ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

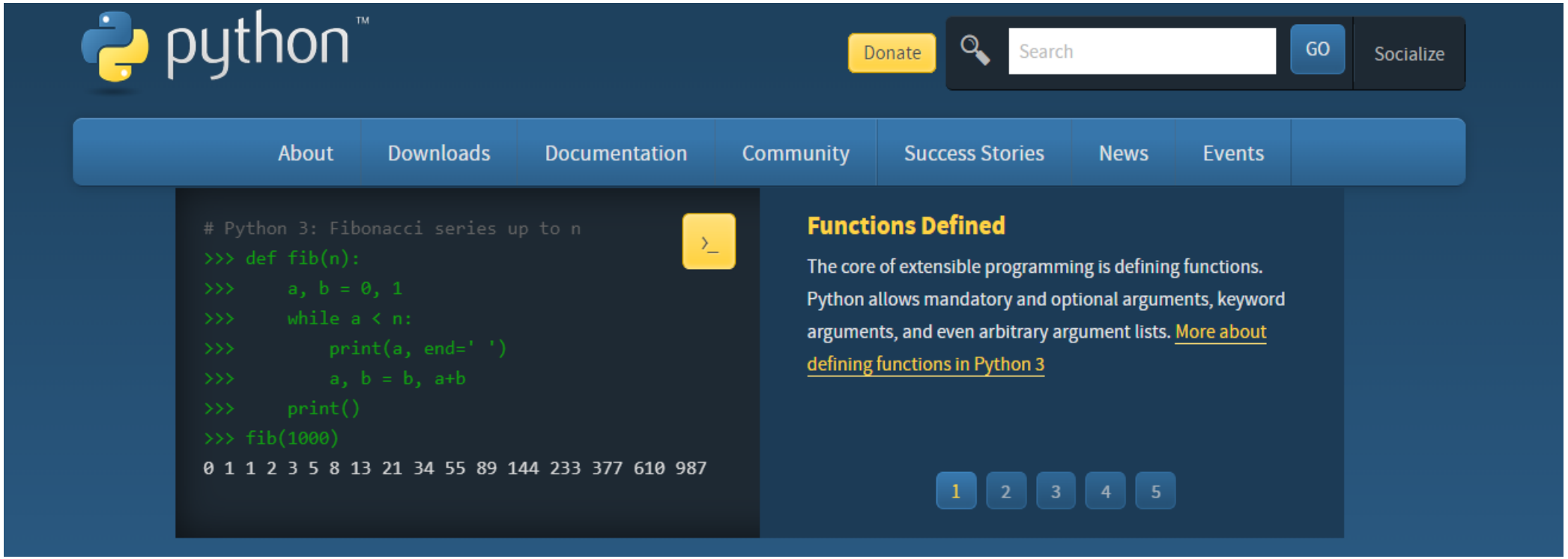
Εισαγωγικές έννοιες στη Python

Αθανάσιος Σάκκας, ΟΠΑ

I. Εγκατάσταση της Python σε περιβάλλον Windows

1. Επισκεφτείτε την επίσημη ιστοσελίδα της Python στη διεύθυνση https://www.python.org/

2. Κατεβάστε την Python που αντιστοιχεί στο λειτουργικό σας σύστημα (πατήστε το κουμπί Downloads και θα σας εμφανίσει αυτόματα την Python που προτείνεται προς εγκατάσταση στον υπολογιστήσας).

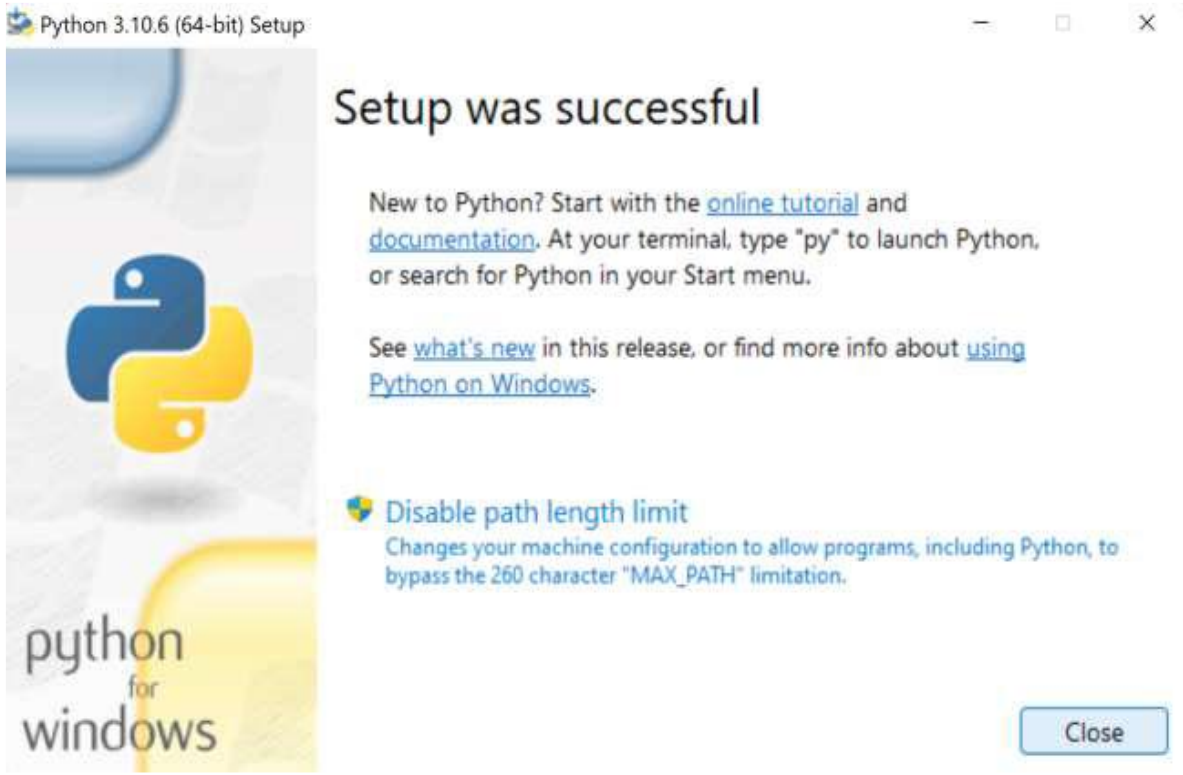


3. Μετά την επιλογή, ένα εκτελέσιμο αρχείο με προέκταση.exe μεταφορτώνεται στον υπολογιστή σας και τοποθετείται στον προκαθορισμένο φάκελο μεταφόρτωσης (συνήθως Downloads).

4. Ανοίξτε (Open) το εκτελέσιμο αρχείο που μεταφορτώθηκε στο προηγούμενο βήμα,είτε κάνοντας διπλό κλικ στο αρχείο ή επιλέγοντας το Open στο διπλανό μενού.

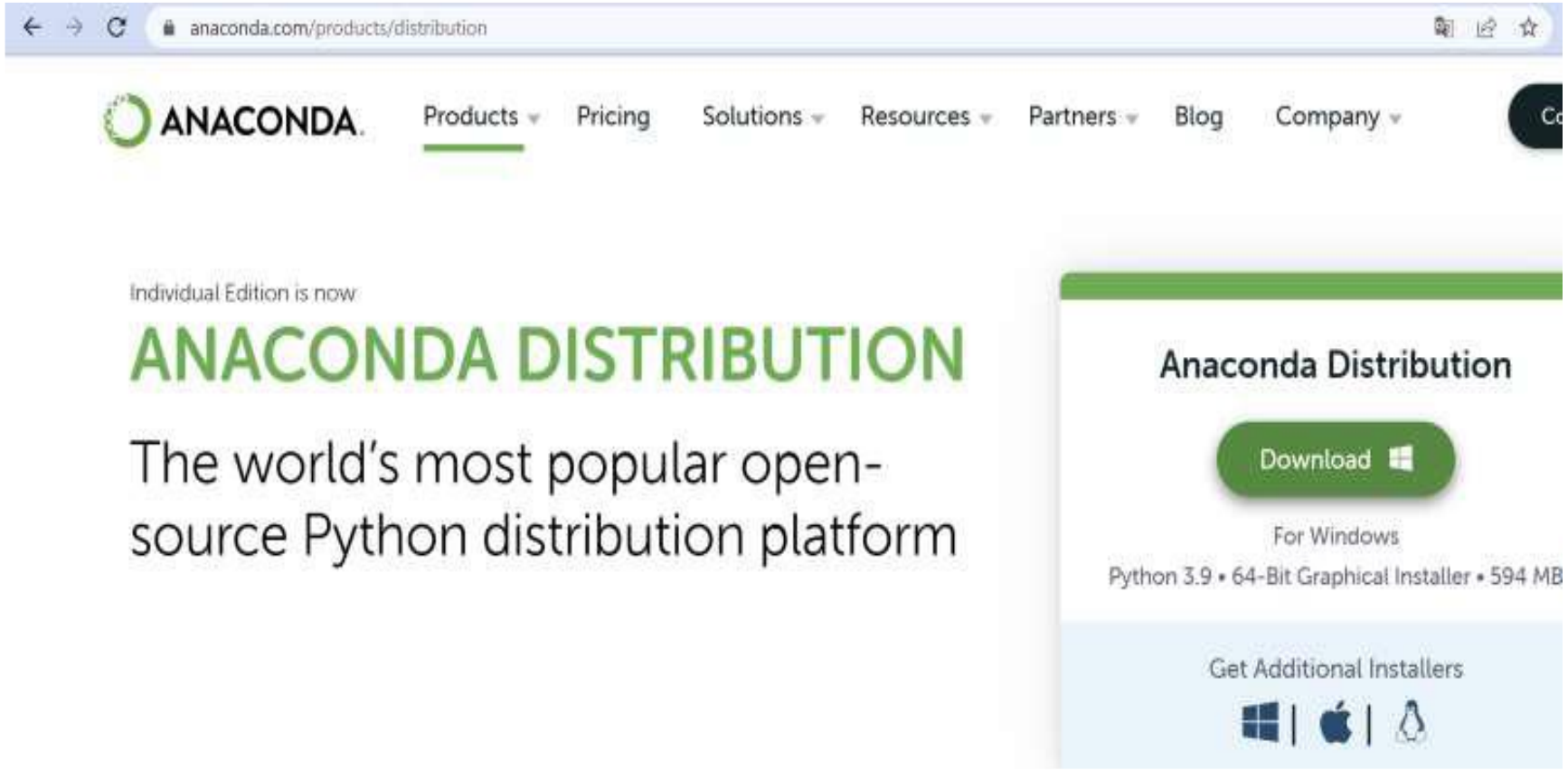
5. Στο παράθυρο εγκατάστασης που εμφανίζεται επιλέξτε Add Python…to Path και στη συνέχεια Install now.

