ΕΙΣΑΓΩΓΗ ΣΤΟΥΣ Η/Υ & ΕΦΑΡΜΟΓΕΣ

Σημειώσεις Εργαστηρίου: Εργαστηριακή Άσκηση 7

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

Περιεχόμενα

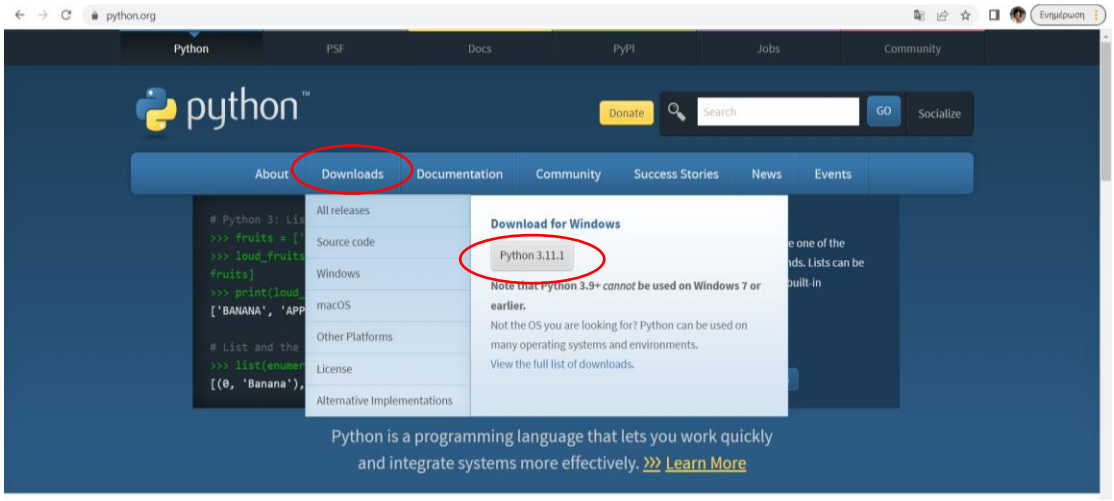
Στόχος

Ο στόχος της ? είναι η γνωριμία και εξοικείωση με τη γλώσσα προγραμματισμού Python.

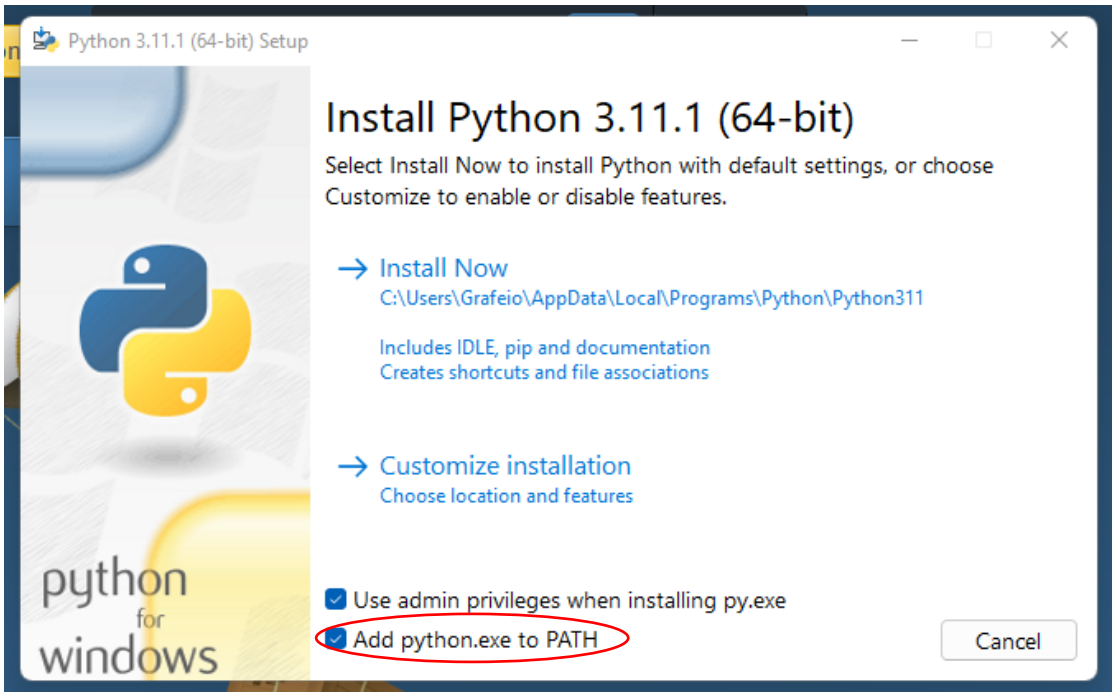
Θα πρέπει να εγκαταστήσετε τη γλώσσα Python, με ενσωματωμένο το περιβάλλον προγραμματισμού IDLE, καθώς και τη βιβλιοθήκη pandas. Η βιβλιοθήκη pandas θα μας επιτρέψει στην επόμενη Εργαστηριακή άσκηση να επεξεργαστούμε αρχεία.

