ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

Εισαγωγή στους Η/Υ και τις Εφαρμογές

Ενότητα 5: Επεξεργασία δεδομένων με τη γλώσσα προγραμματισμού python

Υπο-ενότητα Ν5.2: Τύποι δεδομένων

Μανώλης Τζαγκαράκης, Βικτωρία Δασκάλου

Σχολή Οργάνωσης και Διοίκησης Επιχειρήσεων

Τμήμα Οικονομικών Επιστημών

Σκοποί ενότητας

- Να πραγματοποιηθεί μία εισαγωγή στις δυνατότητες της γλώσσας python
- Να παρουσιαστούν οι βασικές δυνατότητες των γλωσσών προγραμματισμού για την επεξεργασία δεδομένων

Περιεχόμενα ενότητας

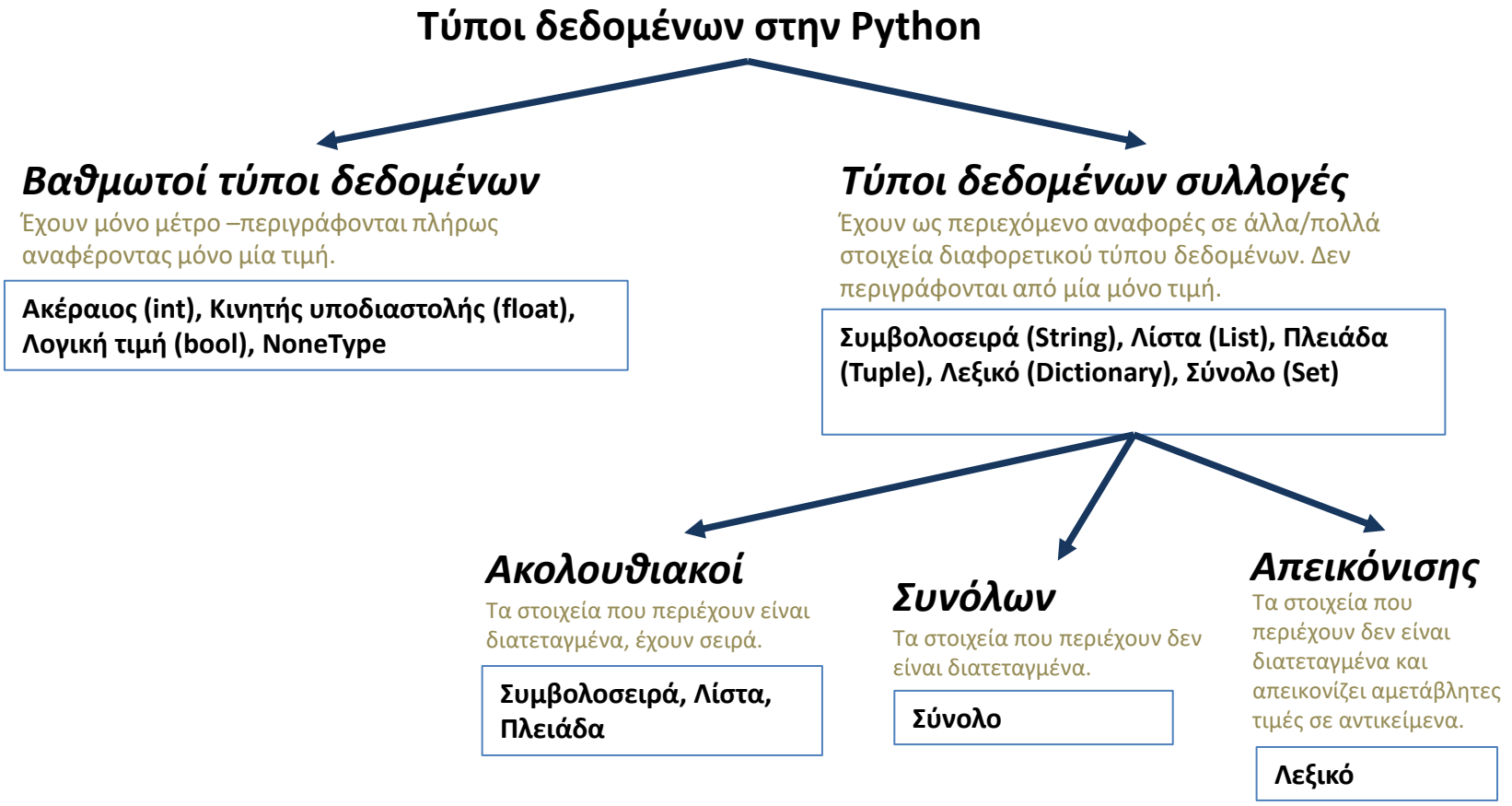
1. Χαρακτηριστικά της γλώσσας python
2. Προγραμματισμός με python
   - Υπολογισμοί και μεταβλητές
   - **Τύποι δεδομένων**
   - Βιβλιοθήκες χειρισμού δεδομένων

Τύποι δεδομένων στη γλώσσα Python

- Υποστηριζόμενοι βασικοί τύποι δεδομένων της Python
  - **Αριθμοί (Numbers) – ακέραιοι και κινητής υποδιαστολής**
  - **Συμβολοσειρές (Strings)**
  - **Λίστες (Lists)**
  - **Πλειάδες (Tuples)**
  - **Σύνολα (Sets)**
  - **Λεξικά (Dictionaries)**
  - **Λογικοί τύποι (Boolean)**
  - **Bytes και Byte arrays**

Τύποι δεδομένων στη γλώσσα Python

- Κατηγορίες βασικών τύπων δεδομένων στην Python βάσει του είδους τιμών που λαμβάνουν