6. Στο τέλος θα πρέπει να εμφανιστεί μήνυμα που σας πληροφορεί για την επιτυχή ολοκλήρωση της εγκατάστασης, όπως στην εικόνα:



7. Επιλέξτε “Close” για να κλείσει το παράθυρο. Η Python εγκαταστάθηκε επιτυχώς στον υπολογιστή σας!

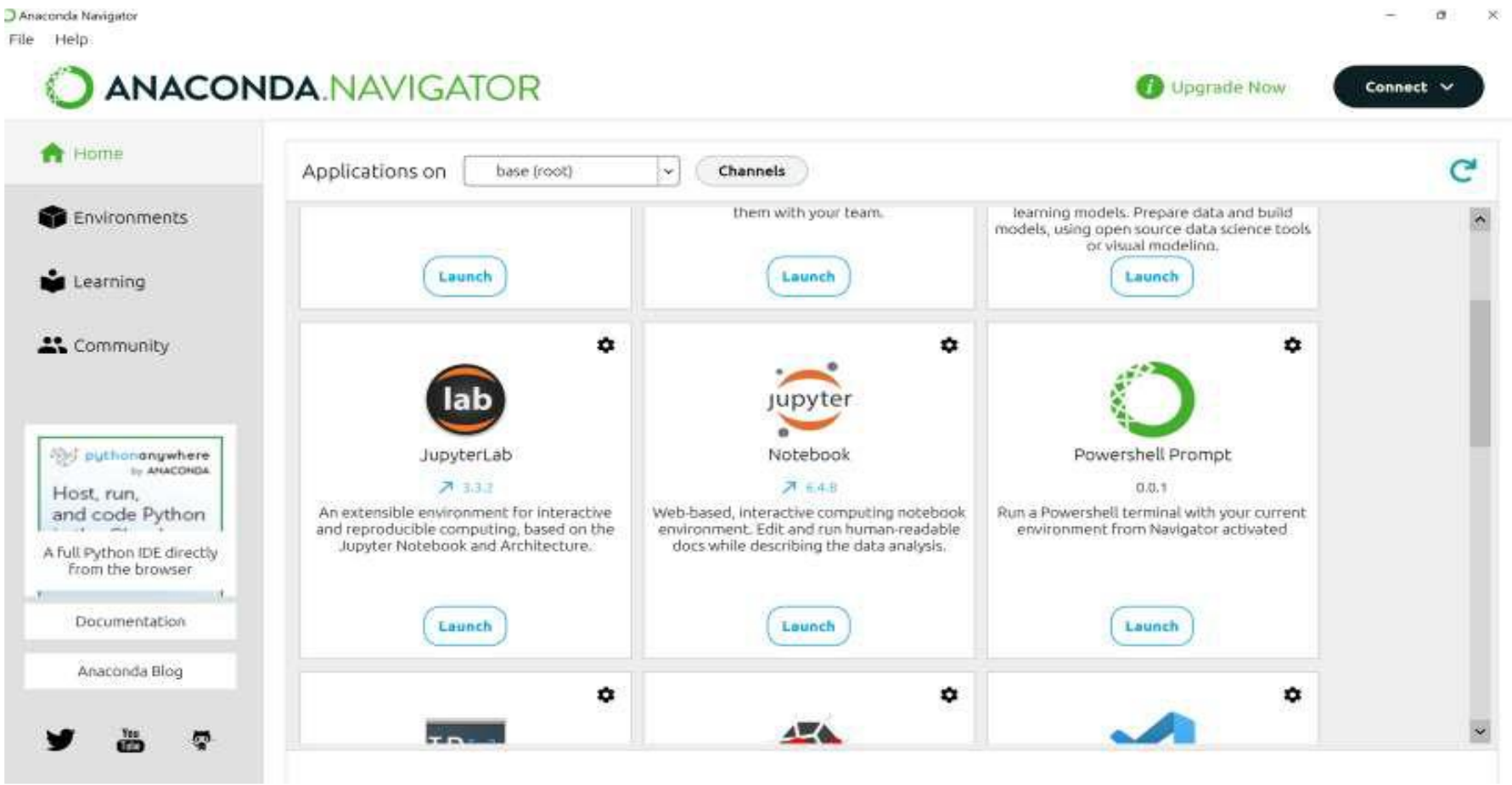
8 .Κατεβάστε την έκδοση Anaconda που αντιστοιχεί στο λειτουργικό σας σύστημα https://www.anaconda.com/



9. Προχωρήστε την εγκατάσταση με βάση τις προτεινόμενες (default) επιλογές.

10. Ανοίξτε το Anaconda Navigator.

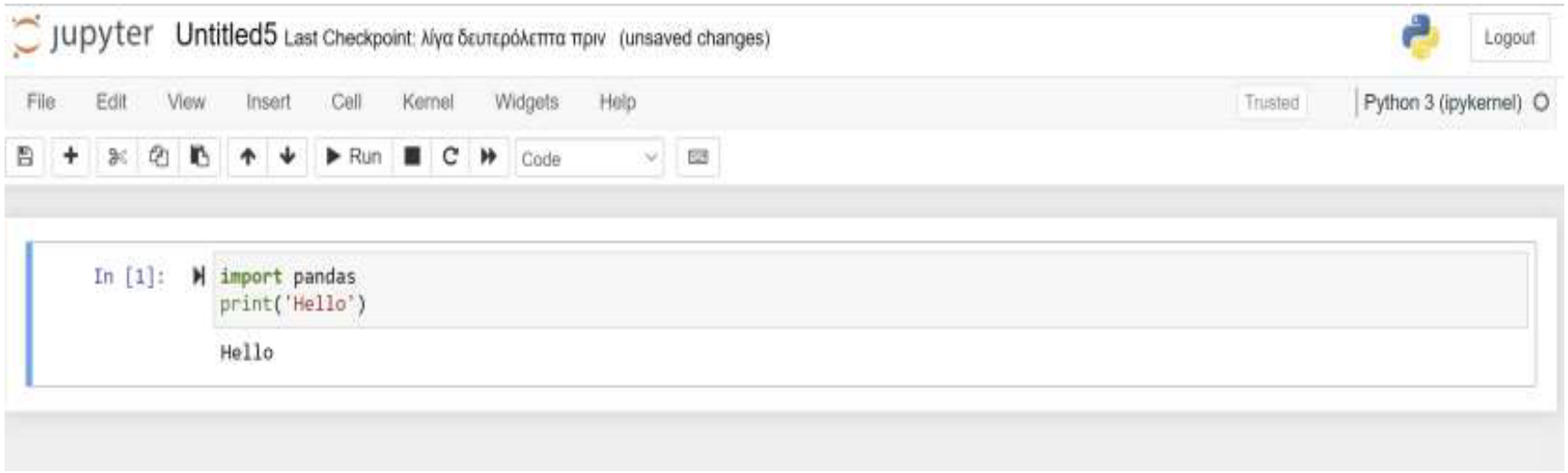
11. Από το Tab “Home” κάνετε launch το Jupyter Notebook.



12. Πατήστε “New” (πάνω δεξιά) και μετά Python3.

13. Σε ένα κελί του Jupyter Notebook κάντε copy-paste τον παρακάτω κώδικα και πατήστε Ctrl+Enter:

```
import pandas
print("Hello")
```



14. Αν από κάτω εμφανιστεί η λέξη Hello τότε όλα πήγαν καλά!

II. Εισαγωγή στη Python

- Η Python είναι μια ερμηνευμένη, υψηλού επιπέδου, γενικής χρήσης γλώσσα προγραμματισμού. Δημιουργήθηκε από τον Guido van Rossum και κυκλοφόρησε για πρώτη φορά το 1991.
- Η Python έχει γίνει μια κοινή γλώσσα για την έρευνα μηχανικής μάθησης.
- Η Python 3.0, που κυκλοφόρησε το 2008, ήταν μια σημαντική αναθεώρηση της γλώσσας. Αυτό το μάθημα χρησιμοποιεί την Python 3.
- Θα ξεκινήσουμε με την εκτύπωση του Hello World

```
In [1]: print("Hello World")
        Hello World
```

- Ο παραπάνω κώδικας μεταβιβάζει μια σταθερή συμβολοσειρά (string), που περιέχει το κείμενο "hello world" σε μια συνάρτηση που ονομάζεται print.
- Μπορείτε επίσης να αφήσετε σχόλια στον κώδικά σας για να εξηγήσετε τι κάνετε. Τα σχόλια μπορούν να ξεκινήσουν οπουδήποτε στη γραμμή με το σύμβολο #.
- Η σταθερή συμβολοσειρά, που περικλείεται σε εισαγωγικά, ορίζει τις κυριολεκτικές (literal) τιμές της συμβολοσειράς μέσα στο πρόγραμμά σας.