Εγκατάσταση γλώσσας python (με IDLE)

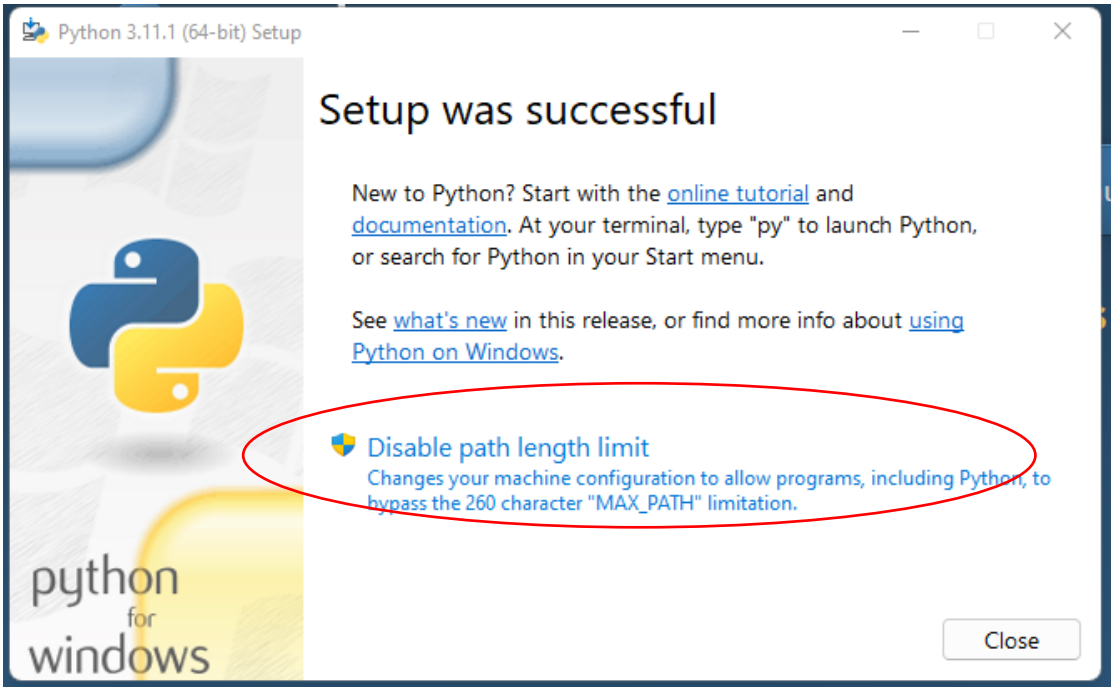
1. Επισκέπτομαι τη σελίδα http://www.python.org/



2. Στο ιστότοπο επιλέγω **Downloads** και μου εμφανίζει να επιλέξω για καταβίβαση την τελευταία έκδοση για το λειτουργικό μου (εδώ την έκδοση Python 3.11.1 για Windows, εσείς θα επιλέξετε την τελευταία έκδοση)
3. Καταβιβάζω και εκτελώ (κάνοντας κλικ) το αρχείο εγκατάστασης
4. Στο παράθυρο που εμφανίζεται ξεκινώντας η εγκατάσταση επιλέγω το «*Add python.exe to PATH*»



5. Προς το τέλος της εγκατάστασης **προσέχω** να επιλέξω την αλλαγή του μεγέθους του path (αν εμφανιστεί)