Τύποι δεδομένων στη γλώσσα Python

- Με την εντολή type() της Python και όρισμα το όνομα μιας μεταβλητής, μπορεί να εμφανιστεί ο τύπος δεδομένων μιας μεταβλητής.

```
>>> s='This is a string' #Μεταβλητή s αποκτά τύπο δεδομένων
                         συμβολοσειρά καθότι ανατίθεται μία κυριολεκτική
                         σταθερά
>>> type(s)
<class 'str'>
>>> a = 42
>>> type(a)
<class 'int'>
>>>b=3.08
>>>type(b)
<class 'float'>
```

Χρήση της type() για την εμφάνιση του τύπου δεδομένων μιας ματαβλητής

Μεταβλητή s είναι τύπου συμβολοσειράς (string)

Μεταβλητή a είναι τύπου ακέραιος (int)

Μεταβλητή b είναι τύπου κινητής υποδιαστολής/πραγματικός (float)

Συμπεριφορά μεταβαλλόμενων τύπων δεδομένων

- Οι τύποι δεδομένων στην Python μπορούν επίσης να χωριστούν σε δύο κατηγορίες αναφορικά με το πως συμπεριφέρονται τα αντικείμενα κατά τις πράξεις τροποποίησης/μεταβολής τους:
  - **Αμετάβλητοί τύποι δεδομένων (immutable)**
    - Τα αντικείμενα δεν μπορούν να αλλάξουν. Ανάθεση νέας τιμής ή τροποποίηση **δημιουργεί πάντα νέο αντικείμενο**.
  - **Μεταβαλλόμενοι τύποι δεδομένων (mutable)**
    - Τα αντικείμενα μπορούν να αλλάξουν δίχως να δημιουργηθούν νέα αντικείμενα (με ορισμένες εξαιρέσεις πράξεων)

Αμετάβλητοι τύποι δεδομένων	Μεταβαλλόμενοι τύποι δεδομένων
Ακέραιοι (int), Αριθμοί κινητής υποδιαστολής (float), Συμβολοσειρές (string), Πλειάδες (tuple), Λογικές τιμές (bool).	**Λίστες (list), Σύνολα (set), Λεξικά (dict)**

Τύποι δεδομένων στη γλώσσα Python

- Αριθμοί
  - Ακέραιοι (int), αριθμοί κινητής υποδιαστολής (floating point numbers-float) και μιγαδικοί (complex)
    - Ο τύπος δεδομένων μιας μεταβλητής συνάγεται αυτόματα από τον διερμηνευτή βάσει της τιμής που ανατίθεται (π.χ. αν ο αριθμός έχει δεκαδικό μέρος/περιέχει υποδιαστολή ή όχι)
  - Στις αριθμητικές εκφράσεις, ο τύπος δεδομένων του αποτελέσματος καθορίζεται από τον τύπο δεδομένων των τιμών των μεταβλητών που μετέχουν στην έκφραση και το αποτέλεσμα της έκφρασης
    - Η python εμμέσως αλλάζει τον τύπο δεδομένων των ορισμάτων προκειμένου να γίνει εφικτή η αποτίμηση της παράστασης

Τύποι δεδομένων στη γλώσσα Python

- Αριθμητικοί τελεστές

Τελεστής	Σημασία
+	Πρόσθεση
-	Αφαίρεση
*	Πολλαπλασιασμός
/	Διαίρεση
%	Modulo (υπόλοιπο Ευκλείδειας διαίρεσης)
**	Ύψωση σε δύναμη
//	Πηλίκο Ευκλείδειας διαίρεσης

Τύποι δεδομένων στη γλώσσα Python

- Τελεστές σύγκρισης
  - **Όχι** μόνο για αριθμητικές τιμές.

Τελεστής	Σημασία
==	Έλεγχος ισότητας τιμής
>	Μεγαλύτερο
<	Μικρότερο
>=	Μεγαλύτερη ή ίσο
<=	Μικρότερο ή ίσο
!=	Διάφορο, όχι όσο

Τύποι δεδομένων στη γλώσσα Python

- Συμβολοσειρές (str)
  - Ακολουθίες οποιονδήποτε χαρακτήρων, οσοδήποτε μεγάλες.
    - Κυριολεκτήματα (σταθερές (string literals) μεταξύ ‘’ ή “ “
    - Χρήση κωδικοσελίδας Unicode.
    - Μεταβλητές τύπου δεδομένων συμβολοσειράς **είναι αμετάβλητες (immutable)**
      - Η τιμή που διατηρούν ΔΕΝ μπορεί να τροποποιηθεί.

```
>>> s=‘This is a string’ #Μεταβλητή s αποκτά τύπο δεδομένων
                         συμβολοσειρά καθότι ανατίθεται μία κυριολεκτική
                         σταθερά

#Τριπλά εισαγωγικά (‘’’ ή “””) για τιμές που καταλαμβάνουν πολλές γραμμές
>>> s=‘’’This is a
        multiline
        string’’’
```

Τύποι δεδομένων στη γλώσσα Python

- Συμβολοσειρές (str)
  - Ως ακολουθιακός τύπος δεδομένων, υποστηρίζει πράξεις δεικτοδότησης (indexing) και τεμαχισμού (slicing) με τον τελεστή [ ]
    - Δεικτοδότηση: Προσπέλαση συγκεκριμένου χαρακτήρα της συμβολοσειράς
    - Τεμαχισμός: Επιστροφή τμήματος της τιμής μιας συμβολοσειράς (υπο-συμβολοσειρά) προσδιορίζοντας τις επιθυμητές θέσεις με δείκτες και τον τελεστή []: **[<δείκτης αρχής> : <δείκτης τέλους> : <διασκελισμός>]**
      - Δείκτες και διασκελισμός σε εκφράσεις τεμαχισμού συμβολοσειράς είναι ακέραιοι, θετικοί ή αρνητικοί. Αν παραλείπεται το όρισμα <διασκελισμός> εννοείται η τιμή 1
    - Στη Python η αρίθμηση **ξεκινά (πάντα) από το 0**