- Το triple quote επιτρέπει πολλές γραμμές κειμένου.

```
In [2]: # Το τριπλό απόσπασμα (triple quote) επιτρέπει πολλαπλές γραμμές κειμένου.
        print("""Print
        Multiple
        Lines
        """)

        Print
        Multiple
        Lines
```

- Η Python επιτρέπει τους αριθμούς ως κυριολεκτικές σταθερές (literal constants) σε προγράμματα. Η Python περιλαμβάνει υποστήριξη για αριθμούς κινητής υποδιαστολής (floating-point), ακέραιους (integer), μιγαδικούς (complex) και άλλους τύπους. Αυτό το μάθημα δεν θα κάνει χρήση μιγαδικών αριθμών. Σε αντίθεση με τις συμβολοσειρές, τα εισαγωγικά δεν περικλείουν αριθμούς.
- Η παρουσία υποδιαστολής διαφοροποιεί τους αριθμούς κινητής υποδιαστολής και ακέραιους αριθμούς. Για παράδειγμα, η τιμή 10 είναι ένας ακέραιος αριθμός, ενώ το 10.3 είναι ένας αριθμός κινητής υποδιαστολής. Για παράδειγμα, ο παρακάτω κώδικας εκτυπώνει δύο αριθμούς.

```
In [3]: print(10)
        print(10.3)

        10
        10.3
```

- Μέχρι στιγμής, έχουμε δει μόνο πώς να ορίσουμε κυριολεκτικές τιμές (literal values) αριθμών και συμβολοσειρών. Αυτές οι κυριολεκτικές τιμές είναι σταθερές και δεν αλλάζουν καθώς εκτελείται το πρόγραμμά σας.
- Οι μεταβλητές επιτρέπουν στο πρόγραμμά σας να διατηρεί τιμές που μπορούν να αλλάξουν καθώς εκτελείται το πρόγραμμα. Οι μεταβλητές έχουν ονόματα που σας επιτρέπουν να αναφέρετε τις τιμές τους. Ο παρακάτω κώδικας εκχωρεί μια ακέραια τιμή σε μια μεταβλητή με όνομα "a" και μια τιμή συμβολοσειράς σε μια μεταβλητή με όνομα "b".

```
In [4]: a = 50
        b = "πενήντα"
        print(a)
        print(b)

        50
        πενήντα
```

Το βασικό χαρακτηριστικό των μεταβλητών είναι ότι μπορούν να αλλάξουν. Ο παρακάτω κώδικας δείχνει πώς να αλλάξετε τις τιμές που διατηρούνται από τις μεταβλητές.

```
In [5]: a = 50
        print(a)
        a = a + 1
        print(a)

        50
        51
```

- Μπορείτε να αναμίξετε συμβολοσειρές και μεταβλητές για εκτύπωση. Αυτή η τεχνική ονομάζεται μορφοποιημένη ή παρεμβαλλόμενη συμβολοσειρά. Οι μεταβλητές πρέπει να βρίσκονται μέσα στα {}. Στην Python, αυτός ο τύπος συμβολοσειράς ονομάζεται γενικά f-string. Η συμβολοσειρά f συμβολίζεται με την τοποθέτηση ενός "f". Ο παρακάτω κώδικας δείχνει τη χρήση μιας συμβολοσειράς f για την ανάμειξη πολλών μεταβλητών με μια κυριολεκτική συμβολοσειρά.

```
In [6]: a = 50
        print(f'The value of a is {a}')

        The value of a is 50
```

- Μπορείτε επίσης να χρησιμοποιήσετε συμβολοσειρές f με μαθηματικά (που ονομάζεται έκφραση). Τα {} μπορούν να περικλείουν οποιαδήποτε έγκυρη έκφραση Python για εκτύπωση. Ο ακόλουθος κώδικας δείχνει τη χρήση μιας έκφρασης μέσα στα {} μιας συμβολοσειράς f.

```
In [7]: a = 50
        print(f'The value of a plus 5 is {a+5}')

        The value of a plus 5 is 55
```

```
In [8]: print(f'a is {a}')

        a is 50
```

- Μπορείτε να χρησιμοποιήσετε τις δηλώσεις if για να εκτελέσετε τη λογική. Προσέξτε τις εσοχές. Αυτές οι δηλώσεις if είναι ο τρόπος με τον οποίο η Python ορίζει μπλοκ κώδικα για να εκτελεστούν μαζί.
- Ένα μπλοκ αρχίζει συνήθως μετά από : και περιλαμβάνει οποιεσδήποτε γραμμές στο ίδιο επίπεδο εσοχής. Σε αντίθεση με πολλές άλλες γλώσσες προγραμματισμού, η Python χρησιμοποιεί κενό χώρο για να ορίσει μπλοκ κώδικα.

```
In [9]: a = 22
        if a>22:
            print('The variable a is greater than 22.')
        else:
            print('The variable a is not greater than 22')

The variable a is not greater than 22
```

- Είναι επίσης σημαντικό να σημειωθεί ότι ο τελεστής διπλό ίσο ("==") χρησιμοποιείται για τον έλεγχο της ισότητας δύο παραστάσεων. Το απλό ίσο ("=") χρησιμοποιείται μόνο για την εκχώρηση τιμών σε μεταβλητές στην Python. Το μεγαλύτερο από (">"), μικρότερο από ("<"), μεγαλύτερο από ή ίσο (">="), μικρότερο ή ίσο ("<=") όλα λειτουργούν όπως θα ήταν γενικά αποδεκτό. Ο έλεγχος για ανισότητα εκτελείται με τον τελεστή όχι ίσο ("!=").

```
In [10]: a = 22
         if a>=22:
             print('The variable a is greater or equal to 22.')
         else:
             print('The variable a is not greater or equal 22')

The variable a is greater or equal to 22.
```

- Είναι σύνηθες στις γλώσσες προγραμματισμού να κάνουν βρόχο σε μια σειρά αριθμών. Η Python το επιτυγχάνει αυτό με τη χρήση της λειτουργίας range. Εδώ μπορείτε να δείτε έναν βρόχο for και μια λειτουργία εύρους που κάνει το πρόγραμμα να κάνει βρόχο μεταξύ 1 και 9.

```
In [11]: for y in range(1, 10):
             print(y)

1
2
3
4
5
6
7
8
9
```

- Η εντολή range χρησιμοποιείται σε συνδυασμό με βρόχους για να περάσει πάνω από ένα συγκεκριμένο εύρος αριθμών.
- Το παρακάτω είναι ένα περαιτέρω παράδειγμα εκτύπωσης σε βρόχο συμβολοσειρών και αριθμών.

```
In [12]: ts = 0
         for x in range(1, 10):
             ts += x
             print(f"Adding {x}, sum so far is {ts}")

         print(f"Final sum: {ts}")

Adding 1, sum so far is 1
Adding 2, sum so far is 3
Adding 3, sum so far is 6
Adding 4, sum so far is 10
Adding 5, sum so far is 15
Adding 6, sum so far is 21
Adding 7, sum so far is 28
Adding 8, sum so far is 36
Adding 9, sum so far is 45
Final sum: 45
```

III. Λίστες, Λεξικά, Πλειάδες και Σύνολα

Στο μάθημα θα επικεντρωθούμε κυρίως σε Λίστες, Σύνολα και Λεξικά. Είναι σημαντικό να κατανοήσουμε τις διαφορές μεταξύ αυτών των τριών θεμελιωδών τύπων συλλογής.

- Λεξικό (**Dictionary)** - Ένα λεξικό είναι μια μεταβαλλόμενη μη διατεταγμένη συλλογή που ευρετηριάζει με ζεύγη ονομάτων και τιμών.
- Λίστα (**List)** - Μια λίστα είναι μια μεταβαλλόμενη διατεταγμένη συλλογή στοιχείων, που επιτρέπει να είναι διπλότυπα.
- Σύνολο (**Set)** - Ένα σύνολο είναι μια μεταβαλλόμενη μη διατεταγμένη συλλογή μοναδικών (χωρίς διπλότυπα) και αμετάβλητων (immutable) στοιχείων.
- Πλειάδα (**Tuple)** - Μια πλειάδα είναι μια αμετάβλητη διατεταγμένη συλλογή στοιχείων, που επιτρέπει να είναι διπλότυπα.

- Οι λίστες και οι πλειάδες είναι πολύ παρόμοιες στην Python και συχνά συγχέονται. Η σημαντική διαφορά είναι ότι μια λίστα είναι μεταβαλλόμενη, αλλά μια πλειάδα δεν είναι. Έτσι, περιλαμβάνουμε μια λίστα όταν θέλουμε να περιέχει παρόμοια αντικείμενα και περιλαμβάνουμε μια πλειάδα όταν γνωρίζουμε ποιες πληροφορίες περιλαμβάνονται σε αυτήν εκ των προτέρων.

- Πολλές γλώσσες προγραμματισμού περιέχουν μια συλλογή δεδομένων που ονομάζεται πίνακας (array). Ο τύπος πίνακα απουσιάζει αισθητά στην Python. Γενικά, ο προγραμματιστής θα χρησιμοποιήσει μια λίστα στη θέση ενός πίνακα στην Python. Η λίστα Python είναι πολύ πιο ευέλικτη καθώς το πρόγραμμα μπορεί να αλλάζει δυναμικά το μέγεθος μιας λίστας.

Λίστες και πλειάδες

- Οι λίστες και οι πλειάδες διαθέτουν μια διατεταγμένη συλλογή αντικειμένων.
- Η κύρια διαφορά που θα δείτε συντακτικά είναι ότι μια λίστα περικλείεται με τετράγωνα άγκιστρα [] και μια πλειάδα περικλείεται από παρένθεση (). Ο παρακάτω κώδικας ορίζει τόσο τη λίστα όσο και την πλειάδα.

```
In [1]: l = ['α', 'β', 'γ', 'δ']
        t = ('α', 'β', 'γ', 'δ')

        print(l)
        print(t)

['α', 'β', 'γ', 'δ']
('α', 'β', 'γ', 'δ')
```

- Η Python έχει μια δήλωση *for*. Αυτή η δήλωση σας επιτρέπει να κάνετε βρόχο σε κάθε στοιχείο μιας συλλογής, όπως μια λίστα ή μια πλειάδα.

```
In [2]: for s in l:
            print(s)
```

```
α
β
γ
δ
```

- Η συνάρτηση *enumerate* είναι χρήσιμη για την απαρίθμηση σε μια συλλογή και για την πρόσβαση στο ευρετήριο του στοιχείου στο οποίο βρισκόμαστε αυτήν τη στιγμή.

```
In [3]: for i,l in enumerate(l):
            print(f"{i}:{l}")
```

```
0:α
1:β
2:γ
3:δ
```

- Σε μια λίστα μπορεί να προστεθούν πολλά αντικείμενα, όπως συμβολοσειρές. Επιτρέπονται διπλότυπες τιμές. Οι πλειάδες δεν επιτρέπουν στο πρόγραμμα να προσθέσει επιπλέον αντικείμενα μετά τον ορισμό.

```
In [4]: A = []
        A.append('α')
        A.append('β')
        A.append('γ')
        A.append('δ')
        print(A)
```

```
['α', 'β', 'γ', 'δ']
```