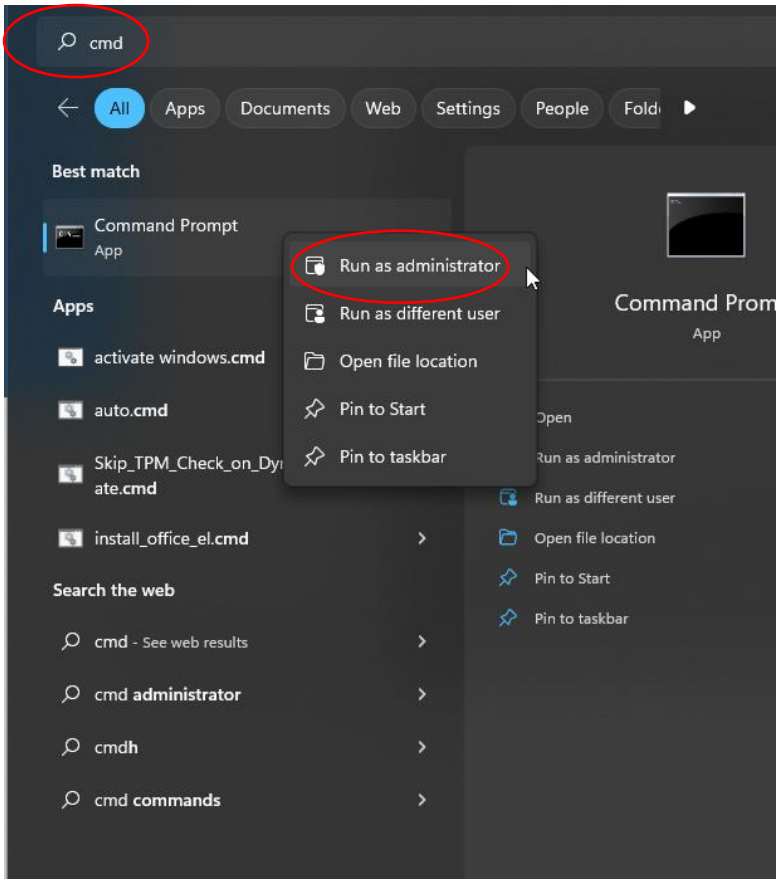


Εγκατάσταση βιβλιοθήκης python pandas

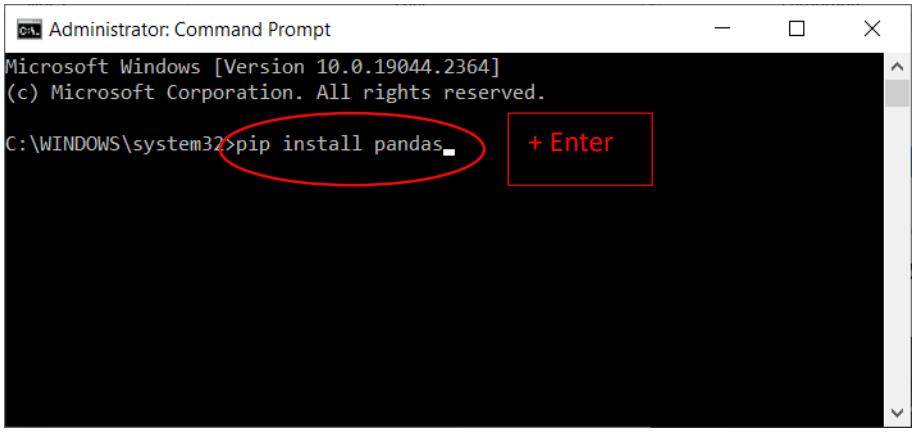
Από τις πιο δημοφιλείς βιβλιοθήκες χειρισμού και ανάλυσης δεδομένων στη γλώσσα Python είναι η βιβλιοθήκη **pandas** (**pan**el **da**ta **s**ystem).

Η βιβλιοθήκη pandas *δεν συμπεριλαμβάνεται στον πυρήνα της γλώσσας Python* και για να την εγκαταστήσουμε πρέπει να κάνουμε τα ακόλουθα:

1. Εκτελώ το πρόγραμμα Command Prompt **ως διαχειριστής** (Run as administrator) από το Start των windows γράφοντας cmd.



2. Στο παράθυρο του Command Prompt γράφω **pip install pandas** και πατώ Enter για να τρέξει, να κατεβάσει τα αρχεία της βιβλιοθήκης και να τα εγκαταστήσει.



Αφού καταβιβάσει τα απαραίτητα αρχεία και ολοκληρωθεί η εγκατάσταση χωρίς πρόβλημα, είμαστε έτοιμοι να την χρησιμοποιήσουμε.

Ονόματα μεταβλητών

Οι επιτρεπτοί χαρακτήρες για τη δημιουργία ονομάτων μεταβλητών είναι οι τα γράμματα, ψηφία, ή underscores (_). Τα ονόματα μεταβλητών ξεκινούν πάντα από γράμμα. Επίσης ως ονόματα μεταβλητών δεν μπορούμε να χρησιμοποιήσουμε τις λέξεις που έχει δεσμεύσει η γλώσσα για συγκεκριμένη συνάρτηση, πχ max.

Προσοχή: Η γλώσσα είναι **case-sensitive**, δηλαδή αντιλαμβάνεται διαφορά μεταξύ πεζών και κεφαλαίων χαρακτήρων.

Ερώτηση 1: Αντιστοιχήστε τον τύπο των μεταβλητών στη γλώσσα Python στις ακόλουθες εντολές ανάθεσης:

Τύποι μεταβλητών και κατηγορίες τους

Η python έχει αρκετούς τύπους μεταβλητών, που αρχικά κατηγοριοποιούνται με το αν είναι **Βαθμωτοί** (περιγράφονται πλήρως αναφέροντας μόνο μία τιμή), ή **Συλλογές** (έχουν ως περιεχόμενο αναφορές σε άλλα/πολλά στοιχεία).

Μία άλλη κατηγοριοποίηση των τύπων στην Python είναι αν είναι **στατικοί (immutable)**, δηλαδή δεν μπορείς να αλλάξεις την τιμή τους ή την τιμή ενός στοιχείου τους, ή **μεταβαλλόμενοι (mutable)**, δηλαδή μπορείς να αλλάζεις τις τιμές των στοιχείων τους. Οι βαθμωτοί τύποι είναι πάντα στατικοί.

Ο **τελεστής ανάθεσης** στην R είναι το ίσον (=).

Ο τύπος μίας μεταβλητής καθορίζεται από το είδος της τιμή που της εκχωρούμε κάθε φορά.

Βαθμωτοί

- **Αριθμητικοί τύποι:**
  - **int (Ακέραιοι):** Για ακέραιους αριθμούς.
    - Παράδειγμα: `x = 10`
  - **float (Δεκαδικοί αριθμοί):** Για αριθμούς κινητής υποδιαστολής (πραγματικοί αριθμοί).
    - Παράδειγμα: `x = 3.14`
  - **complex (Μιγαδικοί αριθμοί):** Για μιγαδικούς αριθμούς, όπου j είναι η φανταστική μονάδα.
    - Παράδειγμα: `z = 2 + 3j`
- **Λογικός τύπος:**
  - **bool:** Έχει δύο τιμές: True ή False.
    - Παράδειγμα: `is_active = True`

Συλλογές

Διατεταγμένες Συλλογές

- **str (Συμβολοσειρές, Strings):** Για **στατικά (immutable)** αλφαριθμητικά δεδομένα μέσα σε quotes (single ' or double quotes ")
  - Παράδειγμα: `name = "Python"`

- Υποστηρίζουν πολλαπλές λειτουργίες όπως σύνδεση, μορφοποίηση, και τεμαχισμό (slicing).

