Αντικειμενοστρεφής Προγραμματισμός - Python

Κ.Π. Γιαλούρης

Στόχοι του σημερινού μαθήματος

- Κατανόηση της έννοιας της λίστας (**`list`**).
- Χειρισμός Λίστας.
  - Δημιουργία
  - Εκτύπωση λίστας
  - Έλεγχος ύπαρξης στοιχείου σε λίστα
  - Εισαγωγή στοιχείου
  - Αφαίρεση στοιχείου
  - Επέκταση λίστας
  - Ταξινόμηση λίστας
  - Σύγκριση λιστών

Η έννοια της λίστας

- Μία λίστα είναι ένα αντικείμενο το οποίο περιέχει πολλαπλά δεδομένα.
- Μία λίστα είναι ένας τύπος μεταβλητής
- Μία λίστα είναι ένας δυναμικός τύπος δεδομένων. Αυτό σημαίνει ότι μπορούμε να προσθέσουμε ή να αφαιρέσουμε στοιχεία σε μία λίστα.

Η έννοια της λίστας

```
ar=[2, 4, 6, 8,10]
```

Η λίστα είναι ένα διατεταγμένο σύνολο δεδομένων.

Η τάξη/αρίθμηση των στοιχείων αρχίζει από το 0 (μηδέν)

Κάθε στοιχείο μίας λίστας μπορεί να προσπελαστεί μέσω ενός δείκτη που καθορίζει τη θέση του στη λίστα

Η έννοια της λίστας

```
ar=[2, 4, 6, 8,10]
onomata=['Νικολάου', 'Πέτρου', 'Γεωργίου']
vathmoi=['Πέτρου', 10, 8, 7]
```

Εκτύπωση λίστας

```
ar=[2, 4, 6, 8,10]
onoma=['Νικολάου', 'Πέτρου', 'Γεωργίου']

print (ar)
[2, 4, 6, 8,10]
print (onoma[2])
Γεωργίου
```

Εκτύπωση λίστας

```
ar=[2, 4, 6, 8,10]
onomata=['Νικολάου', 'Πέτρου', 'Γεωργίου']

print (ar)
[2, 4, 6, 8,10]
print(onomata)
['Νικολάου', 'Πέτρου', 'Γεωργίου']
```

Προσπέλαση μέσω του :

- Εκτός της προσπέλαση μέσω δείκτη υπάρχει και συνδυασμός δείκτη με το σύμβολο ‘:’

Προσπέλαση μέσω του :

- Εκτός της προσπέλαση μέσω δείκτη υπάρχει και συνδυασμός δείκτη με το σύμβολο ‘:’

Προσπέλαση μέσω του :

`lista[:n]`

Προσπέλαση των n πρώτων στοιχείων της λίστας

`lista[n:]`

Προσπέλαση των n τελευταίων στοιχείων της λίστας

Προσπέλαση μέσω του :

`lista[:n]`

Προσπέλαση των n πρώτων στοιχείων της λίστας

`lista[-n:]`

Προσπέλαση των n τελευταίων στοιχείων της λίστας

Προσπέλαση μέσω του :

`lista[n:]`

Προσπέλαση των στοιχείων της λίστας μετά το n-στο στοιχείο

`lista[n:m:k]`

Προσπέλαση των στοιχείων της λίστας από το n-στο στοιχείο μέχρι το m-στο και με βήμα k.

Αν n>m τότε το πρέπει να είναι k<0

Προσπέλαση μέσω του :

`lista[::n]`

Προσπέλαση των στοιχείων της λίστας με επιλογή ανά nστοιχεία

`lista[n:-m]`

Προσπέλαση των στοιχείων της λίστας αφαιρώντας το πρώτα n και τα τελευταία m στοιχεία