- Οι ταξινομημένες συλλογές, όπως λίστες και πλειάδες, σας επιτρέπουν να έχετε πρόσβαση σε ένα στοιχείο με βάση τον αριθμό ευρετηρίου του, όπως γίνεται στον παρακάτω κώδικα. Οι μη ταξινομημένες συλλογές, όπως λεξικά και σύνολα, δεν επιτρέπουν στο πρόγραμμα να έχει πρόσβαση σε αυτά με αυτόν τον τρόπο.

```
In [5]: print(A[1])
β
```

- Σε μια λίστα μπορεί να προστεθούν πολλά αντικείμενα, όπως συμβολοσειρές. Επιτρέπονται διπλότυπες τιμές. Οι πλειάδες δεν επιτρέπουν στο πρόγραμμα να προσθέσει επιπλέον αντικείμενα μετά τον ορισμό. Για τη συνάρτηση εισαγωγής, ένα ευρετήριο, ο προγραμματιστής πρέπει να καθορίσει ένα ευρετήριο. Αυτές οι λειτουργίες δεν επιτρέπονται για πλειάδες επειδή θα οδηγούσαν σε αλλαγή.

```
In [6]: # Insert
        A = ['α', 'β', 'γ']
        A.insert(0, 'α0')
        print(A)
        # Remove
        A.remove('β')
        print(A)
        # Remove at index
        del A[0]
        print(A)
['α0', 'α', 'β', 'γ']
['α0', 'α', 'γ']
['α', 'γ']
```

Σύνολα

- Ένα σύνολο περιέχει μια μη ταξινομημένη συλλογή αντικειμένων, αλλά τα σύνολα δεν επιτρέπουν διπλότυπα. Εάν ένα πρόγραμμα προσθέσει ένα διπλότυπο στοιχείο σε ένα σύνολο, μόνο ένα αντίγραφο κάθε στοιχείου παραμένει στη συλλογή. Η προσθήκη ενός διπλότυπου στοιχείου σε ένα σύνολο δεν οδηγεί σε σφάλμα. Οποιαδήποτε από τις παρακάτω τεχνικές θα ορίσει ένα σύνολο.

```
In [7]: s = set()
        s = { 'α', 'β', 'γ'}
        s = set(['α', 'β', 'γ'])
        print(s)

        {'γ', 'β', 'α'}
```

- Μια λίστα περικλείεται πάντα σε τετράγωνα άγκιστρα [], μια πλειάδα σε παρένθεση (), και ένα σύνολο σε {}. Τα προγράμματα μπορούν να προσθέτουν στοιχεία σε ένα σύνολο καθώς εκτελούνται.

```
In [8]: # Μη αυτόματη προσθήκη στοιχείων, τα σύνολα δεν επιτρέπουν διπλότυπα
        # Τα σύνολα προσθέτουν (add), ενώ οι λίστες όπως είδαμε επεκτείνουν (append).
        B = set()
        B.add('α')
        B.add('β')
        B.add('γ')
        B.add('δ')
        print(B)

        {'γ', 'δ', 'β', 'α'}
```

Λεξικά

- Η Python παρέχει ένα λεξικό, που είναι ουσιαστικά μια συλλογή ζευγών ονόματος-τιμής το οποίο ορίζετα χρησιμοποιώντας {}.

```
In [9]: d = {'name': "Thanos", 'age':"20"}
        print(d)
        print(d['name'])

        if 'name' in d:
            print("Name is defined")

        if 'address' in d:
            print("address defined")
        else:
            print("address undefined")
```

```
{'name': 'Thanos', 'age': '20'}
Thanos
Name is defined
address undefined
```

- Προσέξτε να μην επιχειρήσετε να αποκτήσετε πρόσβαση σε ένα απροσδιόριστο κλειδί, καθώς αυτό θα οδηγήσει σε σφάλμα. Μπορείτε να ελέγξετε εάν έχει οριστεί ένα κλειδί, όπως φαίνεται παραπάνω. Μπορείτε επίσης να αποκτήσετε πρόσβαση στον κατάλογο και να δώσετε μια προεπιλεγμένη τιμή, όπως δείχνει ο ακόλουθος κώδικας.

```
In [10]: d.get('unknown_key', 'default')
Out[10]: 'default'
```

- Μπορείτε επίσης να αποκτήσετε πρόσβαση στα μεμονωμένα κλειδιά και τις τιμές ενός λεξικού.

```
In [11]: d = {'name': "Thanos", 'age':"20"}
         # All of the keys
         print(f"Key: {d.keys()}")

         # All of the values
         print(f"Values: {d.values()}")
```

```
Key: dict_keys(['name', 'age'])
Values: dict_values(['Thanos', '20'])
```

- Τα λεξικά και οι λίστες μπορούν να συνδυαστούν. Τα λεξικά και οι λίστες μαζί είναι ένας καλός τρόπος για τη δημιουργία πολύ περίπλοκων δομών δεδομένων. Ο παρακάτω κώδικας δείχνει μια υβριδική χρήση λεξικών και λιστών.

```
In [12]: # Python list & map structures
         customers = [
             {"name": "Thanos S.", "pets": ["A", "B"]},
             {"name": "Helen K. ", "pets": ["A"]},
             {"name": "Maria P."}
         ]

         print(customers)

         for customer in customers:
             print(f"{customer['name']}:{customer.get('pets', 'no pets')}")
```

```
[{'name': 'Thanos S.', 'pets': ['A', 'B']}, {'name': 'Helen K. ', 'pets': ['A']}, {'name': 'Maria P.'}]
Thanos S.:['A', 'B']
Helen K. :['A']
Maria P.:no pets
```

Σύνθετες λίστες

- Πολλές προηγμένες λειτουργίες είναι διαθέσιμες για λίστες. Μια τέτοια λειτουργία είναι το zip. Δύο λίστες μπορούν να συνδυαστούν σε μια ενιαία λίστα με την εντολή zip. Ο παρακάτω κώδικας δείχνει την εντολή zip.

```
In [1]: a = [1,2,3,4,5]
        b = [5,4,3,2,1]

        print(zip(a,b))
```

```
<zip object at 0x0000011169F58EC0>
```

- Για να δούμε τα αποτελέσματα της συνάρτησης zip, μετατρέπουμε το επιστρεφόμενο αντικείμενο zip στη λίστα. Μπορείτε να δείτε, η συνάρτηση zip επιστρέφει μια λίστα πλειάδων. Κάθε πλειάδα αντιπροσωπεύει ένα ζεύγος στοιχείων. Η σειρά στις δύο λίστες διατηρήθηκε.

```
In [2]: a = [1,2,3,4,5]
        b = [5,4,3,2,1]

        print(list(zip(a,b)))
```

```
[(1, 5), (2, 4), (3, 3), (4, 2), (5, 1)]
```

- Η συνήθης μέθοδος για τη χρήση της εντολής zip είναι μέσα σε έναν βρόχο for. Ο παρακάτω κώδικας δείχνει πώς ένας βρόχος for μπορεί να εκχωρήσει μια μεταβλητή σε κάθε συλλογή που επαναλαμβάνει το πρόγραμμα.

```
In [3]: a = [1,2,3,4,5]
        b = [5,4,3,2,1]

        for x,y in zip(a,b):
            print(f'{x} - {y}')
```

```
1 - 5
2 - 4
3 - 3
4 - 2
5 - 1
```

- Συνήθως, και οι δύο συλλογές θα έχουν το ίδιο μήκος όταν περάσουν στην εντολή zip. Δεν είναι λάθος να έχουμε συλλογές διαφορετικού μήκους. Όπως δείχνει ο ακόλουθος κώδικας, η εντολή zip θα επεξεργάζεται στοιχεία μόνο μέχρι το μήκος της μικρότερης συλλογής.

```
In [4]: a = [1,2,3,4,5]
        b = [5,4,3]

        print(list(zip(a,b)))
```

```
[(1, 5), (2, 4), (3, 3)]
```

- Μερικές φορές μπορεί να θέλετε να γνωρίζετε το τρέχον αριθμητικό ευρετήριο όταν ένας βρόχος for επαναλαμβάνεται μέσω μιας διατεταγμένης συλλογής. Χρησιμοποιήστε την εντολή enumerate για να παρακολουθήσετε τη θέση ευρετηρίου για ένα στοιχείο συλλογής. Επειδή η εντολή enumerate ασχολείται με αριθμητικά ευρετήρια της συλλογής, η εντολή zip θα εκχωρήσει αυθαίρετα ευρετήρια σε στοιχεία από μη ταξινομημένες συλλογές.