- **list (Λίστες):** Μία **μεταβαλλόμενη**, **διατεταγμένη** συλλογή τιμών (δυνητικά) διαφορετικού τύπου, χωριζόμενες με κόμμα, μέσα σε **[]** (square brackets).
  - Παράδειγμα: `aNiceList=[1,'Victoria Daskalou',54,1000.5]`
- **tuple (Πλειάδες):** Μία **στατική (immutable)**, **διατεταγμένη** συλλογή τιμών διαφορετικού τύπου, χωριζόμενες με κόμμα, μέσα σε **()** (παρενθέσεις)
  - Παράδειγμα: `aNiceTuple=(1,'Victoria Daskalou',54,1000.5)`
- **range:** Για ακολουθίες ακέραιων αριθμών, με τη συνάρτηση `range(start,stop,step)`, δημιουργείτε η ακολουθία start, start+step,…,stop-1
  - Παράδειγμα: `myRange=range(1, 10, 2)` δημιουργεί την ακολουθία 1, 3, 5, 7, 9

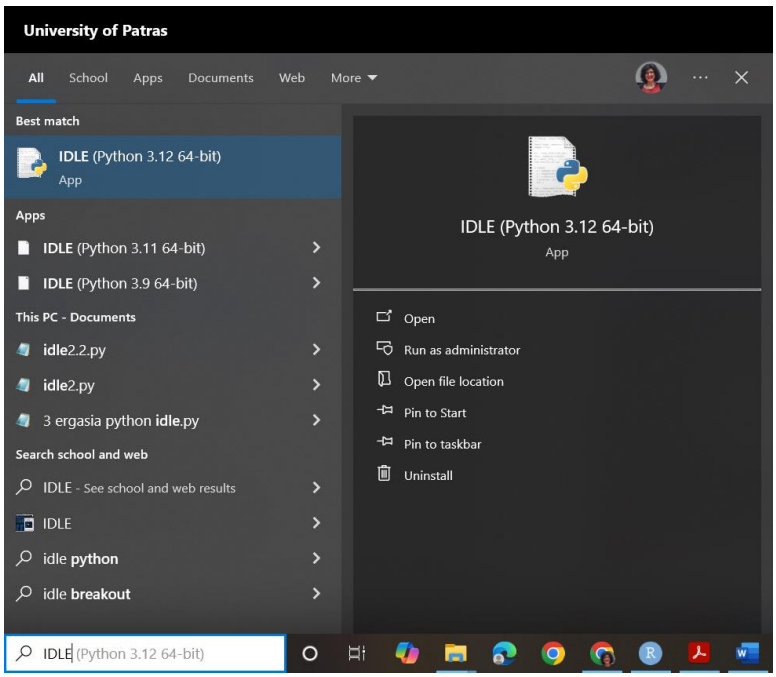
Μη-διατεταγμένες συλλογές

- **set (Σύνολα)**: Μια **μεταβαλλόμενη**, **μη διατεταγμένη** συλλογή **μοναδικών** στοιχείων (δυνητικά) διαφορετικού τύπου μέσα σε **{}** (curly brackets), στα οποία μπορούμε να εφαρμόσουμε εύκολα πράξεις συνόλων.
  - Παράδειγμα: `unique_numbers = {1, 2, 3}`
- **dict (Λεξικά):** Απεικονίσεις σε ζεύγη κλειδιών-τιμών (key-value) (συνήθως διαφορετικού τύπου), μέσα σε **{}** (curly brackets). Τα κλειδιά δεν μεταβάλλονται, οι τιμές μεταβάλλονται.
  - Παράδειγμα: `person = {"name": "John", "age": 30}`

Ένα πρώτο πρόγραμμα

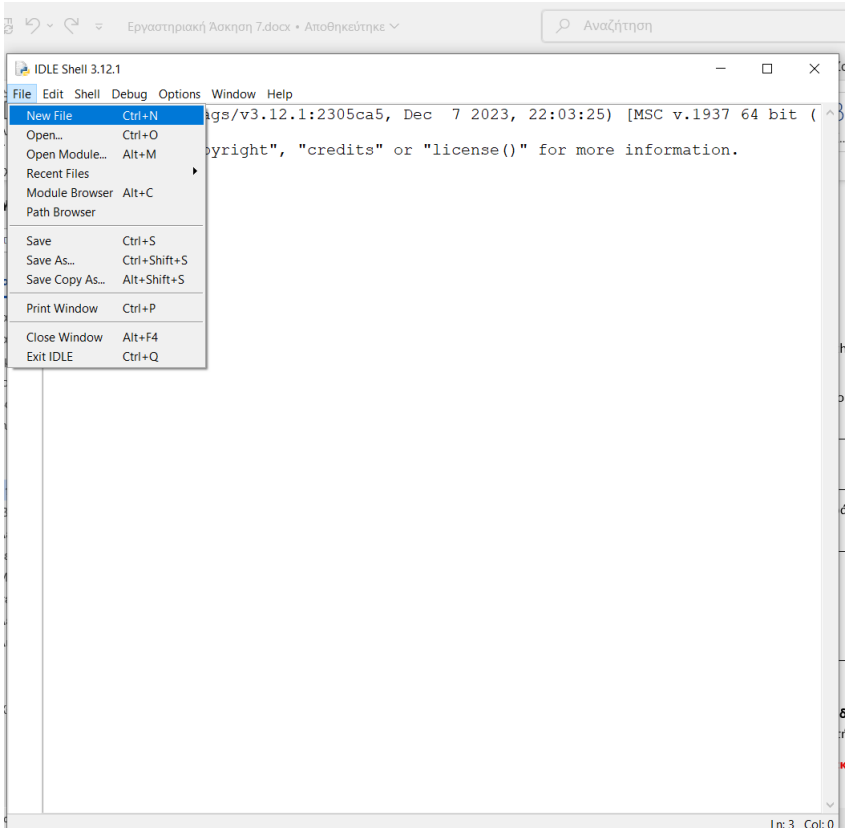
Η γλώσσα Python με την εγκατάστασή της έχει ενσωματωμένο ένα περιβάλλον ανάπτυξης που ονομάζεται IDLE.