Προσπέλαση μέσω του : Παράδειγμα

```
li=[0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38]
print("εκτύπωση  των 5 πρώτων στοιχείων της λίστας")
print(li[:5]) #[0, 2, 4, 6, 8]

print("εκτύπωση των  στοιχείων της λίστας αφαιρώντας τα 5 πρώτα")
print(li[5:])# [10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38]

print("εκτύπωση ανά 3 των στοιχείων της λίστας")
print(li[::3])#[0, 6, 12, 18, 24, 30, 36]

print("εκτύπωση των στοιχείων της λίστας αφαιρώντας το 2 πρώτα και τα 3 τελευταία")
print(li[2:-3])#[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]

print("εκτύπωση των 4 τελευταίων στοιχείων της λίστας")
print(li[-4:])#[32, 34, 36, 38]

print("εκτύπωση των στοιχείων από το 10ο ανά δύο μέχρι το 2ο")
print(li[10:2:-2]) #[20, 16, 12, 8]

print("εκτύπωση των στοιχείων από το 5ο ανά δύο μέχρι το 15ο")
print(li[5:15:2]) #[10, 14, 18, 22, 26]
```

εκτύπωση όλων των στοιχείων της λίστας

```
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38]
εκτύπωση  των 5 πρώτων στοιχείων της λίστας
[0, 2, 4, 6, 8]
εκτύπωση των  στοιχείων της λίστας αφαιρώντας τα 5 πρώτα
[10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38]
εκτύπωση ανά 3 των στοιχείων της λίστας
[0, 6, 12, 18, 24, 30, 36]
εκτύπωση των στοιχείων της λίστας αφαιρώντας το 2 πρώτα και τα 3 τελευταία
[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32]
εκτύπωση των 4 τελευταίων στοιχείων της λίστας
[32, 34, 36, 38]
εκτύπωση των στοιχείων από το 10ο ανά δύο μέχρι το 2ο
[20, 16, 12, 8]
εκτύπωση των στοιχείων από το 5 ανά δύο μέχρι το 15ο
[4, 8, 12, 16, 20, 24, 28]
```

Μέθοδοι χειρισμού λίστας

- `append(στοιχείο)`
- `index(στοιχείο)`
- `insert (δείκτης, στοιχείο)`
- `sort()`
- `remove (στοιχείο)`
- `reverse()`

append

Με τη μέθοδο **`append`** επιτυγχάνεται εισαγωγή ενός στοιχείου σε μία υπάρχουσα λίστα. Το στοιχείο που θα εισαχθεί να τοποθετηθεί στο τέλος της λίστας.

Σύνταξη:

```
lista.append(στοιχείο)
```

append

Παράδειγμα: Έστω η λίστα **students**

```
students=
["Νίκος","Γιώργος","Μαρία","Πέτρος","Ανδρέας"]
```

Μετά την εκτέλεσης της:

```
students.append("Ελένη")
```

η λίστα θα είναι η:

```
Students=["Νίκος","Γιώργος","Μαρία","Πέτρος","Ανδρέας",
"Ελένη"].
```

`insert`

Η μέθοδος **`insert`** δέχεται δύο ορίσματα. Το πρώτο όρισμα ορίζει την τάξη (θέση) μέσα στη λίστα ενώ το δεύτερο όρισμα είναι το προς εισαγωγή στοιχείο.

Σύνταξη:

`lista.insert(δείκτης, στοιχείο)`

`insert`

Η μέθοδος **`insert`** δέχεται δύο ορίσματα. Το πρώτο όρισμα ορίζει την τάξη (θέση) μέσα στη λίστα ενώ το δεύτερο όρισμα είναι το προς εισαγωγή στοιχείο.

Σύνταξη:

`lista.insert(δείκτης, στοιχείο)`

remove

Με τη μέθοδο **remove** επιτυγχάνεται διαγραφή ενός στοιχείου σε μία υπάρχουσα λίστα.

Εάν το στοιχείο υπάρχει περισσότερες από μία φορές διαγράφει το πρώτο που θα βρει.