Τύποι δεδομένων στη γλώσσα Python

- Παράδειγμα δεικτοδότησης και τεμαχισμού

```
>>> s='This is a string'
>>> s[3] # Δεικτοδότηση. Προσπέλαση στοιχείου στη θέση 3
 's'
>>> s[2:5] #Τεμαχισμός συμβολοσειράς s: λήψη του τμήματος της τιμής της
                συμβολοσειράς της s από τη θέση 2 έως και τη θέση 4
 'is '
```

Τύποι δεδομένων στη γλώσσα Python

- Τεμαχισμός συμβολοσειρών (slicing) με τον τελεστή []

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

Έκφραση τεμαχισμού	Σημασία	Παράδειγμα
s[start : stop]	Τμήμα της συμβολοσειράς από τη θέση (δείκτη) start έως και stop-1	s[4:9] -> 'answe'
s[start :]	Τμήμα της συμβολοσειράς από τη θέση start μέχρι το τέλος	s[9:] -> 'r is 42'
s[: stop]	Τμήμα της συμβολοσειράς από την αρχή μέχρι και τη θέση stop-1	s[:5]->'The a'
s[:]	Ολόκληρη η συμβολοσειρά	s[:]->'The answer is 42'
s[x]	Ο χαρακτήρας στη θέση x	s[11]->'i'

Τύποι δεδομένων στη γλώσσα Python

- Τεμαχισμός συμβολοσειρών (slicing)

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

Έκφραση τεμαχισμού	Σημασία	Παράδειγμα
s[-x]	Ο x-οστός κατά σειρά χαρακτήρας μετρώντας απ' το τέλος της συμβολοσειράς	s[-2] -> '4'
s[-x :]	Οι τελευταίοι x (σε πλήθος) χαρακτήρες συμβολοσειράς	s[-5:] ->'is 42'
s[: -x]	Ολόκληρη η συμβολοσειρά εκτός των τελευταίων x (σε πλήθος) χαρακτήρων της συμβολοσειράς	s[:-6]->'The answer'

Τύποι δεδομένων στη γλώσσα Python

- Τεμαχισμός συμβολοσειρών (slicing)

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

Έκφραση τεμαχισμού	Σημασία	Παράδειγμα
s[start : stop : stride]	Τμήμα της συμβολοσειράς από τη θέση start έως και stop-1 αλλά λαμβάνοντας κάθε <stride> χαρακτήρα	s[4:9:3] -> 'aw'
s[start : stop : -stride]	Τμήμα της συμβολοσειράς από τη θέση start μέχρι και τη θέση stop+1 ξεκινώντας από το τέλος της συμβολοσειράς λαμβάνοντας κάθε <stride> χαρακτήρα (απαιτεί start > stop)	s[14:4:-3] -> '4ien'

Τύποι δεδομένων στη γλώσσα Python

- Τεμαχισμός γενικές αρχές ερμηνείας start, stop και stride:
  - **Θετικοί δείκτες** και βήμα σημαίνουν έναρξη από την αρχή της συμβολοσειράς (από τα αριστερά, θέση 0) και αύξηση δείκτη
  - **Αρνητικοί δείκτες** ή/και βήμα σημαίνουν έναρξη από το τέλος της συμβολοσειράς (από τα δεξιά, θέση -1) και μείωση δείκτη

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

Θετικοί δείκτες → ← Αρνητικοί δείκτες

Άσκηση

- Δίνεται μία μεταβλητή τύπου συμβολοσειράς myStr, η οποία αρχικοποιείται με την παρακάτω τιμή:

  myStr=”Hello, World!”

  Ποιο τμήμα της συμβολοσειράς θα επιστραφεί αν εκτελεστούν οι εξής εκφράσεις τεμαχισμού;

  i. myStr[8]

  ii. myStr[1:]

  iii. myStr[:-5]

  iv. myStr[-8:10]

  v. myStr[-10:-3]

  vi. myStr[12:5:-2]

  vii. myStr[-8:3]

  viii. myStr[::-4]

Άσκηση

- Δίνεται μία μεταβλητή τύπου συμβολοσειράς myStr, η οποία αρχικοποιείται με την παρακάτω τιμή:

  myStr=”Hello, World!”

  Ποιο τμήμα της συμβολοσειράς θα επιστραφεί αν εκτελεστούν οι εξής εκφράσεις τεμαχισμού;

  i. myStr[8] ‘o’ # χαρακτήρας στη θέση 8 (ένατος χαρακτήρας κατά σειρά)

  ii. myStr[1:] 'ello, World!‘ # από τη θέση 1 έως το τέλος

  iii. myStr[:-5] 'Hello, W‘# από την αρχή (θέση 0) εκτός των τελευταίων 5 χαρακτήρων

  iv. myStr[-8:10] ', Wor‘# έναρξη από τον όγδοο χαρακτήρα από το τέλος εως και τη θέση 9.

  v. myStr[-10:-3] 'lo, Wor‘# από τη δέκατη θέση απ’το τέλος έως και τη θέση -4

  vi. myStr[12:5:-2] '!lo ' # από τη θέση 12 έως και τη θέση 6, κάθε δεύτερο χαρακτήρα

  vii. myStr[-8:3] ‘’ (κενή συμβολοσειρά) # από τη θέση -8 έως και τη θέση 2 με βήμα 1. Αδύνατο.

  viii. myStr[::-4] '!ooH' # έναρξη από το τέλος και εμφάνιση κάθε 4 χαρακτήρα

Άσκηση

- Δίνεται μία μεταβλητή τύπου συμβολοσειράς myStr, η οποία αρχικοποιείται με την παρακάτω τιμή:

  myStr=”Hello, World!”

  Τί θα συμβεί αν εκτελεστούν κάθε μία από τις παρακάτω εντολές;

  i. myStr = “Hi there”

  ii. myStr[8] = ‘Z’

Άσκηση

- Δίνεται μία μεταβλητή τύπου συμβολοσειράς myStr, η οποία αρχικοποιείται με την παρακάτω τιμή:

  myStr=”Hello, World!”