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

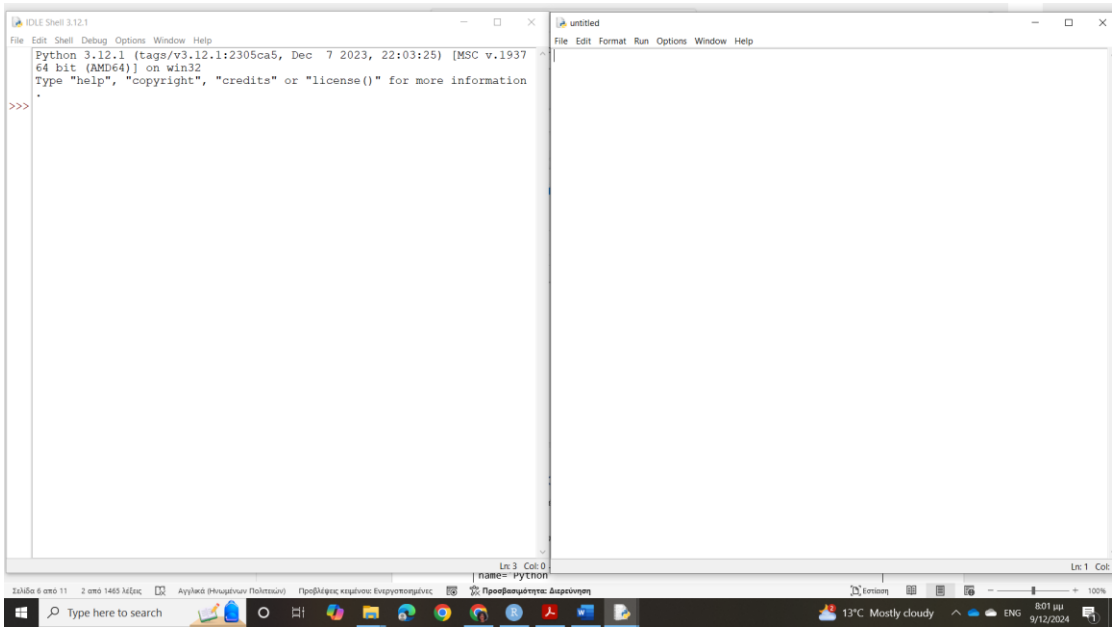


Εμφανίζεται τότε το παράθυρο του IDLE Shell όπου μπορούμε να δίνουμε απευθείας εντολές στη γλώσσα python είτε να δημιουργούμε προγράμματα.

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



Τότε δημιουργείται ένα νέο αρχείο με όνομα *untitled*, στο οποίο θα γράψουμε τις πρώτες μας εντολές.




```
print("Hello world!")

myName="Victoria Daskalou"
```

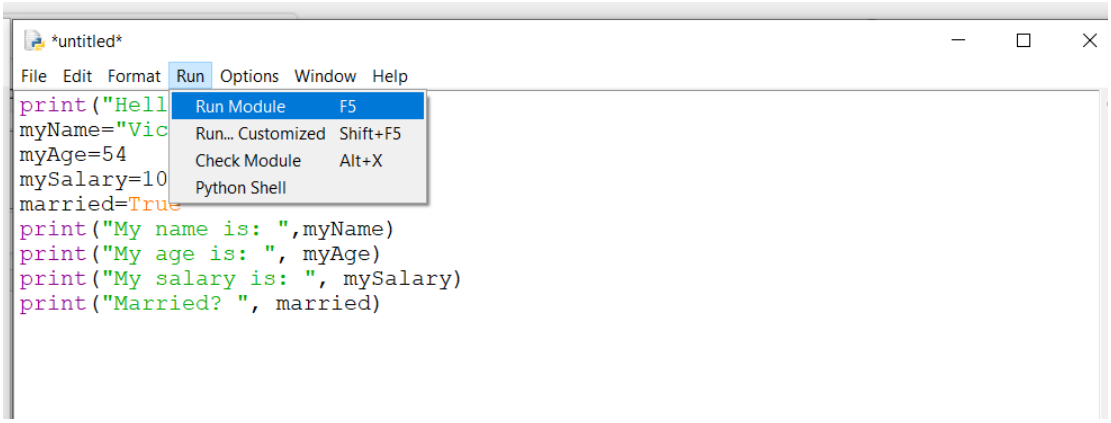
```
myAge=54
mySalary=1000.5
married=True
print("My name is:",myName)
print("My age is:", myAge)
print("My salary is:", mySalary)
print("Married?", married)
```

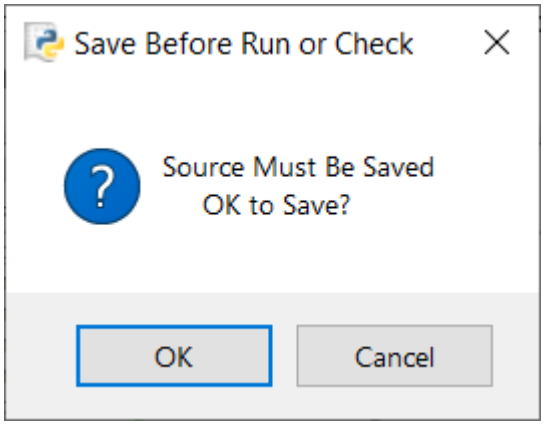
Για την εκτύπωση στην οθόνη χρησιμοποιούμε την εντολή `print()`. Η πλήρης σύνταξη της `print()` είναι:

