλών περιέχουν μόνο μοναδικές τιμές, μπορούμε να χρησιμοποιήσουμε αυτές τις **ετικέτες** για να αναφερθούμε σε συγκεκριμένες γραμμές, στήλες ή κελιά δεδομένων του πλαισίου δεδομένων.

Από την άλλη πλευρά, μπορούμε να περιγράψουμε κάθε γραμμή, στήλη ή σημείο δεδομένων από τη **θέση** της στη δομή του πλαισίου δεδομένων. Οι **αριθμοί θέσεων** είναι ακέραιοι που **ξεκινούν από το 0 για την πρώτη γραμμή ή στήλη και αυξάνονται κατά 1** για κάθε επόμενη σειρά/στήλη, ώστε να μπορούν επίσης να χρησιμοποιηθούν ως μοναδικές συντεταγμένες συγκεκριμένων στοιχείων πλαισίου δεδομένων (γραμμές, στήλες ή σημεία δεδομένων).

Σχηματικά το DataFrame data είναι το ακόλουθο:



Αν θέλω μπορώ να αλλάξω τον δείκτη γραμμών για να φαίνεται η διαφορά ως εξής:

```
= RESTART: C:/Users/vxdas/Documents/Εισαγωγή_HY/test.py =
   name  age  salary
0  John   25   50000
1  Mike   32   60000
2  Sara   45   75000
3  Emma   18   30000
>>> data.index = ['row_1', 'row_2', 'row_3', 'row_4']
>>> data
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
```

Τεμαχισμός (Slicing)

Για να πάρουμε ένα τεμάχιο ενός πλαισίου δεδομένων μπορούμε να χρησιμοποιήσουμε κυρίως τους ακόλουθους τρόπους. **Όπου data, εσείς θα χρησιμοποιείτε το όνομα της μεταβλητής τύπου DataFrame που έχετε ορίσει στον κώδικά σας.**

Τελεστή []	Χρήση με **ετικέτες (ονόματα) στηλών** `data['όνομα_στήλης']` Χρήση με **θέσεις γραμμών** `data[`*θέση_αρχής*`:`*θέση_τέλους*`:`*βήμα*`]` **Περιοχή**: `data[:][['όνομα_στήλης1', 'όνομα_στήλης2',..]]`
Μέθοδος loc	Χρήση με **ετικέτες γραμμών και στηλών**, δηλαδή **δείκτες και ονόματα στηλών** **Στοιχείο** `data.loc[δείκτης , όνομα_στήλης]`
Μέθοδος iloc	Χρήση με **θέσεις γραμμών και στηλών** **Στοιχείο** `data.iloc[θέση_γραμμής , θέση_στήλης]`

Στη θέση των γραμμών στον τελεστή [] και στη μέθοδο loc μπορούμε να χρησιμοποιήσουμε κριτήρια.

Τελεστής []

Στήλες

Ο τελεστής [] χρησιμοποιείται με **ετικέτες (ονόματα) στηλών** ώστε να αναφερθούμε σε **μία στήλη** στη μορφή:

`data['όνομα_στήλης']`

ή στη μορφή

`data[['όνομα_στήλης1', 'όνομα_στήλης2',..]]`

για να αναφερθούμε σε **πολλές στήλες**, όπου τα ονόματα των στηλών είναι σε μία **λίστα** μέσα σε [] και χωριζόμενα με κόμμα.

Για να αναφερθούμε στη στήλη age θα γράψουμε:

```
>>> data['age'] # μία στήλη, η age
...
row_1    25
row_2    32
row_3    45
row_4    18
Name: age, dtype: int64
```

Για να αναφερθούμε, σε πολλές στήλες, όπως η name και salary, θα γράψουμε τα ονόματά τους σε λίστα, και θα επιστραφεί ένα DataFrame:

```
>>> data[['name','salary']] #πολλές στήλες, τα ονόματα σε λίστα
...
       name  salary
row_1  John   50000
```

```
row_2  Mike   60000
row_3  Sara   75000
row_4  Emma   30000
```

Γραμμές

Ο **τελεστής []** χρησιμοποιείται και με **θέσεις γραμμών** στη γενική μορφή

data[*θέση_αρχής*:*θέση_τέλους*:*βήμα*]