  Τί θα συμβεί αν εκτελεστούν κάθε μία από τις παρακάτω εντολές;

  i. myStr = “Hi there” # Μεταβλητή myStr θα αλλάξει τιμή. Νέα τιμή “Hi there”

  ii. myStr[8] = ‘Z’ # Σφάλμα. Οι μεταβλητές τύπου συμβολοσειράς είναι αμετάβλητες (immutable), δηλαδή το περιεχόμενό τους δεν μπορεί να τροποποιηθεί

```
myStr[8]='Z'
TypeError: 'str' object does not support item assignment
```

Τύποι δεδομένων στη γλώσσα Python

- Συμβολοσειρές (str)
  - Ως αντικείμενα (όπως κάθε τιμή στην Python) οι τιμές τύπου δεδομένων συμβολοσειράς έχουν σύνολο μεθόδων που μπορούν να κληθούν για την επεξεργασία της τιμής της μεταβλητής.
    - Π.χ. len() για μήκος συμβολοσειράς, .upper() για τη μετατροπή όλων των χαρακτήρων της συμβολοσειράς σε κεφαλαία κλπ.
    - Οι μέθοδοι αυτές ωστόσο δεν μπορούν να αλλάξουν την τιμή του υπάρχοντος αντικειμένου αφού συμβολοσειρές στην Python είναι αμετάβλητοι τύποι δεδομένων.

Τύποι δεδομένων στη γλώσσα Python

- Εαν

  `yourString = “this is a simple test”`

  τότε
  - `yourString.`**`upper`**`()` - η συμβολοσειρά σε κεφαλαία
  - `yourString.`**`lower`**`()` - η συμβολοσειρά σε πεζά
  - `yourString.`**`capitalize`**`()` - η συμβολοσειρά με το πρώτο γράμμα κεφαλαίο
  - `yourString.`**`title`**`()` - η συμβολοσειρά με το πρώτο γράμμα κάθε λέξης κεφαλαίο
  - `yourString.`**`replace`**`(x,y)` - η συμβολοσειρά με αντικατάσταση του χαρακτήρα **x** με χαρακτήρα **y**
  - **`len`**`(yourString)` - το μήκος της συμβολοσειράς

Περισσότερες συναρτήσεις σε συμβολοσειρές:

- https://docs.python.org/3.4/library/stdtypes.html#string-methods

Άσκηση

- Τί θα εμφανιστεί στην οθόνη εάν εκτελεστεί παρακάτω έκφραση;

  “a simple test”[3]

  i. Η εκτέλεση της έκφρασης θα δημιουργήσει σφάλμα. Δεν μπορεί να γίνει χρήση του τελεστή [ ] για σταθερές τύπου συμβολοσειράς (κυριολεκτήματα)

  ii. ‘i’

  iii. ‘s’

Άσκηση

- Τί θα εμφανίσει στην οθόνη εάν εκτελεστεί παρακάτω έκφραση;

  “a simple test”[3]

  i. Η εκτέλεση της έκφρασης θα δημιουργήσει σφάλμα. Δεν μπορεί να γίνει χρήση του τελεστή [ ] για σταθερές τύπου συμβολοσειράς (κυριολεκτήματα)

  ii. ‘i’ # Σωστή απάντηση. Στην Python και οι σταθερές τιμές (κυριολεκτήματα - literals) είναι αντικείμενα όπου μπορούν να εφαρμοστούν όλοι οι σχετικοί τελεστές και μέθοδοι.

  iii. ‘s’

Τύποι δεδομένων στη γλώσσα Python

- Λίστες (List)
  - Δυναμική δομή δεδομένων για διατεταγμένη συλλογή στοιχείων διαφορετικού τύπου δεδομένων
    - **Δυναμική**: Μπορεί να τροποποιηθεί το περιεχόμενο της λίστας (πρόσθεση/αφαίρεση/ενημέρωση στοιχείων)
    - **Διατεταγμένη**: Η σειρά των στοιχείων εντός της λίστας παίζει ρόλο.
    - Τα στοιχεία της λίστας που διατηρεί μπορεί να είναι διαφορετικού τύπου δεδομένων
      - int, float, str ακόμη και lists
    - Δημιουργία λίστας με τη χρήση αγκύλων [] και διαχωρισμός στοιχείων/τιμών με κόμμα (,)

```
>>> myList=['banana', 'apple', 1, 2,3] #δημιουργία μεταβλητής τύπου λίστας
                                        που περιέχει 5 στοιχεία: 2 τύπου
                                        συμβολοσειράς και 3 τύπου ακέραιος.
```

Τύποι δεδομένων στη γλώσσα Python

- Λίστες (List)
  - Προσπέλαση μεμονωμένων στοιχείων λίστας ή τμημάτων αυτής με χρήση του τελεστή [] : [start:end:step]
    - Δεικτοδότηση στοιχείων **ξεκινά από το 0.**
    - Τελεστής [] επιστρέφει τα στοιχεία που προσδιορίζονται, **πάντα ως λίστα** .
    - Σύνταξη και ερμηνεία ακριβώς ίδια με αυτήν της τμηματοποίησης συμβολοσειρών
      - start, end, step μπορούν να είναι θετικοί και αρνητικοί

```
>>> myList=['banana', 'apple', 1, 2,3]
>>>myList[3] # Επιστροφή στοιχείου στη θέση 3. Επιστρέφεται ως ακέραιος.
2
>>>myList[0:2] #Επιστροφή στοιχείων λίστας από το στοιχείο στη θέση 0
                έως και το στοιχείο στη θέση 1 με τη μορφή λίστας.
['banana', 'apple']
>>>myList[-3:] #Επιστροφή τελευταίων 3 στοιχείων της λίστας
[1,2,3]
```