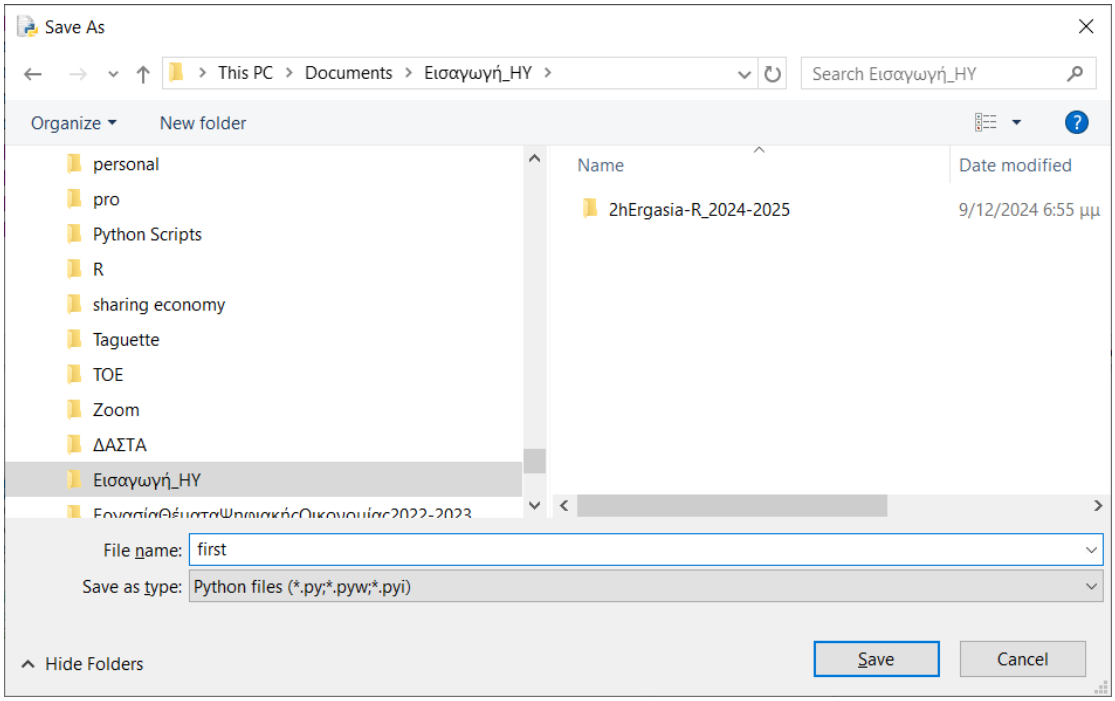
`print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)`

Δηλαδή, παίρνει ως κύριο όρισμα τα αντικείμενα (`objects`), χωριζόμενα με κόμμα, που τα εκτυπώνει σε αρχείο (`file`) διαχωριζόμενα από τη συμβολοσειρά `sep`, τελειώνοντας με τον χαρακτήρα `end`. Το προκαθορισμένο αρχείο είναι το `sys.stdout`, standard output, δηλαδή η οθόνη.

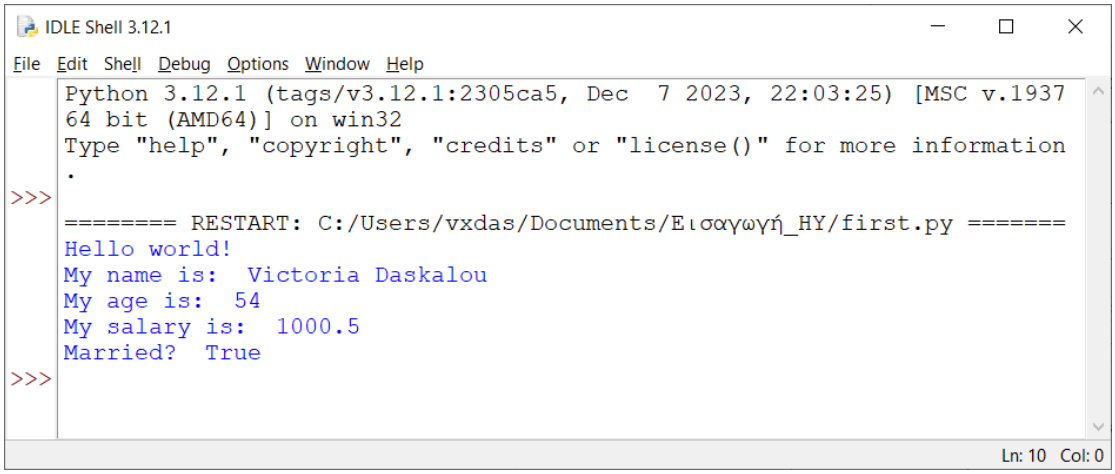
Στη συνέχεια εκτελούμε το πρόγραμμα επιλέγοντας από το μενού *Run>Run Module ή F5*. Τότε θα μας ζητήσει να αποθηκεύσουμε πρώτα το πρόγραμμα και θα εμφανίζει το παράθυρο *Save As* όπου επιλέγουμε το όνομα του προγράμματος, εδώ first.py







Κάθε εκτέλεση προγράμματος εμφανίζονται τα αποτελέσματα στο παράθυρο του Shell. Φυσικά θα πρέπει να υπάρχει η αντίστοιχη εντολή `print()` για να εμφανιστεί το περιεχόμενο μίας μεταβλητής στην οθόνη.