- Σκεφτείτε πώς μπορείτε να κατασκευάσετε ένα πρόγραμμα Python για να αλλάξετε κάθε στοιχείο μεγαλύτερο από 5 στην τιμή του 5. Το παρακάτω πρόγραμμα εκτελεί αυτόν τον μετασχηματισμό. Η εντολή enumerate επιτρέπει στον βρόχο να γνωρίζει σε ποιο ευρετήριο στοιχείων βρίσκεται αυτήν τη στιγμή, επιτρέποντας έτσι στο πρόγραμμα να μπορεί να αλλάξει την τιμή του τρέχοντος στοιχείου της συλλογής.

```
In [5]: a = [2, 10, 3, 11, 10, 3, 2, 1]
        for i, x in enumerate(a):
            if x>5:
                a[i] = 5
        print(a)

        [2, 5, 3, 5, 5, 3, 2, 1]
```

- Η εντολή κατανόησης (comprehension command) μπορεί να δημιουργήσει δυναμικά μια λίστα. Η παρακάτω κατανόηση μετράει από το 0 έως το 9 και προσθέτει κάθε τιμή (πολλαπλασιασμένη επί 10) σε μια λίστα.

```
In [6]: lst = [x*10 for x in range(10)]
        print(lst)

        [0, 10, 20, 30, 40, 50, 60, 70, 80, 90]
```

- Ένα λεξικό μπορεί επίσης να είναι μια κατανόηση. Η γενική μορφή για αυτό είναι:

dict_variable = {key:value for (key,value) in dictonary.items()}

- Μια κοινή χρήση για αυτό είναι η δημιουργία ενός ευρετηρίου για συμβολικά ονόματα στηλών.

```
In [7]: text = ['col-zero','col-one', 'col-two', 'col-three']
        lookup = {key:value for (value,key) in enumerate(text)}
        print(lookup)

        {'col-zero': 0, 'col-one': 1, 'col-two': 2, 'col-three': 3}
```

- Αυτό μπορεί να χρησιμοποιηθεί για την εύκολη εύρεση του ευρετηρίου μιας στήλης με βάση το όνομα.

```
In [8]: print(f'The index of "col-two" is {lookup["col-two"]}')

        The index of "col-two" is 2
```

IV. Χειρισμός αρχείων

Υπάρχουν πολλοί διαφορετικοί τύποι αρχείων που πρέπει να επεξεργαστείτε. Μερικοί από αυτούς τους τύπους αρχείων παρατίθενται εδώ:

- **Τα αρχεία CSV** (γενικά έχουν την επέκταση .csv) περιέχουν δεδομένα πίνακα που μοιάζουν με δεδομένα υπολογιστικού φύλλου (excel).
- **Τα αρχεία εικόνων** (γενικά με την επέκταση .png ή .jpg) περιέχουν εικόνες για όραση υπολογιστή.
- **Τα αρχεία κειμένου** (συχνά έχουν την επέκταση .txt) περιέχουν μη δομημένο κείμενο και είναι απαραίτητα για την επεξεργασία φυσικής γλώσσας.
- **Το JSON** (συχνά έχει την επέκταση .json) περιέχει ημι-δομημένα δεδομένα κειμένου σε μορφή κειμένου με δυνατότητα ανάγνωσης από τον άνθρωπο.
- **Τα αρχεία ήχου** (συχνά έχουν επέκταση όπως .au ή .wav) περιέχουν ηχογραφημένο ήχο.

Read a CSV File

- Τα προγράμματα Python μπορούν να διαβάσουν αρχεία CSV με Pandas (λεπτομέρειες για το Pandas στην επόμενη ενότητα). Η γενική τους μορφή είναι:

```
]: import pandas as pd

   df = pd.read_csv("https://raw.githubusercontent.com/thanossakkas/data/main/GDP_EU.csv")
   df
```

]:

	TIME	AUT	BEL	CZE
0	1970	3829.731721	3876.367950	NaN
1	1971	4210.441192	4210.433244	NaN
2	1972	4638.534351	4606.148069	NaN
3	1973	5103.430018	5140.661848	NaN
4	1974	5772.158418	5820.638690	NaN
5	1975	6300.543674	6258.089288	NaN
6	1976	6963.477714	6964.342545	NaN
7	1977	7768.883976	7435.042458	NaN
8	1978	8304.538089	8177.509638	NaN
9	1979	9491.450941	9056.835512	NaN
10	1980	10527.998100	10306.817050	NaN
11	1981	11478.234850	11242.562010	NaN
12	1982	12423.597360	11998.441080	NaN
13	1983	13315.535430	12500.938120	NaN
14	1984	13803.863450	13270.032670	NaN
15	1985	14589.375570	13912.992310	NaN
16	1986	15216.037180	14448.962580	NaN
17	1987	15793.916420	15130.905660	NaN
18	1988	16866.050590	16348.194070	NaN
19	1989	18126.710730	17514.815960	NaN
20	1990	19473.504510	18687.891450	12689.40152
21	1991	20618.036940	19599.302470	11655.62003
22	1992	21293.741180	20269.731530	11850.36547
23	1993	21733.353720	20471.138380	12123.72874
24	1994	22643.344320	21518.981950	12736.02198
25	1995	23698.629270	22447.550310	13855.63267
26	1996	24561.160500	22744.399560	14689.90856

Read a Text File

In [1]:
```
import urllib.request
import codecs
url = "https://raw.githubusercontent.com/thanossakkas/data/main/ithaki.txt"
with urllib.request.urlopen(url) as urlstream:
    for line in codecs.iterdecode(urlstream, 'utf-8'):
        print(line.rstrip())
```

```
ITHAKI
(I have used the greek written form of Ulysse's island:
Ithaki (I'thaki) -note that the accent is on the second syllable.)
When you set out on the way to Ithaki
wish for a long, eventful journey,
full of adventure, full of understanding.
Of Laistrygonians and Cyclops,
the raging Poseidon, never be afraid.
You'll never find such things on your way
if all your thoughts remain noble, and if your spirit
and body are touched by worthy emotion.
The Laistrygonians and Cyclops,
the fierce Poseidon, will never be encountered
if they're not carried in your soul, and if your soul
does not erect them on your path, before you.
```

Read an Image

Μπορείτε να χρησιμοποιήσετε το πακέτο Python PIL για επεξεργασία εικόνας. Ο παρακάτω κώδικας δείχνει πώς να φορτώσετε μια εικόνα από μια διεύθυνση URL και να την εμφανίσετε.

In [1]:
```
%matplotlib inline
from PIL import Image
import requests
from io import BytesIO

url = "https://upload.wikimedia.org/wikipedia/en/a/a9/Example.jpg"

response = requests.get(url)
img = Image.open(BytesIO(response.content))

img
```

Out[1]:

V. Συναρτήσεις

- Η συνάρτηση ορίζεται με το def.
- Μπορείτε να ορίσεται μια συνάρτηση με τις παραμέτρους της. Οι παράμετροι συνάρτησης μπορούν να ονομαστούν

```
In [1]: def say_hello(teacher, person_to_greet, greeting = "Hello"):
            print(f'{greeting} {person_to_greet}, this is {teacher}.')

        say_hello('Panos', "George")
        say_hello('Panos', "George", "Goodnight")
        say_hello(teacher="Panos", person_to_greet="George", greeting = "Goodnight")

        Hello George, this is Panos.
        Goodnight George, this is Panos.
        Goodnight George, this is Panos.
```

- ή να μην ονομαστούν

```
In [2]: def fun():
          print("Welcome to GFG")
```

- Επίσης μπορείτε να την καλείτε αρκετά εύκολα.

```
In [3]: fun()

        Welcome to GFG
```

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Python για Μηχανική Μάθηση

Μέθοδοι Μηχανικής Μάθησης στα Χρηματοοικονομικά

Αθανάσιος Σάκκας, ΟΠΑ

I.Εισαγωγή στο Pandas

- Το Pandas είναι μια βιβλιοθήκη ανοιχτού κώδικα που παρέχει υψηλής απόδοσης, εύχρηστες δομές δεδομένων και εργαλεία ανάλυσης δεδομένων για τη γλώσσα προγραμματισμού Python. Βασίζεται στην έννοια του πλαισίου δεδομένων (data frame).
- Το πλαίσιο δεδομένων είναι ένα κρίσιμο συστατικό των Pandas.
- Ο ακόλουθος κώδικας φορτώνει το σύνολο δεδομένων σε ένα πλαίσιο δεδομένων:

Αρχείο 2.1 Introduction to Pandas

In [2]:
```
#Simple dataframe
import os
import pandas as pd

raw = pd.read_csv("https://raw.githubusercontent.com/thanossakkas/data/main/Demographic_S
# check the raw data
print("Size of the dataset (row, col): ", raw.shape)
print("\nFirst 5 rows\n", raw.head(n=5))
print("\nFirst 5 rows and 5 columns\n",raw .iloc[:5 , :5])
```

```
Size of the dataset (row, col):  (236, 46)

First 5 rows
    JURISDICTION NAME  COUNT PARTICIPANTS  COUNT FEMALE  PERCENT FEMALE  \
0              10001                  44            22            0.50
1              10002                  35            19            0.54
2              10003                   1             1            1.00
3              10004                   0             0            0.00
4              10005                   2             2            1.00

   COUNT MALE  PERCENT MALE  COUNT GENDER UNKNOWN  PERCENT GENDER UNKNOWN  \
0          22          0.50                     0                       0
1          16          0.46                     0                       0
2           0          0.00                     0                       0
3           0          0.00                     0                       0
4           0          0.00                     0                       0

   COUNT GENDER TOTAL  PERCENT GENDER TOTAL  ...  COUNT CITIZEN STATUS TOTAL  \
0                  44                   100  ...                          44
1                  35                   100  ...                          35
2                   1                   100  ...                           1
3                   0                     0  ...                           0
4                   2                   100  ...                           2

   PERCENT CITIZEN STATUS TOTAL  COUNT RECEIVES PUBLIC ASSISTANCE  \
0                           100                               20
1                           100                                2
2                           100                                0
```

```
First 5 rows and 5 columns
    JURISDICTION NAME  COUNT PARTICIPANTS  COUNT FEMALE  PERCENT FEMALE  \
0               10001                  44            22            0.50
1               10002                  35            19            0.54
2               10003                   1             1            1.00
3               10004                   0             0            0.00
4               10005                   2             2            1.00

   COUNT MALE
0          22
1          16
2           0
3           0
4           0
```

- Η λειτουργία **display** παρέχει «καθαρότερη» εικόνα από την απλή εκτύπωση του πλαισίου δεδομένων. Ο καθορισμός των μέγιστων γραμμών και στηλών σας επιτρέπει να επιτύχετε μεγαλύτερο έλεγχο στην οθόνη.

```
In [3]: pd.set_option('display.max_columns', 5)
        pd.set_option('display.max_rows', 5)
        display(raw)
```

	JURISDICTION NAME	COUNT PARTICIPANTS	...	COUNT PUBLIC ASSISTANCE TOTAL	PERCENT PUBLIC ASSISTANCE TOTAL
0	10001	44	...	44	100
1	10002	35	...	35	100
...	...	...	...	...	...
234	16091	0	...	0	0
235	20459	0	...	0	0

236 rows × 46 columns

- Υπολογισμός των Summary Statistics και Correlation matrix

```
In [4]: # print summary statistics
        print("\nSummary statistics\n", raw.describe())
        print("\nCorrelation matrix\n", raw.corr())
```

```
Summary statistics
        JURISDICTION NAME  COUNT PARTICIPANTS  ...  \
count         236.000000          236.000000  ...
mean        11127.173729           17.661017  ...
...                  ...                 ...  ...
75%         11422.250000           13.000000  ...
max         20459.000000          272.000000  ...

       COUNT PUBLIC ASSISTANCE TOTAL  PERCENT PUBLIC ASSISTANCE TOTAL
count                     236.000000                       236.000000
mean                       17.661017                        44.491525
...                              ...                              ...
75%                        13.000000                       100.000000
max                       272.000000                       100.000000

[8 rows x 46 columns]

Correlation matrix
```

- Είναι δυνατή η δημιουργία ενός δεύτερου πλαισίου δεδομένων για την εμφάνιση στατιστικών πληροφοριών σχετικά με το πρώτο πλαίσιο δεδομένων.