Τύποι δεδομένων στη γλώσσα Python

- Λίστες (List)
  - Προσθήκη/αφαίρεση/τροποποίηση στοιχείων λίστας
    - Προσθήκη στοιχείων
      - Τελεστής +: συνένωση δύο λιστών με τη νέα λίστα να προστίθεται στο τέλος του ορίσματος στα αριστερά του τελεστή
      - Μέθοδος .append() προθέτει νέο στοιχείο στο τέλος της λίστας
      - Μέθοδος .extend() προσθέτει νέα λίστα στο τέλος μιας άλλης λίστας
      - Μέθοδος .insert() εισάγει νέο στοιχείο σε συγκεκριμένη θέση της λίστας.
    - Αφαίρεση στοιχείων
      - Μέθοδος .remove() αφαιρεί συγκεκριμένο στοιχείο (την πρώτη του εμφάνιση εντός της λίστας), παίρνοντας ως όρισμα την τιμή του στοιχείου προς διαγραφή. Αν δεν υπάρχει το προς διαγραφή στοιχείο στη λίστα, προκαλείται σφάλμα.
      - Μέθοδος .pop() αφαιρεί και επιστρέφει το τελευταίο στοιχείο της λίστας αν δεν δοθεί όρισμα ή το στοιχείο στη θέση που δίνεται ως όρισμα
      - Τελεστής del δίνοντας τη θέση του στοιχείου προς διαγραφή

Τύποι δεδομένων στη γλώσσα Python

- Λίστες (List)
  - Τροποποίηση υπάρχουσας τιμής
    - Με τελεστή [] και αναφορά θέσης που επιθυμείται τροποποίηση τιμής.
  - Μεμονωμένα στοιχεία της λίστας μπορούν να ανατεθούν σε ξεχωριστές μεταβλητές με τον τελεστή =

```
>>> myList=[42, 43, -8, “Hi there”] # Δημιουργία λίστας με 4 στοιχεία.
>>> a,b,c,d = myList # Το στοιχείο 0 της λίστας myList θα ανατεθεί στη
                     μεταβλητή a, το στοιχείο 1 στη μεταβλητή b κοκ.
>>> c
-8
```

Άσκηση

- Δίνεται μία μεταβλητή τύπου λίστας myList, η οποία αρχικοποιείται ως ακολούθως:

  myList=[-5, 6, “Hello, World!”, [42, -42, True], [“One”, “Two”], 4.98]

  Ποια στοιχεία της λίστας θα επιστραφούν αν εκτελεστούν οι εξής εκφράσεις τεμαχισμού;

  i. myList[2:5]

  ii. myList[:-3]

  iii. myList[:3]

  iv. myList[2:7:2]

  v. myList[3:4][0][2]

  vi. myList[::-4]

Άσκηση

- Δίνεται μία μεταβλητή τύπου λίστας myList, η οποία αρχικοποιείται ως ακολούθως:

  myList=[-5, 6, “Hello, World!”, [42, -42, True], [“One”, “Two”], 4.98]

  Ποια στοιχεία της λίστας θα επιστραφούν αν εκτελεστούν οι εξής εκφράσεις τεμαχισμού;

  i. myList[2:5] ['Hello, World!', [42, -42, True], ['One', 'Two']]

  ii. myList[:-3] [-5, 6, 'Hello, World!']

  iii. myList[:3] [-5, 6, 'Hello, World!']

  iv. myList[2:7:2] ['Hello, World!', ['One', 'Two']]

  v. myList[3:4][0][2] True # myList[3:4] επιστρέφει λίστα [[42,-42,True]] με ένα μόνο στοιχείο στη θέση 0

  vi. myList[::-4] [4.98, 6] # Από το τέλος της λίστας έως την αρχή, επιστροφή κάθε τέταρτου στοιχείου

Τύποι δεδομένων στη γλώσσα Python

- Πλειάδες (tuple)
  - Στατική δομή δεδομένων για διατεταγμένη συλλογή στοιχείων διαφορετικού τύπου δεδομένων
    - **Στατική:** **ΔΕΝ** μπορούν να τροποποιηθούν τα περιεχόμενά μιας πλειάδας
    - **Διατεταγμένη:** Η σειρά των στοιχείων εντός της πλειάδας παίζει ρόλο.
    - Τα στοιχεία της πλειάδας μπορεί να είναι διαφορετικού τύπου δεδομένων
      - int, float, str, list ακόμη και tuples
    - Δημιουργία πλειάδας με τη χρήση παρενθέσεων () και διαχωρισμός στοιχείων/τιμών με κόμμα (,)

```
>>> myTuple=('banana', 'apple', 1, 2,3) #δημιουργία μεταβλητής τύπου πλειάδα που περιέχει 5 στοιχεία: 2 τύπου συμβολοσειράς και 3 τύπου ακέραιος.
```

Τύποι δεδομένων στη γλώσσα Python

- Πλειάδες (tuple)
  - Δεν παρέχουν μεθόδους ή τελεστές για την τροποποίηση των στοιχείων που διατηρούν.
  - Δεικτοδότηση μεμονωμένων στοιχείων ή τμημάτων πλειάδας με τον τελεστή []: [start:end:step]
    - Σύνταξη και ερμηνεία ίδια με αυτήν της τμηματοποίησης συμβολοσειρών και λιστών
      - start, end: θετικοί ή/και αρνητικοί

```
>>> myTuple=('banana', 'apple', 1, 2,3) #δημιουργία μεταβλητής τύπου
                                        πλειάδας που περιέχει 5 στοιχεία: 2
                                        τύπου συμβολοσειράς και 3 τύπου
                                        ακέραιου.
>>>myTuple[-3:] #Επιστροφή τελευταίων 3 στοιχείων της πλειάδας
(1, 2, 3)
```