όπου επιστρέφει τις γραμμές από τη ***θέση_αρχής έως και τη θέση_τέλους -1*, με *βήμα***. Υπενθυμίζεται ότι οι θέσεις των γραμμών και των στηλών στην Python ξεκινούν από μηδέν (0).

```
>>> data[0:1:1] # πρώτη γραμμή
...
       name  age  salary
row_1  John   25   50000
>>> data[:] #όλες οι γραμμές
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data[1:4:2] #από γραμμή 1 έως και γραμμή 3 με βήμα 2
...
       name  age  salary
row_2  Mike   32   60000
row_4  Emma   18   30000
```

Στοιχείο ή Περιοχή

Συνδυάζοντας τα παραπάνω, για να αναφερθούμε σε ένα στοιχείο ή μία περιοχή μπορούμε στη γενική μορφή να γράψουμε:

data[:][['όνομα_στήλης1', 'όνομα_στήλης2',..]]

Για παράδειγμα για να αναφερθούμε στις στήλες name και salary της πρώτης γραμμής θα γράψουμε:

```
>>> data[:1][['name','salary']] #πρώτη γραμμή-θέση 0, στήλες name, salary
...
       name  salary
row_1  John   50000
```

```
>>> data[:1]['name'] # όνομα της πρώτης γραμμής
...
row_1    John
Name: name, dtype: object
```

Μέθοδος loc

Η μέθοδος loc χρησιμοποιείται σε ένα DataFrame με το όνομα της μεταβλητής (εδώ data) και τις **ετικέτες γραμμών και στηλών**, δηλαδή **δείκτης και ονόματα στηλών.**

Γραμμές

Για την αναφορά σε γραμμές με τη μέθοδο loc χρησιμοποιούμε το όνομα της μεταβλητής (εδώ data) και τον δείκτη (index) των γραμμών ως εξής:

Για μία **συγκεκριμένη γραμμή** η γενική μορφή είναι: **`data.loc[δείκτης]`**

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.loc['row_3'] # γραμμή με δείκτη row_3
...
name       Sara
age          45
salary    75000
Name: row_3, dtype: object
```

Για **εύρος συνεχόμενων γραμμών** η γενική μορφή είναι η ακόλουθη. Προσοχή η γραμμή στο δείκτη τέλους συμπεριλαμβάνεται:
`data.loc[δείκτης_αρχή:δείκτης_τέλους_συμπεριλαμβανομένου:βήμα]`

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
```

```
row_4  Emma   18   30000
>>> data['row_1':'row_4':2] # από γραμμή δείκτη row_1 έως και row4, βήμα 2
...
       name  age  salary
row_1  John   25   50000
row_3  Sara   45   75000
```

Για **μεμονωμένες γραμμές**, η γενική μορφή είναι ότι βάζουμε τους **δείκτες γραμμών** που θέλουμε σε **λίστα**, δηλαδή μέσα σε [] χωριζόμενες με κόμμα:

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.loc[['row_1','row_3']] #μεμωνομένες γραμμές, λίστα δεικτών
...
       name  age  salary
row_1  John   25   50000
row_3  Sara   45   75000
```

Στήλες

Για την αναφορά σε στήλες με τη μέθοδο loc χρησιμοποιούμε το όνομα της μεταβλητής (εδώ data) και το όνομα στηλών ως εξής:

Για μία **συγκεκριμένη στήλη** η γενική μορφή είναι **`data.loc[:, όνομα_στήλης]`**

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.loc[:,'name'] # όλες γραμμές, μια στήλη name
```

```
row_1    John
row_2    Mike
row_3    Sara
row_4    Emma
Name: name, dtype: object
```

Για **εύρος συνεχόμενων στηλών** η γενική μορφή είναι η ακόλουθη:
data.loc[:, όνομα_αρχή:όνομα_τέλους]