In [5]:
```
# Strip non-numerics
raw = raw.select_dtypes(include=['int', 'float'])

headers = list(raw.columns.values)
fields = []

for field in headers:
    fields.append({
        'name' : field,
        'mean': raw[field].mean(),
        'var': raw[field].var(),
        'sdev': raw[field].std()
    })

for field in fields:
    print(field)
```

```
{'name': 'JURISDICTION NAME', 'mean': 11127.17372881356, 'var': 1107612.671817526, 'sdev': 1052.431789627017}
{'name': 'COUNT PARTICIPANTS', 'mean': 17.661016949152543, 'var': 1873.135665344391, 'sdev': 43.279737352996854}
{'name': 'COUNT FEMALE', 'mean': 10.296610169491526, 'var': 794.6265416516422, 'sdev': 28.189120980471213}
{'name': 'PERCENT FEMALE', 'mean': 0.24398305084745767, 'var': 0.11134066354129064, 'sdev': 0.3336774843187515}
{'name': 'COUNT MALE', 'mean': 7.364406779661017, 'var': 356.48791922106005, 'sdev': 18.8808767036815}
{'name': 'PERCENT MALE', 'mean': 0.20101694915254234, 'var': 0.08745683375405687, 'sdev': 0.29573101588108214}
{'name': 'COUNT GENDER UNKNOWN', 'mean': 0.0, 'var': 0.0, 'sdev': 0.0}
{'name': 'PERCENT GENDER UNKNOWN', 'mean': 0.0, 'var': 0.0, 'sdev': 0.0}
{'name': 'COUNT GENDER TOTAL', 'mean': 17.661016949152543, 'var': 1873.135665344391, 'sdev': 43.279737352996854}
{'name': 'PERCENT GENDER TOTAL', 'mean': 44.49152542372882, 'var': 2480.1658853227614, 'sdev': 49.801263892824664}
{'name': 'COUNT PACIFIC ISLANDER', 'mean': 0.025423728813559324, 'var': 0.04190407500901555, 'sdev': 0.20470484852346696}
{'name': 'PERCENT PACIFIC ISLANDER', 'mean': 0.0002966101694915254, 'var': 5.443562928236557e-06, 'sdev': 0.00233144429356347
5}
{'name': 'COUNT HISPANIC LATINO', 'mean': 1.8559322033898304, 'var': 36.030219978362815, 'sdev': 6.002517803252466}
{'name': 'PERCENT HISPANIC LATINO', 'mean': 0.07983050847457628, 'var': 0.033005928597186936, 'sdev': 0.18167533843972036}
{'name': 'COUNT AMERICAN INDIAN', 'mean': 0.0211864406779661, 'var': 0.02933645870897932, 'sdev': 0.17127889160366294}
{'name': 'PERCENT AMERICAN INDIAN', 'mean': 0.001016949152542373, 'var': 0.0001717273710782533, 'sdev': 0.013104479046427343}
{'name': 'COUNT ASIAN NON HISPANIC', 'mean': 0.5254237288135594, 'var': 4.931265777136714, 'sdev': 2.2206453514995848}
{'name': 'PERCENT ASIAN NON HISPANIC', 'mean': 0.056567796610169474, 'var': 0.04006348900108193, 'sdev': 0.2001586595705565}
{'name': 'COUNT WHITE NON HISPANIC', 'mean': 12.190677966101696, 'var': 1593.9336999639365, 'sdev': 39.92409923797826}
{'name': 'PERCENT WHITE NON HISPANIC', 'mean': 0.17775423728813558, 'var': 0.12585408402452222, 'sdev': 0.3547591915997699}
```

- Αυτός ο κώδικας εξάγει μια λίστα λεξικών που περιέχουν αυτές τις στατιστικές πληροφορίες. Αυτές οι πληροφορίες μοιάζουν με τον κώδικα JSON.
- Το πρόγραμμα Python μπορεί να μετατρέψει αυτές τις πληροφορίες τύπου JSON σε πλαίσιο δεδομένων για καλύτερη εμφάνιση.

```
In [6]: pd.set_option('display.max_columns', 0)
        pd.set_option('display.max_rows', 0)
        raw2 = pd.DataFrame(fields)
        display(raw2)
```

	name	mean	var	sdev
0	JURISDICTION NAME	11127.173729	1.107613e+06	1052.431790
1	COUNT PARTICIPANTS	17.661017	1.873136e+03	43.279737
2	COUNT FEMALE	10.296610	7.946265e+02	28.189121
3	PERCENT FEMALE	0.243983	1.113407e-01	0.333677
4	COUNT MALE	7.364407	3.564879e+02	18.880888
5	PERCENT MALE	0.201017	8.745683e-02	0.295731
6	COUNT GENDER UNKNOWN	0.000000	0.000000e+00	0.000000
7	PERCENT GENDER UNKNOWN	0.000000	0.000000e+00	0.000000
8	COUNT GENDER TOTAL	17.661017	1.873136e+03	43.279737
9	PERCENT GENDER TOTAL	44.491525	2.480166e+03	49.801264
10	COUNT PACIFIC ISLANDER	0.025424	4.190408e-02	0.204705
11	PERCENT PACIFIC ISLANDER	0.000297	5.443563e-06	0.002333
...	...	...	...	...
34	COUNT CITIZEN STATUS UNKNOWN	0.000000	0.000000e+00	0.000000
35	PERCENT CITIZEN STATUS UNKNOWN	0.000000	0.000000e+00	0.000000
36	COUNT CITIZEN STATUS TOTAL	17.661017	1.873136e+03	43.279737
37	PERCENT CITIZEN STATUS TOTAL	44.487288	2.479698e+03	49.796563
38	COUNT RECEIVES PUBLIC ASSISTANCE	5.974576	2.823653e+02	16.803729
39	PERCENT RECEIVES PUBLIC ASSISTANCE	0.139195	5.197424e-02	0.227979
40	COUNT NRECEIVES PUBLIC ASSISTANCE	11.686441	8.625651e+02	29.369458
41	PERCENT NRECEIVES PUBLIC ASSISTANCE	0.305805	1.448576e-01	0.380602
42	COUNT PUBLIC ASSISTANCE UNKNOWN	0.000000	0.000000e+00	0.000000

Check for missing values

- Ο ευκολότερος τρόπος για να ελέγξετε για τιμές που λείπουν σε ένα πλαίσιο δεδομένων Pandas είναι μέσω της συνάρτησης isna(). Η συνάρτηση isna() επιστρέφει μια δυαδική τιμή (True ή False) εάν λείπει η τιμή της στήλης Pandas, επομένως εάν εκτελέσετε τη raw.isna() θα λάβετε πίσω ένα πλαίσιο δεδομένων που θα σας δείχνει ένα φόρτο δυαδικών τιμών.

```
In [9]: raw.isna().head()
```

Out[9]:

	JURISDICTION NAME	COUNT PARTICIPANTS	COUNT FEMALE	PERCENT FEMALE	COUNT MALE	PERCENT MALE	COUNT GENDER UNKNOWN	PERCENT GENDER UNKNOWN	COUNT GENDER TOTAL	PERCENT GENDER TOTAL	COUNT PACIFIC ISLANDER	PERCENT PACIFIC ISLANDER	CO HISP LA
0	False	False	False	False	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	False	False	False	

- Αυτό συνήθως δεν είναι πολύ χρήσιμο, επομένως θα υπολογίσουμε το sum() των τιμών που λείπουν εκτελώντας το raw.isna().sum(). Αυτό επιστρέφει τις στήλες στο πλαίσιο δεδομένων Pandas μαζί με τον αριθμό των τιμών που λείπουν που εντοπίστηκαν σε καθεμία, επομένως το 0 σημαίνει ότι δεν λείπουν τιμές και το 1 σημαίνει ότι λείπει μία τιμή.

```
In [17]: check = raw.isna().sum()
         check

Out[17]: JURISDICTION NAME                    0
         COUNT PARTICIPANTS                   0
         COUNT FEMALE                         0
         PERCENT FEMALE                       0
         COUNT MALE                           0
         PERCENT MALE                         0
         COUNT GENDER UNKNOWN                 0
         PERCENT GENDER UNKNOWN               0
         COUNT GENDER TOTAL                   0
         PERCENT GENDER TOTAL                 0
         COUNT PACIFIC ISLANDER               0
         PERCENT PACIFIC ISLANDER             0
         COUNT HISPANIC LATINO                0
         PERCENT HISPANIC LATINO              0
         COUNT AMERICAN INDIAN                0
                                              ..
         PERCENT US CITIZEN                   0
         COUNT OTHER CITIZEN STATUS           0
         PERCENT OTHER CITIZEN STATUS         0
         COUNT CITIZEN STATUS UNKNOWN         0
         PERCENT CITIZEN STATUS UNKNOWN       0
         COUNT CITIZEN STATUS TOTAL           0
         PERCENT CITIZEN STATUS TOTAL         0
         COUNT RECEIVES PUBLIC ASSISTANCE     0
         PERCENT RECEIVES PUBLIC ASSISTANCE   0
         COUNT NRECEIVES PUBLIC ASSISTANCE    0
         PERCENT NRECEIVES PUBLIC ASSISTANCE  0
         COUNT PUBLIC ASSISTANCE UNKNOWN      0
         PERCENT PUBLIC ASSISTANCE UNKNOWN    0
         COUNT PUBLIC ASSISTANCE TOTAL        0
         PERCENT PUBLIC ASSISTANCE TOTAL      0
         Length: 46, dtype: int64
```