Τύποι δεδομένων στη γλώσσα Python

- Πλειάδες (tuple)
  - Πλειάδα μπορεί να μετατραπεί σε λίστα με τη χρήση της list()
  - Λίστα μπορεί να μετατραπεί σε πλειάδα με χρήση της tuple()

```
>>> myTuple=('banana', 'apple', 1, 2, 3)
>>>myList = list(myTuple) #Μετατροπή της πλειάδας myTuple σε λίστα και
                          ανάθεση λίστας στη μεταβλητή myList
>>>myList
['banana', 'apple', 1, 2, 3]
>>>anotherTuple = tuple(myList) #Μετατροπή της λίστας myList σε πλειάδα και
                                 ανάθεση πλειάδας στη μεταβλητή
                                 anotherTuple
>>>anotherTuple
('banana', 'apple', 1, 2, 3)
```

Τύποι δεδομένων στη γλώσσα Python

- Σύνολα (Set)
  - Δυναμική δομή δεδομένων για μια μη διατεταγμένη συλλογή διακριτών, αμετάβλητων στοιχείων διαφορετικού τύπου δεδομένων
    - **Δυναμική**: μπορούν να προστεθούν/αφαιρεθούν στοιχεία
    - **Μη διατεταγμένη**: Δεν υπάρχει διάταξη/σειρά μεταξύ των στοιχείων του συνόλου.
    - Διακριτά στοιχεία: μία τιμή εμφανίζεται μία μόνο φορά στο σύνολο
    - Αμετάβλητα στοιχεία: τα στοιχεία του συνόλου δεν μπορούν να τροποποιηθούν. Το ίδιο το σύνολο μπορεί.
    - Τα στοιχεία ενός συνόλου μπορεί να είναι διαφορετικών τύπων δεδομένων
      - int, float, str, **αλλά όχι tuple, list ή set**
    - Δημιουργία συνόλου με τη χρήση αγκιστρών {} και διαχωρισμός στοιχείων/τιμών με κόμμα (,)

```
>>> mySet={‘banana’, ‘apple’, 1, 2,3} #δημιουργία μεταβλητής τύπου σύνολο
                                      που περιέχει 5 στοιχεία: 2 τύπου
                                      συμβολοσειράς και 3 τύπου ακέραιος.
```

Τύποι δεδομένων στη γλώσσα Python

- Σύνολα (Set)
  - Δεν υποστηρίζεται δεικτοδότηση μεμονωμένων στοιχείων ή τμημάτων με τον τελεστή [start:end:step] καθότι είναι **μη-διατεταγμένη συλλογή** στοιχείων.
  - Προσθήκη/αφαίρεση στοιχείων συνόλου
    - Προσθήκη
      - Μέθοδος .add() προσθέτει στο σύνολο ένα μεμονωμένο στοιχείο που δίνεται ως όρισμα
      - Μέθοδος .update() δέχεται πολλαπλά ορίσματα τύπου λίστας, πλειάδας ή σύνολο και προσθέτει στο σύνολο κάθε στοιχείο που εμφανίζεται στους τύπους δεδομένους αυτούς.
    - Αφαίρεση
      - Μέθοδος .discard() αφαιρεί από το σύνολο η τιμή που δίνεται ως όρισμα. Αν η τιμή δεν υπάρχει στο σύνολο, δεν προκύπτει σφάλμα
      - Μέθοδος .remove() αφαιρεί από το σύνολο η τιμή που δίνεται ως όρισμα. Αν η τιμή δεν υπάρχει στο σύνολο, προκαλείται σφάλμα και το πρόγραμμα διακόπτει την εκτέλεση

Τύποι δεδομένων στη γλώσσα Python

- Σύνολα (Set)
  - Πράξεις μεταξύ συνόλων
    - Τελεστής in: εξετάζει αν μία τιμή είναι στοιχείο ενός συνόλου. Επιστρέφει true/false
    - Μέθοδος .union(): επιστρέφει νέο σύνολο που είναι η ένωση δύο συνόλων
    - Μέθοδος .intersection(): επιστρέφει νέο σύνολο που είναι η τομή δυο συνόλων
    - Μέθοδος .difference(): επιστρέφει νέο σύνολο που είναι η διαφορά των δύο συνόλων
    - Μέθοδος .symmetric_difference(): επιστρέφει νέο σύνολο που περιέχει εκείνα τα στοιχεία τα οποία υπάρχουν σε ένα μόνο από τα δύο σύνολα

Άσκηση

- Γιατί τα στοιχεία ενός συνόλου (set) στην Python δεν μπορεί να είναι τύπου δεδομένων λίστας (list) ή συνόλου (set);

Άσκηση

- Γιατί τα στοιχεία ενός συνόλου (set) στην Python δεν μπορεί να είναι τύπου δεδομένων λίστας (list) ή συνόλου (set);

  Απάντηση

  Γιατί στον ορισμό του συνόλου υπάρχει η απαίτηση τα στοιχεία του συνόλου να είναι αμετάβλητα (immutable). Η λίστα και το σύνολο δεν είναι αμετάβλητοι τύποι δεδομένων.

Τύποι δεδομένων στη γλώσσα Python

- Λεξικά (Dictionaries ή Dict)
  - Δυναμική συλλογή δεδομένων για μια μη διατεταγμένη συλλογή ζευγών κλειδιού-τιμής. Τα κλειδιά δεικτοδοτούν τις τιμές του λεξικού.
    - Δυναμική: Ζεύγη κλειδιών-τιμές μπορούν να προστεθούν/αφαιρεθούν από το λεξικό
    - Μη διατεταγμένη: Δεν υπάρχει διάταξη/σειρά μεταξύ των στοιχείων του λεξικού.
    - Οι τιμές κλειδιών μπορεί να είναι διαφορετικού τύπου δεδομένων
      - int, float, str, tuple, list, tuple, set κλπ
    - Δημιουργία λεξικού με τη χρήση αγκιστρών {} και ζευγών της μορφής *κλειδί:τιμή* που διαχωρίζονται με κόμμα (,)

```
>>> myDict={‘name’:’Jim’, ‘age’:28} #Δημιουργία μεταβλητής τύπου λεξικό
                                    που περιέχει 2 ζεύγη κλειδιού-τιμής:
                                    κλειδί name με τιμή Jim, κλειδί age με
                                    τιμή 28
```