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.loc[:,'name':'age'] #όλες γραμμές, στήλες από name έως age
       name  age
row_1  John   25
row_2  Mike   32
row_3  Sara   45
row_4  Emma   18
```

Για **μεμονωμένες στήλες**, η γενική μορφή είναι ότι βάζουμε τα ονόματα των στηλών που θέλουμε σε λίστα, δηλαδή μέσα σε [] χωριζόμενα με κόμμα:

data.loc[:, [όνομα1,όνομα2,…]]

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.loc[:,['name','salary']] # όλες οι γραμμές, λίστα στηλών
       name  salary
row_1  John   50000
```

```
row_2  Mike   60000

row_3  Sara   75000

row_4  Emma   30000
```

Κελί

Για τον καθορισμό **ενός κελιού** αναφέρουμε στη γενική μορφή:
data.loc[δείκτης,όνομα_στήλης]

```
>>> data

...

       name  age  salary

row_1  John   25   50000

row_2  Mike   32   60000

row_3  Sara   45   75000

row_4  Emma   18   30000

>>> data.loc['row_1','name']

'John'
```

Περιοχή

Για τον καθορισμό **μίας περιοχής** χρησιμοποιούμε συνδυασμό των ανωτέρω με γενική μορφή:

data.loc[δείκτης_αρχή:δείκτης_τέλους_συμπεριλαμβανομένου:βήμα , όνομα_στήλης ή εύρος_ονομάτων_στηλών ή λίστα_ονομάτων_στηλών]

Για παράδειγμα, για να πάρουμε στις *δύο πρώτες γραμμές* τις *στήλες name και salary* θα γράψουμε:

```
>>> data

...

       name  age  salary

row_1  John   25   50000

row_2  Mike   32   60000

row_3  Sara   45   75000

row_4  Emma   18   30000

>>> data.loc['row_1':'row_2',['name','salary']]

       name  salary

row_1  John   50000

row_2  Mike   60000
```

Μέθοδος iloc

Η μέθοδος iloc χρησιμοποιείται σε ένα DataFrame, με το όνομα της μεταβλητής (εδώ data) και απόλυτες **θέσεις γραμμών και στηλών, ξεκινώντας από την τιμή 0.** Η λογική είναι σχεδόν ίδια με τη μέθοδο loc.

Γραμμές

Γραμμή `data.iloc[θέση_γραμμής]`

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.iloc[2] # γραμμή 3η, θέση 2
...
name       Sara
age          45
salary    75000
Name: row_3, dtype: object
```

Γραμμές συνεχόμενες `data.iloc[`*θέση_αρχής*`:`*θέση_τέλους*`:`*βήμα*`]`
όπου η *θέση_τέλους* δεν συμπεριλαμβάνεται.

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.iloc[0:4:2] # από γραμμή θέσης 0 έως και 3, βήμα 2
...
       name  age  salary
row_1  John   25   50000
row_3  Sara   45   75000
```

Γραμμές μεμονωμένες `data.iloc[[`*θέση1,θέση2,…*`]]`

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.iloc[[0,2]] #μεμωνομένες γραμμές θέσης 0 κ 2, λίστα θέσεων
...
       name  age  salary
row_1  John   25   50000
row_3  Sara   45   75000
```

Στήλες

Συγκεκριμένη Στήλη `data.iloc[:,`θέση_στήλης`]`

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.iloc[:,0] # όλες γραμμές, στήλη θέσης 0 (name)
row_1    John
row_2    Mike
row_3    Sara
row_4    Emma
Name: name, dtype: object
```

Στήλες συνεχόμενες `data.iloc[:,`*θέση_αρχής*`:`*θέση_τέλους*`:`*βήμα*`]`

```
>>> data
...
      name  age  salary
```

```
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.iloc[:,0:2] #όλες γραμμές, στήλες από 0 (name) έως και 1 (age)
       name  age
row_1  John   25
row_2  Mike   32
row_3  Sara   45
row_4  Emma   18
```

Στήλες μεμονωμένες data.iloc[**:,***[θέση1, θέση2,…]*]

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.iloc[:,[0,2]] # όλες οι γραμμές, λίστα στηλών 0 κ 2
       name  salary
row_1  John   50000
row_2  Mike   60000
row_3  Sara   75000
row_4  Emma   30000
```

Κελί

Για τον καθορισμό **ενός κελιού** αναφέρουμε στη γενική μορφή:
data.iloc[θέση_γραμής,θέση_στήλης]

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.iloc[0,0]
```

```
'John'
```