Ερώτηση 2: Ποιό το αποτέλεσμα εκτέλεσης των ακόλουθων εντολών στη γλώσσα Python;

```
>>> myStr="Hello world!"
>>> myStr[1]='V'
```

Ερωτήσεις 3-8: Αν στη γλώσσα python εκχωρήσουμε στη μεταβλητή myStr το ακόλουθο λεκτικό, ποιο είναι το αποτέλεσμα της έκφρασης …:

```
myStr="Hello world!"
```

Συμβολοσειρές

Οι συμβολοσειρές (str, string) στην Python είναι διατεταγμένες, στατικές (immutable), ακολουθίες χαρακτήρων.

Μία σταθερά τύπου str είναι μία ακολουθία χαρακτήρων μέσα σε quotes (single ' or double quotes ").

```
name='Python'

name="Python"
```

Αν θέλουμε να ορίσουμε συμβολοσειρά που να εκτείνεται σε πολλαπλές γραμμές βάζουμε τρία μονά quotes, ως εξής:

```
multi_lines='''This is a

multi

line

string'''
```

Δεικτοδότηση (indexing)

Μπορούμε να χρησιμοποιήσουμε ένα **δείκτη (index)** μέσα στον τελεστή **[]** (square brackets) και να πάρουμε όποιο στοιχείο-χαρακτήρα θέλουμε από μία συμβολοσειρά.

Όλοι οι δείκτες στη γλώσσα Python ξεκινούν από το 0 (zero-based language).

Κάθε συμβολοσειρά έχει **δύο δείκτες**:

1. ένα **δείκτη θετικό που ξεκινά από το *0* στην αρχή της συμβολοσειρά**ς και φτάνει έως το ***μήκος της συμβολοσειράς μείον 1*** στο τέλος της.
2. ένα **δείκτη αρνητικό που ξεκινά από το *-1* στο τέλος της συμβολοσειράς** και φτάνει έως το ***-μήκος συμβολοσειράς*** στην αρχή της.

Για παράδειγμα αν ορίσουμε τη μεταβλητή `s='The answer is 52'` (όπως παρουσιάζονται στο shell του IDLE):

```
>>> s='The answer is 52'

>>> s

'The answer is 52'

>>> len(s)
```

```
16
```

για το δείκτη ισχύουν τα ακόλουθα:

s =	T	h	e		a	n	s	w	e	r		i	s		4	2
Δείκτης/θέση	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Αρνητικοί δείκτες	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

Δηλαδή για να πάρουμε ένα συγκεκριμένο χαρακτήρα της συμβολοσειράς χρησιμοποιούμε το όνομα της μεταβλητής με τον τελεστή [] και τη θέση του χαρακτήρα ως δείκτη μέσα στα []. Για παράδειγμα το χαρακτήρα `'a'` στην αρχή της λέξης `answer` ισχύει:

```
>>> s
'The answer is 42'
>>> s[4]
'a'
>>> s[-12]
'a'
```

Οι συμβολοσειρές είναι στατικές. Δηλαδή δεν μπορούμε να αλλάξουμε κανένα στοιχείο της. Δείτε το αποτέλεσμα της εντολής ανάθεσης `s[7]='W'`, όπου με κόκκινο χρώμα εμφανίζεται το μήνυμα λάθους της εντολής.

```
>>> s
'The answer is 42'
>>> s[7]
'w'
>>> s[7]='W'
Traceback (most recent call last):
  File "<pyshell#47>", line 1, in <module>
    s[7]='W'
TypeError: 'str' object does not support item assignment
```

Τεμαχισμός (slicing)

Τεμαχισμός σημαίνει επιστροφή τμήματος της τιμής μιας συμβολοσειράς (υπο-συμβολοσειρά) προσδιορίζοντας τις επιθυμητές θέσεις με δείκτες και τον τελεστή []:

[<δείκτης_αρχής> : <δείκτης_τέλους> : <διασκελισμός>]

Που σημαίνει: ***φέρε τους χαρακτήρες από τη θέση δείκτη_αρχής έως τη θέση δείκτη_τέλους-1 και για τις θέσεις δείκτη_αρχής+διασκελισμός (+διασκελισμός)…***

Οι δείκτες και ο διασκελισμός σε εκφράσεις τεμαχισμού συμβολοσειράς είναι ακέραιοι, θετικοί ή αρνητικοί.

Μπορεί να παραλείπεται οποιαδήποτε τιμή από τις <δείκτης_αρχής>, <δείκτης_τέλους>, <διασκελισμός>, όπως θα εξηγηθεί στη συνέχεια.

Αν παραλείπεται η θέση <δείκτης_αρχής> σημαίνει η αρχή της συμβολοσειράς.

Αν παραλείπεται η <δείκτης_τέλους> σημαίνει το τέλος της συμβολοσειράς.

Αν παραλείπεται το όρισμα <διασκελισμός> εννοείται η τιμή 1.