Τύποι δεδομένων στη γλώσσα Python

- Λεξικά (Dict)
  - Δεικτοδότηση/ανάκτηση τιμών λεξικού γίνεται με χρήση του ονόματος του κλειδιού και τον τελεστή [<όνομα κλειδιού>]
  - Κλειδιά
    - Μπορεί να είναι οποιουδήποτε αμετάβλητου τύπου δεδομένων όπως συμβολοσειρά, ακέραιοι αλλά όχι λίστες
      - Αν το κλειδί είναι συμβολοσειρά, πεζά-κεφαλαία παίζουν ρόλο (case sensitive)
    - Τα κλειδιά είναι μοναδικά εντός ενός λεξικού

```
# Δημιουργία μεταβλητής myDict τύπου λεξικό με 3 ζευγή κλειδιού-τιμής. Οι
# τιμές έχουν τύπο δεδομένων συμβολοσειρά ('Jim'), ακέραιο (28) και
# λίστα([123,456]) ενώ τα κλειδιά είναι όλα τύπου συμβολοσειράς
>>> myDict={ 'name':'Jim', 'age':28, 'telephone':[123,456] }
>>>myDict['name'] #Δεικτοδότηση/ανάκτηση τιμής του κλειδιού 'name'
Jim
>>>myDict['telephone'] #Ανάκτηση τιμής του κλειδιού 'telephone'
[123, 456]
```

Τύποι δεδομένων στη γλώσσα Python

- Λεξικά (Dict)
  - Πράξεις
    - Ενημέρωση τιμής υπάρχοντος κλειδιού με τον τελεστή [<υπάρχον κλειδί>]
    - Προσθήκη νέου ζεύγους κλειδιού-τιμής με τον τελεστή [<νέο κλειδί>]
    - Αφαίρεση ζεύγους κλειδιού τιμής με τον τελεστή del

```
>>> myDict={ 'name':'Jim', 'age':28, 'telephones':[123,456] }
>>>myDict['age'] = 33 #Ενημέρωση της τιμής του υπάρχοντος κλειδιού 'age'
>>>myDict
{ 'name':'Jim', 'age':33, 'telephones':[123,456] }
>>>myDict['address']='Rodeo drive' #Προσθήκη νέου ζεύγους κλειδί-τιμής
>>>myDict
{ 'name':'Jim', 'age':33, 'telephones':[123,456], 'address':'Rodeo drive' }
>>>del myDict['telephones'] #Αφαίρεση του ζεύγους με κλειδί 'telephones' από το λεξικό
>>>myDict
{ 'name':'Jim', 'age':33, 'address':'Rodeo drive' }
```

Τύποι δεδομένων στη γλώσσα Python

- Λεξικά (Dict)
  - Μέθοδοι λεξικών (help(dict) για περισσότερες μεθόδους)

Μέθοδος	Σημασία
.get(key, [,value])	Επιστρέφει την τιμή του κλειδιού (key, που δίνεται ως συμβολοσειρά) στο λεξικό. Αν δοθεί όρισμα value, θα επιστραφεί η τιμή value αν το κλειδί δεν υπάρχει στο λεξικό.
.keys()	Επιστρέφει λίστα με τα ονόματα κλειδιών που υπάρχουν στο λεξικό
.values()	Επιστρέφει λίστα με μόνο τις τιμές που υπάρχουν στο λεξικό
.clear()	Αφαιρεί όλα τα ζεύγη κλειδιού-τιμής που υπάρχουν στο λεξικό

```
>>> myDict={‘name’:’Jim’, ‘age’:28}
>>>myDict.keys() # Επιστροφή όλων των κλειδιών του λεξικού ως λίστα
dict_keys(['name', 'age’])
>>>myDict.values() # Επιστροφή όλων των τιμών του λεξικού
dict_values(['Jim', 28])
>>>myDict.clear() # Αφαίρεση όλων των ζευγών κλειδιού-τιμής από το λεξικό
```

Άσκηση

- Τί θα εμφανίσει η εκτέλεση του παρακάτω προγράμματος Python;

```
aDict = { ‘name’: ‘Aguirre’, ‘grades’:[5.6, 7.9, 2.4, 6.5]}
print( aDict[‘name’] )
print( aDict.get(‘name’)[2] )
print( aDict.get(‘grades’)[:-2] )
print( aDict.get(‘Address’, “Error – Not found”) )
print( aDict.get(‘Address’, None) )
```

Άσκηση

- Τί θα εμφανίσει η εκτέλεση του παρακάτω προγράμματος Python;

```
aDict = { ‘name’: ‘Aguirre’, ‘grades’:[5.6, 7.9, 2.4, 6.5]}
print( aDict[‘name’] )
print( aDict.get(‘name’)[2] )
print( aDict.get(‘grades’)[:-2] )
print( aDict.get(‘Address’, “Error – Not found”) )
print( aDict.get(‘Address’, None) )
```

Αποτέλεσμα

```
‘Aguirre’
u
[5.6, 7.9]
Error – Not found
None
```

Άσκηση

- Η python δεν υποστηρίζει εγγενώς δισδιάστατους πίνακες (2D arrays) όπως κάνουν πολλές άλλες γλώσσες προγραμματισμού. Με ποιον τρόπο θα μπορούσαν να υποστηριχθούν δισδιάστατοι πίνακες στην python;

Άσκηση

- Η python δεν υποστηρίζει εγγενώς δισδιάστατους πίνακες (2D arrays) όπως κάνουν πολλές άλλες γλώσσες προγραμματισμού. Με ποιον τρόπο θα μπορούσαν να υποστηριχθούν δισδιάστατοι πίνακες στην python;