Εάν το στοιχείο δεν υπάρχει δημιουργείται λάθος εκτέλεσης του προγράμματος

Σύνταξη:

```
lista.remove(στοιχείο)
```

remove

Παράδειγμα:

Έστω η λίστα **students** με τις παρακάτω τιμές :

["Νίκος","Γιώργος","Μαρία","Πέτρος", "Ανδρέας","Ελένη"]

- Η εντολή **students.remove("Μαρία")** θα διαγράψει το στοιχείο με όνομα **"Μαρία"**.

index

Η μέθοδος **index** δέχεται ως όρισμα ένα στοιχείο μιας λίστας και επιστρέφει την τάξη του στη λίστα.

Έστω η λίστα **students** με τις παρακάτω τιμές. Αν το στοιχείο δεν υπάρχει στη λίστα τότε δημιουργείται λάθος εκτέλεσης.

Σύνταξη:

μεταβλητή=lista.index(στοιχείο)

index

Παράδειγμα έστω η λίστα **students** με τιμές :

["Νίκος", "Γιώργος", "Μαρία", "Πέτρος", "Ανδρέας", "Ελένη"]

Η εντολή **print(students.index("Ελένη")** θα εμφανίσει τον αριθμό **5**. Δεδομένου ότι η τάξη του πρώτου στοιχείου είναι το 0 (μηδέν)

sort

Η μέθοδος **sort** ταξινομεί μία λίστα.

Σύνταξη:

```
lista.sort()                    # ταξινόμηση αύξουσα
lista.sort(reverse=True)        # ταξινόμηση φθίνουσα
```

reverse

Η μέθοδος **reverse** αντιστρέφει τη σειρά των στοιχείων μιας λίστας

Σύνταξη:

```
lista.reverse()
```

count

Η μέθοδος **count** επιστρέφει το πλήθος των επαναλήψεων ενός στοιχείου μιας λίστας.

Σύνταξη:

```
lista.count(στοιχείο)
```

Συναρτήσεις με όρισμα λίστα

α/α	Όνομα συνάρτησης	Λειτουργία
1	**`len`**	Δέχεται ως όρισμα μία λίστα και επιστρέφει το πλήθος των στοιχείων της λίστας
2	**`min`**	Δέχεται ως όρισμα μία λίστα και επιστρέφει το μικρότερο των στοιχείων της λίστας
3	**`max`**	Δέχεται ως όρισμα μία λίστα και επιστρέφει το μεγαλύτερο των στοιχείων της λίστας

Παραδείγματα

```
students = ["Νίκος", "Γιώργος", "Μαρία", "Πέτρος", "Ανδρέας"]
print(students)
```

```
students.append("Ελένη")
print(students)
print(students[4])
students.insert(3,"Τάσος")
print(students)
students.remove('Μαρία')
print(students)
del(students[2])
print(students)
students.sort()
print(students)
students.reverse()
print(students)
print (min(students),max(students))
```

Αποτελέσματα

```
['Νίκος', 'Γιώργος', 'Μαρία', 'Πέτρος', 'Ανδρέας']
['Νίκος', 'Γιώργος', 'Μαρία', 'Πέτρος', 'Ανδρέας', 'Ελένη']
Ανδρέας
['Νίκος', 'Γιώργος', 'Μαρία', 'Τάσος', 'Πέτρος', 'Ανδρέας',
'Ελένη']
['Νίκος', 'Γιώργος', 'Τάσος', 'Πέτρος', 'Ανδρέας', 'Ελένη']
['Νίκος', 'Γιώργος', 'Πέτρος', 'Ανδρέας', 'Ελένη']
['Ανδρέας', 'Γιώργος', 'Ελένη', 'Νίκος', 'Πέτρος']
['Πέτρος', 'Νίκος', 'Ελένη', 'Γιώργος', 'Ανδρέας']
Ανδρέας Πέτρος
```

Πρόσθεση λιστών

Με την έννοια **πρόσθεση λιστών** εννοούμε τη συνένωση

```
a=[1,2,3,4]
b=[5,6,7,8,9]
x=a+b
```

Η λίστα **x** θα έχει τη μορφή

```
x=[1,2,3,4,5,6,7,8,9]
```

Πολλαπλασιασμός λίστας με αριθμό

```
a=[1,2,3,4]
x=a*2
```

- Η λίστα θα είναι
  - x=[1,2,3,4,1,2,3,4]