Περιοχή

Για τον καθορισμό **μίας περιοχής** χρησιμοποιούμε συνδυασμό των ανωτέρω με γενική μορφή:

`data.iloc[θέση_αρχή:θέση_τέλους:βήμα , θέση_στήλης ή εύρος_θέσεων_στηλών ή θέσεων_στηλών]`

Για παράδειγμα για να πάρουμε τις δύο πρώτες γραμμές για τις στήλες name και salary θα γράψουμε:

```
>>> data
...
       name  age  salary
row_1  John   25   50000
row_2  Mike   32   60000
row_3  Sara   45   75000
row_4  Emma   18   30000
>>> data.iloc[0:2,[0,2]]
       name  salary
row_1  John   50000
row_2  Mike   60000
```

Δεικτοδότηση με κριτήρια

Με τον όρο κριτήρια εννοούμε λογικές συνθήκες σε σχέση με τα περιεχόμενα των στηλών. Στο παράδειγμά μας μία περίπτωση κριτηρίου θα ήταν «τους εργαζόμενους με ηλικία μεγαλύτερη των 30».

Επιλογή με κριτήρια μπορεί να πραγματοποιηθεί σε τεμαχισμούς με τον **τελεστή [] και την μέθοδο loc**, όπου τα κριτήρια μπαίνουν όπου καθορίζουμε τις γραμμές.

Για να σχηματίσουμε μία λογική συνθήκη χρησιμοποιούμε το **όνομα μίας στήλης** και έναν από τους **τελεστές σύγκρισης**. Το αποτέλεσμα είναι ένα **αντικείμενο τύπου Series με λογικές τιμές** (True, False) που προκύπτει από τη σύγκριση κάθε στοιχείου της στήλης με το κριτήριο.

```
>>> data
   name  age  salary
0  John   25   50000
1  Mike   32   60000
2  Sara   45   75000
3  Emma   18   30000
```

```
>>> data['age']>30
0    False
1     True
2     True
3    False
Name: age, dtype: bool
```

Οι τελεστές σύγκρισης είναι οι ακόλουθοι:

Τελεστής	**Ερμηνεία**
==	Ίσο
>	Μεγαλύτερο
<	Μικρότερο
>=	Μεγαλύτερο ή ίσο
<=	Μικρότερο ή ίσο
!=	Διάφορο

Οι λογικοί τελεστές είναι οι ακόλουθοι. **Προσοχή: κάθε λογική συνθήκη πρέπει να μπαίνει σε παρένθεση**.

Τελεστής R	**Ερμηνεία**	**Παράδειγμα συνθήκης & Αποτέλεσμα**

&	ΚΑΙ (True αν όλες οι συνθήκες True)	`>>> (data['age']>30)&(data['salary']>7000)`
		`0    False`
		`1     True`
		`2     True`
		`3    False`
		`dtype: bool`

|	Ή (True αν έστω μία συνθήκη True)	`(data['age']>30)|(data['salary']>7000)`
		`0    True`
		`1    True`
		`2    True`
		`3    True`
		`dtype: bool`

~	ΟΧΙ (αντίθετο)	`~(data['age']>18)`
		`0    False`
		`1    False`
		`2    False`
		`3     True`
		`Name: age, dtype: bool`

Για να ανακτήσουμε **τις γραμμές που πληρούν κάποιο κριτήριο και όλες τις στήλες**, με το **κριτήριο στη θέση των γραμμών** χρησιμοποιούμε τον **τελεστή []** ή τη **μέθοδο loc**. Άρα στη γενική μορφή θα γράψουμε:

[]	data[*κριτήρια*]	`data[(data['age']>30)&(data['salary']>7000)]`

loc	data.loc[*κριτήρια*] ή
	data.loc[*κριτήρια* ,]
		`data.loc[(data['age']>30)&(data['salary']>7000)]`
		