Με βάση τη συμβολοσειρά:

s =	T	h	e		a	n	s	w	e	r		i	s		4	2
Δείκτης/θέση	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Αρνητικοί δείκτες	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

Ας εξηγήσουμε τα ακόλουθα τεμάχια:

s[start : stop]	Από τη θέση s[start] έως και τη θέση s[stop-1]	`s[4:9]`	`'answe'`
s[start :]	Από τη θέση s[start] έως το τέλος της συμβολοσειράς	`s[9:]`	`'r is 42'`
s[: stop]	Από την αρχή της συμβολοσειράς έως και τη θέση s[stop-1]	`s[ :5]`	`'The a'`
s[:]	Από την αρχή έως το τέλος, όλη η συμβολοσειρά.	`s[:]`	`'The answer is 42'`
s[-x :]	Από τη θέση s[-x] έως το τέλος, δηλαδή οι x (πλήθος) χαρακτήρες από το τέλος.	`s[-5:]`	`'is 42'`
s[: -x]	Από την αρχή έως και τη θέση s[-x-1], δηλαδή εκτός τους τελευταίους x (πλήθος) χαρακτήρες.	`s[:-5]`	`'The answer '`

s[start : stop : stride]	Από τη θέση s[start] έως τη θέση s[stop-1] αυξάνοντας τη θέση κατά stride	`s[4:9:3]` (s[4] έως και s[8] για τις θέσεις 4, 7	`'aw'`
s[start : stop : -stride]	Από τη θέση s[start] έως και τη θέση s[stop-1] μειώνοντας τη θέση κατά stride (πρέπει start>stop)	`s[14:4:-3]` (s[14] έως και s[3] στις θέσεις 14, 11, 8, 5)	`'4ien'`

Μέθοδοι χειρισμού συμβολοσειρών

Εκτός από τη συνάρτηση len(mystr), που επιστρέφει το μήκος μίας συμβολοσειράς mystr, υπάρχουν πολλές συναρτήσεις-μέθοδοι για τις συμβολοσειρές. Μπορείτε να τις βρείτε στην τεκμηρίωση εδώ: https://docs.python.org/3/library/stdtypes.html#string-methods

Ερώτηση 9: Ο ακόλουθος κώδικας σε γλώσσα Python δημιουργεί:

```
>>> [1, 2, 'stock', 4]
```

Λίστες

Οι λίστες είναι μία δυναμική δομή δεδομένων για **διατεταγμένη** συλλογή στοιχείων **διαφορετικού τύπου,** χωριζόμενα με κόμμα, μέσα σε **[]** (square brackets). Για παράδειγμα αποδεκτές λίστες είναι:

```
keniLista=[]
myList=['banana', 'apple', 1, 2,3.5, False]
listofLists=[[1,2,3],['Maria','Nikos','Panos']]
anOtherList=[x for x in range(0,11)]
```

Ερώτηση 10: Έστω ο ακόλουθος κώδικας σε γλώσσα Python. Ταιριάξετε τα δυνατά αποτελέσματα:

```
>>> colors = ['red', 'yellow', 'blue', 'black']
```

Ερώτηση 11: Έστω ο ακόλουθος κώδικας σε γλώσσα Python. Ταιριάξετε τα δυνατά αποτελέσματα:

```
a = [[4, 5], [6, 7], [8, 9]]
```

Δεικτοδότηση και Τεμαχισμός

Στις λίστες ισχύουν ακριβώς ότι έχουμε μάθει για τη δεικτοδότηση και τον τεμαχισμό των συμβολοσειρών, προσδιορίζοντας τις επιθυμητές θέσεις με δείκτες και τον τελεστή []:

[<δείκτης_αρχής> : <δείκτης_τέλους> : <διασκελισμός>]

```
>>> myList
['banana', 'apple', 1, 2, 3.5, False]
>>> myList[1]
'apple'
>>> myList[-1]
False
```

```
>>> myList[::-1]
[False, 3.5, 2, 1, 'apple', 'banana']
```

Όταν ένα στοιχείο της λίστας είναι μία άλλη λίστα μπορούμε να αναφερθούμε και στα στοιχεία της ή στα τμήματά της με τη λογική:

onoma_listas[δείκτης_έξω_λίστας][δείκτης_έσω_λίστας]

όπως παρουσιάζεται στο παράδειγμα:

```
>>> listofLists
[[1, 2, 3], ['Maria', 'Nikos', 'Panos']]
>>> listofLists[1]
['Maria', 'Nikos', 'Panos']
>>> listofLists[-1]
['Maria', 'Nikos', 'Panos']
>>> listofLists[1][1]
'Nikos'
```

Διαχείριση λιστών

Αλλαγή στοιχείου

Επειδή οι λίστες ως τύπος δεδομένων είναι μεταβλητός (mutable) μπορούμε να αλλάξουμε τα στοιχεία τους με ανάθεση. Για παράδειγμα:

```
>>> myList
['banana', 'apple', 1, 2, 3.5, False]
>>> myList[0]='mango'
>>> myList
['mango', 'apple', 1, 2, 3.5, False]
```

Πράξεις σε λίστες

Μπορούμε να βρούμε το μέγεθος μίας λίστας με τη συνάρτηση len(), προσθέσουμε ή να αφαιρέσουμε στοιχεία σε μία λίστα και να κάνουμε και άλλες πράξεις. Στις λίστες ο τελεστής + συνενώνει δύο λίστες.

```
>>> myList
['mango', 'apple', 1, 2, 3.5, False]
>>> anOtherList
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> myList+anOtherList
['mango', 'apple', 1, 2, 3.5, False, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

Περισσότερες πράξεις για διαχείριση των λιστών μπορείτε να βρείτε εδώ:
https://docs.python.org/3/library/stdtypes.html#typesseq-mutable