- Ένας γρήγορος τρόπος για να το αποθηκεύσετε το Dataframe σε CSV είναι ο ακόλουθος

```
In [18]: check.to_csv('checkformissing.csv')
```

Missing Values

- Τα missing values που λείπουν είναι μια πραγματικότητα. Στην ιδανική περίπτωση, κάθε σειρά δεδομένων θα έχει τιμές για όλες τις στήλες. Ωστόσο, αυτό συμβαίνει σπάνια.
- Μια πρακτική είναι κάνεις drop με την εντολή dropna(). Δείτε το *2.2 dropnanvalues.ipynb*

```
In [1]: #Simple dataframe
import os
import pandas as pd
raw = pd.read_csv("https://raw.githubusercontent.com/thanossakkas/data/main/GDP_EU.csv")
# check the raw data
print("Size of the dataset (row, col): ", raw.shape)
print("\nFirst 5 rows\n", raw.head(n=5))
print("\nFirst 5 rows and 5 columns\n",raw .iloc[:5 , :5])

Size of the dataset (row, col):  (52, 4)

First 5 rows
    TIME          AUT          BEL  CZE
0  1970  3829.731721  3876.367950  NaN
1  1971  4210.441192  4210.433244  NaN
2  1972  4638.534351  4606.148069  NaN
3  1973  5103.430018  5140.661848  NaN
4  1974  5772.158418  5820.638690  NaN

First 5 rows and 5 columns
    TIME          AUT          BEL  CZE
0  1970  3829.731721  3876.367950  NaN
1  1971  4210.441192  4210.433244  NaN
2  1972  4638.534351  4606.148069  NaN
3  1973  5103.430018  5140.661848  NaN
4  1974  5772.158418  5820.638690  NaN

In [3]: rawwithoutnan= raw.dropna()

In [6]: rawwithoutnan.to_csv('rawwithoutnan.csv')
```

- Μια άλλη πρακτική είναι η αντικατάσταση των τιμών που λείπουν με τη διάμεσο για αυτήν τη στήλη. Ο ακόλουθος κώδικας αντικαθιστά τυχόν τιμές nan με τη διάμεσο:
- Δείτε το *2.3 mediannanvalues.ipynb*

In [5]:
```
#Simple dataframe
import os
import pandas as pd
raw = pd.read_csv("https://raw.githubusercontent.com/thanossakkas/data/main/GDP_EU_missing.csv")
raw
```

Με NaN

	TIME	AUT	BEL	CZE
0	1990	19473.50451	18687.89145	12689.40152
1	1991	20618.03694	19599.30247	11655.62003
2	1992	21293.74118	20269.73153	11850.36547
3	1993	21733.35372	20471.13838	12123.72874
4	1994	22643.34432	21518.98195	12736.02198
5	1995	23698.62927	22447.55031	13855.63267
6	1996	24561.16050	22744.39956	14689.90856
7	1997	25427.22674	23733.31705	14825.65829
8	1998	26676.24650	24370.04646	14977.57594
9	1999	27606.48424	25441.89139	15397.92282
10	2000	29380.03111	27789.05351	16210.13770
11	2001	29707.46226	28791.40584	17610.74958
12	2002	31178.05144	30281.66799	18245.66901
13	2003	NaN	30934.59859	19524.25779
14	2004	33784.43265	32063.67380	20912.04591
15	2005	35024.55748	33176.68088	22045.99888
16	2006	37659.84067	35253.92329	23855.58792
17	2007	39436.42013	36794.23420	NaN
18	2008	41316.02264	37883.23342	27853.54990
19	2009	40929.33675	37753.27627	27637.15866
20	2010	42020.55064	39837.99795	27768.00435
21	2011	44469.20964	40943.34348	28999.75476
22	2012	46477.65508	42290.47767	29258.90485
23	2013	47936.67796	43672.71229	30828.52641
24	2014	48813.53441	44929.93333	NaN
25	2015	49942.05629	46201.68589	33909.30924
26	2016	52665.08746	48599.20268	36101.28560

Αντικαθιστώντας τα NaN με τη διάμεσο της κάθε στήλης

	TIME	AUT	BEL	CZE
0	1990	19473.50451	18687.89145	12689.401520
1	1991	20618.03694	19599.30247	11655.620030
2	1992	21293.74118	20269.73153	11850.365470
3	1993	21733.35372	20471.13838	12123.728740
4	1994	22643.34432	21518.98195	12736.021980
5	1995	23698.62927	22447.55031	13855.632670
6	1996	24561.16050	22744.39956	14689.908560
7	1997	25427.22674	23733.31705	14825.658290
8	1998	26676.24650	24370.04646	14977.575940
9	1999	27606.48424	25441.89139	15397.922820
10	2000	29380.03111	27789.05351	16210.137700
11	2001	29707.46226	28791.40584	17610.749580
12	2002	31178.05144	30281.66799	18245.669010
13	2003	37659.84067	30934.59859	19524.257790
14	2004	33784.43265	32063.67380	20912.045910
15	2005	35024.55748	33176.68088	22045.998880
16	2006	37659.84067	35253.92329	23855.587920
17	2007	39436.42013	36794.23420	21479.022395
18	2008	41316.02264	37883.23342	27853.549900
19	2009	40929.33675	37753.27627	27637.158660
20	2010	42020.55064	39837.99795	27768.004350
21	2011	44469.20964	40943.34348	28999.754760
22	2012	46477.65508	42290.47767	29258.904850
23	2013	47936.67796	43672.71229	30828.526410
24	2014	48813.53441	44929.93333	21479.022395
25	2015	49942.05629	46201.68589	33909.309240
26	2016	52665.08746	48599.20268	36101.285600

Dealing with Outliers

- Οι ακραίες τιμές (outliers) είναι τιμές που είναι ασυνήθιστα υψηλές ή χαμηλές. Μερικές φορές οι ακραίες τιμές είναι απλώς σφάλματα, αυτό είναι αποτέλεσμα λάθους παρατήρησης. Οι ακραίες τιμές μπορεί επίσης να είναι πραγματικά μεγάλες ή μικρές τιμές που μπορεί να είναι δύσκολο να αντιμετωπιστούν.
- Επιλογές για την αντιμετώπισή τους:¶

α. Διορθώστε τα δεδομένα: Δείτε τα δεδομένα και διορθώστε τα. Μπορεί να είναι δαπανηρή ή αδύνατη αυτή η αντιμετώπιση.

β. Trimming: Διαγραφή παρατηρήσεων που είναι ακραίες.

γ.Winsorization: Αλλάξτε την τιμή έτσι ώστε να είναι πιο κοντά στην υπόλοιπη κατανομή

- Παράδειγμα: Οποιαδήποτε τιμή πάνω από το 99ο εκατοστημόριο για μια μεταβλητή αλλάζει ώστε να ισούται με το 99ο εκατοστημόριο.
- Αυτή είναι μια συνηθισμένη και χωρίς κόστος (ad-hoc) διόρθωση που υποβαθμίζει το βάρος της ακραίας τιμής στην ανάλυσή σας επειδή οι τιμές μειώνονται, χωρίς να απορρίπτεται εντελώς η παρατήρηση.
- Δύσκολη ερώτηση που εξαρτάται από τα δεδομένα/εφαρμογή: Ποιο είναι το «σωστό» ποσό της διανομής για winsorize.

Δείτε το αρχείο *2.4 removeoutliers.ipynb*

Removing duplicates

- **1η μέθοδος: Χρησιμοποιώντας *set()**

Καταργεί πρώτα τα διπλότυπα και επιστρέφει ένα λεξικό που πρέπει να μετατραπεί σε λίστα.

- **2η μέθοδος: Χρησιμοποιώντας collections.OrderedDict.fromkeys()**

Καταργεί πρώτα τα διπλότυπα και επιστρέφει ένα λεξικό που πρέπει να μετατραπεί σε λίστα. Αυτό λειτουργεί καλά και στην περίπτωση των strings .

- **3η μέθοδος: Χρησιμοποιώντας numpy unique method**

Note: Install numpy module using command "pip install numpy"

Αυτή η μέθοδος χρησιμοποιείται όταν η λίστα περιέχει στοιχεία του ίδιου τύπου και χρησιμοποιείται για την κατάργηση διπλότυπων από τη λίστα. Πρώτα μετατρέπει τη λίστα σε έναν πίνακα numpy και στη συνέχεια χρησιμοποιεί τη μέθοδο numpy unique() για να αφαιρέσει όλα τα διπλότυπα στοιχεία από τη λίστα.

Δείτε το αρχείο *2.5 removedupl.ipynb*

Μετατρέποντας ένα Dataframe σε Matrix

- Το πρόγραμμα χρησιμοποιεί την ιδιότητα values για να μετατρέψει τα δεδομένα σε μήτρα (matrix).
- Δείτε το αρχείο *2.6 dataframe2numeric.ipynb*

```
In [1]: #Simple dataframe
        import os
        import pandas as pd
        raw = pd.read_csv("https://raw.githubusercontent.com/thanossakkas/data/main/
```

```
In [2]: raw.values
```

```
Out[2]: array([[ 1990.     , 19473.50451, 18687.89145, 12689.40152],
               [ 1991.     , 20618.03694, 19599.30247, 11655.62003],
               [ 1992.     , 21293.74118, 20269.73153, 11850.36547],
               [ 1993.     , 21733.35372, 20471.13838, 12123.72874],
               [ 1994.     , 22643.34432, 21518.98195, 12736.02198],
               [ 1995.     , 23698.62927, 22447.55031, 13855.63267],
               [ 1996.     , 24561.1605 , 22744.39956, 14689.90856],
               [ 1997.     , 25427.22674, 23733.31705, 14825.65829],
               [ 1998.     , 26676.2465 , 24370.04646, 14977.57594],
               [ 1999.     , 27606.48424, 25441.89139, 15397.92282],
               [ 2000.     , 29380.03111, 27789.05351, 16210.1377 ],
               [ 2001.     , 29707.46226, 28791.40584, 17610.74958],
               [ 2002.     , 31178.05144, 30281.66799, 18245.66901],
               [ 2003.     ,         nan, 30934.59859, 19524.25779],
               [ 2004.     , 33784.43265, 32063.6738 , 20912.04591],
               [ 2005.     , 35024.55748, 33176.68088, 22045.99888],
               [ 2006.     , 37659.84067, 35253.92329, 23855.58792],
               [ 2007.     , 39436.42013, 36794.2342 ,         nan],
               [ 2008.     , 41316.02264, 37883.23342, 27853.5499 ],
               [ 2009.     , 40929.33675, 37753.27627, 27637.15866],
               [ 2010.     , 42020.55064, 39837.99795, 27768.00435],
               [ 2011.     , 44469.20964, 40943.34348, 28999.75476],
               [ 2012.     , 46477.65508, 42290.47767, 29258.90485],
               [ 2013.     , 47936.67796, 43672.71229, 30828.52641],
               [ 2014.     , 48813.53441, 44929.93333,         nan],
               [ 2015.     , 49942.05629, 46201.68589, 33909.30924],
               [ 2016.     , 52665.08746, 48599.20268, 36101.2856 ],
               [ 2017.     , 54188.36067, 50442.94752, 38842.89627],
               [ 2018.     , 56956.11056,         nan, 41157.37041],
               [ 2019.     , 59719.33165, 55800.82599, 44223.08816],
               [ 2020.     , 57253.30056, 54539.03253, 42813.74134],
               [ 2021.     , 59976.26467, 58806.11925, 44801.66688]])
```