		`data.loc[(data['age']>30)&(data['salary']>7000),]`

Για να ανακτήσουμε τις γραμμές που πληρούν κάποιο κριτήριο και συγκεκριμένες στήλες συνδυάζουμε τις προηγούμενες μεθόδους. Για παράδειγμα, για να πάρουμε «*το όνομα και την ηλικία των υπαλλήλων με ηλικία >30 και μισθό > 7000*», θα γράψουμε:

```
data[(data['age']>30)&(data['salary']>7000)][['name','age']]
```

```
data.loc[(data['age']>30)&(data['salary']>7000),['name','age']]
```

Δημιουργία νέας στήλη

Για να δημιουργήσουμε μία νέα στήλη θέτουμε το όνομα της νέας στήλης και τα περιεχόμενά της. Για παράδειγμα μία στήλη wage για το ημερομίσθιο μπορεί να χρησιμοποιηθεί ως:

```
>>> data['wage']=data['salary']/25
>>> data
   name  age  salary    wage
0  John   25   50000  2000.0
1  Mike   32   60000  2400.0
2  Sara   45   75000  3000.0
3  Emma   18   30000  1200.0
```

Πρώτη στατιστική ανάλυση

Η βιβλιοθήκη pandas διαθέτει μεθόδους για περιγραφική στατιστική που εφαρμόζονται απευθείας σε μία μεταβλητή τύπου DataFrame ή Series. Για παράδειγμα, για να ανακτηθεί ο ελάχιστος μισθός μπορούμε να ορίσουμε:

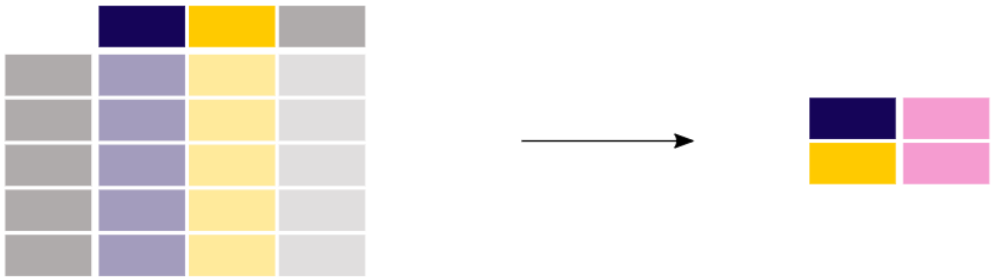
```
>>> data['salary'].min()
30000
```

Αν εφαρμόσουμε μία μέθοδο σε ένα ολόκληρο dataframe ή σε ένα τμήμα του, το αποτέλεσμα εφαρμόζεται σε κάθε στήλη και δημιουργεί μία νέα σειρά με δείκτη (index) τα ονόματα των στηλών. Για παράδειγμα:

```
>>> data[['age','salary']]
   age  salary
0   25   50000
1   32   60000
```

```
2   45   75000
3   18   30000
>>> data[['age','salary']].min()
age          18
salary    30000
dtype: int64
```

Σχηματικά η εφαρμογή της μεθόδου παρουσιάζεται στη συνέχεια:



Αν θέλουμε να εφαρμόσουμε μία μέθοδο **ανά γραμμή** και όχι ανά στήλη, μπορούμε να εφαρμόσουμε τη μέθοδο με παράμετρο **`axis=1`**. Για παράδειγμα (χωρίς να έχει κάποιο εννοιολογικό νόημα στο παράδειγμα) για να πάρουμε το άθροισμα των γραμμών στις στήλες salary και wage:

```
>>> data[['salary','wage']]
   salary    wage
0   50000  2000.0
1   60000  2400.0
2   75000  3000.0
3   30000  1200.0
>>> data[['salary','wage']].sum()
salary    215000.0
wage        8600.0
dtype: float64
>>> data[['salary','wage']].sum(axis=1)
0    52000.0
1    62400.0
2    78000.0
3    31200.0
```

```
dtype: float64
```