  Απάντηση

  Με τη χρήση λίστας, που έχει ως στοιχεία λίστες.

```
>>> array2D=[ [1,2,3], [4,5,6] ] #Ορισμός δισδιάστατου πίνακα 2 γραμμών και 3
                                  στηλών, με πρώτη γραμμή τη λίστα [1,2,3] και
                                  δεύτερη γραμμή τη λίστα [5,6,7]
>>>array2D[0][2] #Στοιχείο στην πρώτη γραμμή, τρίτη στήλη
3
>>>array2D[1][0] = 7 #Ενημέρωση στοιχείου στη δεύτερη γραμμή, πρώτη
                      στήλη. Η τιμή 4 θα γίνει 7.
```

Τύποι δεδομένων

- Επαναλήψιμοι τύποι δεδομένων (iterable)
  - Τύποι δεδομένων όπου είναι δυνατόν η προσπέλαση των στοιχείων που διατηρούν, ένα προς ένα, με τη σειρά που εμφανίζονται στον τύπο δεδομένων αυτόν.
  - Επαναλήψιμοι τύποι δεδομένων στην Python
    - Συμβολοσειρές
    - Λίστες
    - Πλειάδες
    - Σύνολα
    - Λεξικά

Τύποι δεδομένων

- Επαναλήψιμοι τύποι δεδομένων δίνουν την εξής δυνατότητα σύνταξης επαναλήψεων με for:

```
aString="Pulchra enim sunt ubera"
for  element in aString:
        print(element)
```

```
P
u
l
c
h
r
a

e
n
i
m
.
.
.
```

```
myList=['banana', 1, 2, [True, 9] ]
for  element in myList:
        print(element)
```

```
'banana'
1
2
[True, 9]
```

Συμβολοσειρές και λίστες είναι επαναλήψιμοι τύποι δεδομένων.

Συμπεριφορά μεταβαλλόμενων τύπων δεδομένων

- Κατά τις πράξεις τροποποίησης τιμής οι μεταβαλλόμενοι τύποι δεδομένων συμπεριφέρονται διαφορετικά από τους αμετάβλητους τύπους δεδομένων

Αμετάβλητοι τύποι δεδομένων	Μεταβαλλόμενοι τύποι δεδομένων
Ακέραιοι (int),Αριθμοί κινητής υποδιαστολής (float), Συμβολοσειρές (string), Πλειάδες (tuple), Λογικές τιμές (bool).	**Λίστες (list), Σύνολα (set), Λεξικά (dict)**

Συμπεριφορά μεταβαλλόμενων τύπων δεδομένων

- Συμπεριφορά πράξεων π.χ. σε Λίστες
  - .append(), .insert() ή ο τελεστής επαυξημένης εκχώρησης (+=) σε λίστες δεν δημιουργούν νέο αντικείμενο. Τροποποιούν το αντικείμενο όπου αναφέρονται οι μεταβλητές.

```
>>> l=[1,2,3,4] #Ορισμός μεταβαλλόμενου τύπου, λίστας
>>>m = l  # Η μεταβλητή m αναφέρεται στο ίδιο αντικείμενο με τη l
>>>id(l)
2599478153728
>>>id(m)                  # l και m αναφέρονται στο ίδιο πλέον αντικείμενο
2599478153728
>>>l.append(5) # Τροποποίηση αντικειμένου μέσω της αναφοράς l: προσθήκη νέου στοιχείου στη
                                    λίστα
>>> l
[1, 2, 3, 4, 5]
>>> m
[1, 2, 3, 4, 5] # Το νέο στοιχείο που προστέθηκε μέσω της l υπάρχει και στη λίστα m, αφού
                                    αναφέρονται στο ίδιο αντικείμενο και έγινε χρήση της .append().
>>> id(l)
2599478153728                # l και m συνεχίζουν να αναφέρονται στο ίδιο αντικείμενο
>>>id(m)                    (μεταβαλλόμενοι τύποι) παρόλο που αυτό έχει μεταβληθεί.
2599478153728
```

Συμπεριφορά μεταβαλλόμενων τύπων δεδομένων

- Συμπεριφορά πράξεων π.χ. σε Λίστες
  - .append(), .insert() ή ο τελεστής επαυξημένης εκχώρησης (+=) σε λίστες δεν δημιουργούν νέο αντικείμενο.

```
>>> l += [6]  # Χρήση τελεστή επαυξημένης εκχώρησης += για προσθήκη του στοιχείο 6 στη λίστα l
>>> l
[1, 2, 3, 4, 5, 6]
>>>m                # l και m συνεχίζουν να αναφέρονται στο ίδιο αντικείμενο επειδή η προσθήκη
[1, 2, 3, 4, 5, 6]  έγινε μέσω του τελεστή επαυξημένης εκχώρησης +=
>>>id(l)
2599478153728
>>>id(m)
2599478153728
```

Συμπεριφορά μεταβαλλόμενων τύπων δεδομένων

- Συμπεριφορά πράξεων π.χ. σε Λίστες
  - Αντιθέτως, επέκταση της λίστας με τελεστή απλής ανάθεσης = θα οδηγήσει σε νέο αντικείμενο.

```
>>> l = l + [7]  #Χρήση τελεστή απλής ανάθεσης = για επέκταση της λίστας l με νέο στοιχείο, 7
>>> l
[1, 2, 3, 4, 5, 6, 7]
>>>m
[1, 2, 3, 4, 5, 6]      # l και m αναφέρονται σε ΔΙΑΦΟΡΕΤΙΚΑ πλέον αντικείμενα επειδή η επέκταση της
>>>id(l)                 λίστας - μέσω της αναφορά l - έγινε με τον τελεστή απλής ανάθεσης =
2497535787904
>>>id(m)
2599478153728
```

Τέλος Υπο-ενότητας