Πηγές δεδομένων

- http://www.bls.gov - Bureau of Labor Statistics
- http://www.federalreserve.gov - Federal Reserve Board
- http://research.stlouisfed.org/fred2 - Federal Reserve Bank of St. Louis
- http://www.nationwide.co.uk/hpi/datadownload/data_download.htm - Nationwide
- http://www.oanda.com/convert/fxhistory - Oanda
- http://finance.yahoo.com - Yahoo! Finance
- http://www.dallasfed.org/ - Federal Reserve of Bank of Dallas
- http://www.bankofengland.co.uk/Pages/home.aspx - Bank of England
- https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html - Kenneth R. French - Data Library

- Βάσεις Δεδομένων με Συνδρομή: Thomson Reuters Eikon, Bloomberg, etc.

«Κατεβάζοντας» δεδομένα από το Yahoo Finance

Ανοίξτε το αρχείο *2.7 retrieve_yahoodata.ipynb*

In [29]:
```
from pandas_datareader import data as pdr
import yfinance as yf

yf.pdr_override()

stocks = ['msft', 'aapl', 'twtr', 'intc', 'tsm', 'goog', 'amzn', 'nvda']
start = datetime.datetime(2012,5,31)
end = datetime.datetime(2023,1,1)

yahoodata_specificperiod = pdr.get_data_yahoo(stocks, start=start, end=end)
yahoodata_maxperiod = pdr.get_data_yahoo(stocks,period="max")
```

```
[**********100%**********]  8 of 8 completed
[**********100%**********]  8 of 8 completed
```

In [30]:
```
yahoodata_specificperiod.to_csv('yahoodata_specificperiod.csv')
yahoodata_maxperiod.to_csv('yahoodata_fullperiod.csv')
```

«Κατεβάζοντας» δεδομένα από το Kenneth R. French - Data Library

- Δείτε το αρχείο *2.8 FF_factors.ipynb*

In [9]:
```
import urllib.request
import zipfile
ff_url = "http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/ftp/Europe_5_Factors_D
# Download the file and save it
# We will name it fama_french.zip file
urllib.request.urlretrieve(ff_url,'Europe_5_Factors_Daily_CSV.zip')
zip_file = zipfile.ZipFile('Europe_5_Factors_Daily_CSV.zip', 'r')
# Next we extact the file data
# We will call it ff_factors.csv
zip_file.extractall()
# Make sure you close the file after extraction
zip_file.close()
import pandas as pd
ff_factors = pd.read_csv('Europe_5_Factors_Daily.csv', skiprows = 3)
print(ff_factors.head())
```

```
   Unnamed: 0  Mkt-RF   SMB   HML   RMW   CMA    RF
0    19900702    0.99  0.00 -0.56  0.43 -0.25  0.03
1    19900703    0.33 -0.09  0.00  0.02  0.28  0.03
2    19900704    0.24  0.03 -0.19 -0.09  0.23  0.03
3    19900705   -0.64  0.22  0.08 -0.36  0.07  0.03
4    19900706    0.07 -0.24  0.13  0.02  0.11  0.03
```

In [10]:
```
print(ff_factors.iloc[1112:1120],)
```

```
      Unnamed: 0  Mkt-RF   SMB   HML   RMW   CMA    RF
1112    19941005   -1.32  0.83 -0.16  0.24 -0.09  0.02
1113    19941006    0.39 -0.57 -0.17  0.14  0.05  0.02
1114    19941007   -0.18 -0.26 -0.10  0.26 -0.05  0.02
1115    19941010    1.38 -1.24  0.40 -0.09 -0.01  0.02
1116    19941011    0.69 -0.74 -0.15  0.16 -0.09  0.02
1117    19941012    0.59 -0.05 -0.23  0.05 -0.10  0.02
1118    19941013    1.59 -0.91  0.20 -0.08 -0.05  0.02
1119    19941014    0.08  0.53  0.05 -0.17  0.01  0.02
```

In [11]:
```
ff_factors.to_csv('ff_factors.csv')
```

II. Categorical και Continuous Values

- Προτού εξετάσουμε συγκεκριμένους τρόπους επεξεργασίας δεδομένων, είναι σημαντικό να εξετάσουμε τέσσερις βασικούς τύπους δεδομένων:
- Character Data (strings)
  - **Nominal** - Individual discrete items, no order. Για παράδειγμα, color, zip code, shape.
  - **Ordinal** - Individual distinct items have an implied order. Για παράδειγμα grade level, job title, Starbucks coffee size (tall, vente, grande)
- Numeric Data
  - **Interval** - Numeric values, no defined start. Για παράδειγμα, temperature. Δε θα πείτε ποτέ, "yesterday was twice as hot as today."
  - **Ratio** - Numeric values, clearly defined start. Για παράδειγμα, speed. Θα πείτε "The first car is going twice as fast as the second."

- **Encoding Continuous Values**

Ένας κοινός μετασχηματισμός είναι η κανονικοποίηση normalization των εισόδων. Μερικές φορές είναι πολύτιμο για την κανονικοποίηση των αριθμητικών εισόδων να τίθενται σε τυπική μορφή, έτσι ώστε το πρόγραμμα να μπορεί εύκολα να συγκρίνει αυτές τις δύο τιμές. Σκεφτείτε αν κάποιος φίλος σας είπε ότι έλαβε έκπτωση 10 δολαρίων. Είναι αυτή μια καλή συμφωνία; Μπορεί. Αλλά το κόστος δεν κανονικοποιείται. Εάν ο φίλος σας αγόρασε ένα αυτοκίνητο, τότε η έκπτωση δεν είναι τόσο καλή. Εάν ο φίλος σας αγόρασε δείπνο, αυτή είναι μια εξαιρετική έκπτωση!

Τα ποσοστά είναι μια διαδεδομένη μορφή κανονικοποίησης. Εάν ο φίλος σας σας πει ότι έχει έκπτωση 10%, ξέρουμε ότι αυτή είναι μια καλύτερη έκπτωση από το 5%. Δεν έχει σημασία πόσο ήταν η τιμή αγοράς.

Μια ευρέως διαδεδομένη κανονικοποίηση μηχανικής μάθησης είναι το Z-Score:

$$\text{Z-score}: \quad \text{Value} \rightarrow \frac{\text{Value}-\text{Mean}}{\text{SD}}$$

- **Encoding Categorical Values as Dummies**

Το παραδοσιακό μέσο για την κωδικοποίηση κατηγορικών τιμών είναι η δημιουργία τους σε dummy variables.

- **Encoding Categorical Values as Ordinal**

Συνήθως οι κατηγορίες θα κωδικοποιούνται ως dummy variables. Ωστόσο, ενδέχεται να υπάρχουν άλλες τεχνικές για τη μετατροπή κατηγοριών σε αριθμητικές. Κάθε φορά που υπάρχει ένα order στις κατηγορίες, θα πρέπει να χρησιμοποιείται ένας αριθμός. Σκεφτείτε εάν είχατε μια κατηγορία που περιέγραφε το τρέχον επίπεδο εκπαίδευσης ενός ατόμου.

Kindergarten (0), First Grade (1), Second Grade (2), Third Grade (3), Fourth Grade (4), Fifth Grade (5), Sixth Grade (6), Seventh Grade (7), Eighth Grade (8), High School Freshman (9), High School Sophomore (10), High School Junior (11), High School Senior (12), College Freshman (13), College Sophomore (14), College Junior (15), College Senior (16), Graduate Student (17), PhD Candidate (18), Doctorate (19), Post Doctorate (20)

Η παραπάνω λίστα έχει 21 επίπεδα. Αυτό θα χρειαζόταν 21 dummy μεταβλητές. Ωστόσο, η απλή κωδικοποίηση αυτού σε dummy θα χάσει τις πληροφορίες του ordering. Ίσως η πιο εύκολη προσέγγιση θα ήταν να τους εκχωρήσετε απλώς τον αριθμό και να εκχωρήσετε στην κατηγορία έναν μοναδικό αριθμό που είναι ίσος με την τιμή στην παραπάνω παρένθεση. Ωστόσο, ίσως μπορέσουμε να τα πάμε ακόμα καλύτερα. Ο προπτυχιακός φοιτητής είναι πιθανό να υπερβαίνει το ένα έτος, επομένως μπορεί να αυξήσετε περισσότερο την τιμή που του αναλογεί.

III. Grouping, Sorting, and Shuffling

- Η **ταξινόμηση** του συνόλου δεδομένων σάς επιτρέπει να ταξινομήσετε τις σειρές με αύξουσα ή φθίνουσα σειρά για μία ή περισσότερες στήλες.

Δείτε το αρχείο *2.9 sorting.ipynb*

- Η **ομαδοποίηση** είναι μια τυπική λειτουργία σε σύνολα δεδομένων.

Δείτε το αρχείο *2.10 grouping.ipynb*

- **Ανακάτεμα** ενός συνόλου δεδομένων

Δείτε το αρχείο *2.11 shuffling.ipynb*