Η μέθοδος **`describe()`** παρέχει όλα τα βασικά στατιστικά μεγέθη στις αριθμητικές στήλες ενός dataframe:

```
>>> data.describe()
             age        salary         wage
count   4.000000      4.000000     4.000000
mean   30.000000  53750.000000  2150.000000
std    11.518102  18874.586088   754.983444
min    18.000000  30000.000000  1200.000000
25%    23.250000  45000.000000  1800.000000
50%    28.500000  55000.000000  2200.000000
75%    35.250000  63750.000000  2550.000000
max    45.000000  75000.000000  3000.000000
```

Οι σημαντικότερες στατιστικές μέθοδοι σε pandas είναι:

Στατιστικό μέγεθος	**Μέθοδος με προκαθορισμένες τιμές**
	axis: {στον άξονα του index-κάθετα (0), στον άξονα των στηλών-οριζόντια (1)} **skipname=True**: αγνοεί τα κενά κελιά (NA) **numeric_only=True**: λαμβάνει υπόψην μόνο κελιά τύπου float, int, bool
Μέγιστο	**DataFrame.max(*axis*=*0*, *skipna*=*True*, *numeric_only*=*False*)**
Ελάχιστο	**DataFrame.min(*axis*=*0*, *skipna*=*True*, *numeric_only*=*False*)**
Αριθμητικός μέσος	**DataFrame.mean(*axis*=*0*, *skipna*=*True*, *numeric_only*=*False*)**
Διάμεσος	**DataFrame.median(*axis*=*0*, *skipna*=*True*, *numeric_only*=*False*)**
Διακύμανση δείγματος, πληθυσμού	**DataFrame.var(*axis*=*0*, *skipna*=*True*, *ddof*=*1*, *numeric_only*=*False*)** ddof {δείγματος *(1), πληθυσμού (0)}*
Τυπική απόκλιση δείγματος, πληθυσμού	**DataFrame.std(*axis*=*0*, *skipna*=*True*, *ddof*=*1*, *numeric_only*=*False*)** ddof {δείγματος *(1), πληθυσμού (0)}*
Πλήθος (χωρίς κενά)	**DataFrame.count(*axis*=*0*, *numeric_only*=*False*)**
Άθροισμα	**DataFrame.sum(*axis*=*0*, *skipna*=*True*, *numeric_only*=*False*)**

Στατιστική ανάλυση σε τεμάχια με κριτήρια

Εφαρμόζοντας τις μεθόδους τεμαχισμού και τις συναρτήσεις μπορούμε να βρούμε το στατιστικό μέγεθος οποιουδήποτε τεμαχίου. Για παράδειγμα για να βρούμε «*το μέσο μισθό των υπαλλήλων με ηλικία μικρότερη του 30 και μισθό άνω των 5000*» πρέπει πρώτα να βρούμε το σωστό κριτήριο και τη σωστή στήλη και μετά να εφαρμόσουμε τη μέθοδο.

Το **κριτήριο** είναι:

```
>>> (data['age'] < 30)&(data['salary' ]>5000)
...
0     True
1    False
2    False
3     True
dtype: bool
```

Η **στήλη** μπορεί να ανακτηθεί με δύο τρόπους: με τον τελεστή [] ή με τη μέθοδος loc:

```
>>> data[:]['salary']
0    50000
1    60000
2    75000
3    30000
Name: salary, dtype: int64
>>> data.loc[:,'salary']
0    50000
1    60000
2    75000
3    30000
Name: salary, dtype: int64
```

Το **κριτήριο στη θέση των γραμμών** θα μου δώσει τη στήλη του μισθού για όσους πληρούν το κριτήριο:

```
>>> data[( data['age'] < 30) &(data['salary' ]>5000)]['salary']
0    50000
3    30000
Name: salary, dtype: int64
>>> data.loc[( data['age'] < 30) &(data['salary' ]>5000),'salary']
0    50000
3    30000
Name: salary, dtype: int64
```

Τώρα εφαρμόζω τη **μέθοδο του μέσου όρου (mean)**:

```
>>> data[( data['age'] < 30) &(data['salary' ]>5000)]['salary'].mean()
```

```
40000.0
>>> data.loc[( data['age'] < 30) &(data['salary' ]>5000),'salary'].mean()
...
40000.0
```

Τώρα είστε έτοιμοι να λύσετε την Εργαστηριακή Άσκηση!

Δοκιμάστε τις!