UNIVERSITATEA TEHNICĂ A MOLDOVEI

Programare interactivă

Tema: Colecții de date – Liste

lect. univ. Victoria ALEXEI
Departament Informatică și Ingineria Sistemelor
Facultatea Calculatoare, Informatică și Microelectronică

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Liste în Python, prelucrarea listelor

În acest curs vom învăța:

- Lista – noțiuni elementare
- Operații cu liste
- Inserări şi eliminări de componente dintr-o listă, slice-ing
- Ordonarea componentelor unei liste
- Metode aplicabile listelor

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Lista – noțiuni elementare

O listă în Python este o structură de date mutabilă care poate conţine mai multe elemente

Listele sunt incluse **în paranteze pătrate**, elementele listei fiind separate prin virgulă

Listele pot păstra elemente de tipuri diferite, chiar și altă listă

Lista poate fi vidă

Exemple de liste

```
L = [3,56,8,45,90]

L1 = ['carte', 'caiect','pix',"creioane", "masa", "scaun"]

L2 = ['trei', 8, 5.6, 31, ['sapun', 1, 'abac'], -6 ]

L3 = []
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Crearea unei liste

O listă poate fi creată cu funcția predefinită list()

Exemple

```
L4 = list('caractere')

L5 = list(range(3,30,4))

L6= list(range(8))

L7 = list()
```

Rezultate

```
['c', 'a', 'r', 'a', 'c', 't', 'e', 'r', 'e']

[3, 7, 11, 15, 19, 23, 27]

[0, 1, 2, 3, 4, 5, 6, 7]

[]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Crearea unei liste cu ajutorul funcției range()

```
>>> help(range)
Help on class range in module builtins:

class range(object)
 |  range(stop) -> range object
 |  range(start, stop[, step]) -> range object
 |
 |  Return an object that produces a sequence of integers from start (inclusive)
 |  to stop (exclusive) by step.  range(i, j) produces i, i+1, i+2, ..., j-1.
 |  start defaults to 0, and stop is omitted!  range(4) produces 0, 1, 2, 3.
 |  These are exactly the valid indices for a list of 4 elements.
 |  When step is given, it specifies the increment (or decrement).
 |
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Crearea unei liste cu ajutorul funcției *range()*

```
>>> list(range(10))
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>>
>>> list(range(5,13))
[5, 6, 7, 8, 9, 10, 11, 12]
>>>
>>> list(range(3,23,3))
[3, 6, 9, 12, 15, 18, 21]
>>>
>>> list(range(10, -10, -3))
[10, 7, 4, 1, -2, -5, -8]
>>>
```

```
>>> s=list(range(5))
>>> s
[0, 1, 2, 3, 4]
>>> s=list(range(5,12))
>>> s
[5, 6, 7, 8, 9, 10, 11]
>>> s=list(range(5,20,3))
>>> s
[5, 8, 11, 14, 17]
>>>
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Operații cu liste

La fel ca șiruri de caractere, putem obține orice element dintr-o listă utilizând un **index specificat în paranteze pătrate.**

```
L1 = ['carte', 'caiect','pix',"creioane", "masa", "scaun"]
```

'carte'	'caiect'	'pix'	"creioane"	"masa"	"scaun"
0	1	2	3	4	5

Exemple

```
L1[3]
'creioane'
```

```
L1[1]
'caiect'
```

```
In [61]: L1[7]
Traceback (most recent call last):

  File "<ipython-input-61-73c592bbee41>", line 1, in <module>
    L1[7]

IndexError: list index out of range
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Operații cu liste

Concatinarea listelor – operatorul de concatinare“**+**”. Operatorul + este utilizat pentru concatenarea / unirea a două sau mai multe liste.

Exemplu

```
l1 = ['doi',34,'cinci',-8]
l2 = ['joi',"23",7,32]

rez = l1+l2
print(rez)
```

Rezultat

```
['doi', 34, 'cinci', -8, 'joi', '23', 7, 32]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Operații cu liste

Concatinarea listelor – operatorul de concatinare“**+**”.

Exemplu

```
l1 = ['doi',34,'cinci',-8]
rez1 = l1 + 'sase'
```

Rezultat

```
Traceback (most recent call last):

  File "C:\Users\Admin\Desktop\Lista.py", line 30, in <module>
    rez1 = l1 + 'sase'

TypeError: can only concatenate list (not "str") to list
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Operații cu liste

Repetarea listelor – operatorul de repetare “*”, este utilizat pentru a repeta o listă de un anumit număr de ori.

Exemple

```
l1 = ['doi',34,'cinci',-8]
rez1 = l1 * 3
print(rez1)
```

```
repl = [0] * 7
print(repl)
```

Rezultat

```
['doi', 34, 'cinci', -8, 'doi', 34, 'cinci', -8, 'doi', 34, 'cinci', -8]
```

```
[0, 0, 0, 0, 0, 0, 0]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Operații cu liste

Repetarea listelor – operatorul de repetare “*”

Exemplu

```
l1 = ['doi',34,'cinci',-8]
l2 = ['joi',"23",7,32]
```

```
rez2 = l1 * l2
```

NB! pentru adunare, ambele elemente trebuie să fie liste, iar pentru înmulţire este necesar o listă şi un număr întreg.

Rezultat

```
Traceback (most recent call last):

  File "C:\Users\Admin\Desktop\Lista.py", line 32, in <module>
    rez2 = l1 * l2

TypeError: can't multiply sequence by non-int of type 'list'
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Operații cu liste

Compararea listelor – operatorii de comparare ==, >,<, >=,<=

Exemple

```
l3 = [2,3,5]
l4 = [2,3,5]
print(l3 == l4)
```

Rezultate

```
True
```

Operatorul == returnează **True** dacă există o potrivire exactă, altfel **False** va fi returnat.
Operatorul **!** = returnează **True** dacă nu există potrivire.

```
l1 = ['doi',34,'cinci',-8]
l2 = ['joi',"23",7,32]
print(l1 == l2)
```

```
False
```

```
print([2,6,7] > [1,7,3])
print([1,6,7] > [1,7,3])
print([1,6,7] < [1,7,3])
```

```
True
False
True
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Operații cu liste

Apartenența unui element a unei liste cu instrucțiunea ”**in**”, ”**not in**”

in: Acesta verifică dacă un element este prezent într-o listă sau nu.

Exemplu

```
['doi', 34, 'cinci', -8, 'doi', 34, 'cinci', -8, 'doi', 34, 'cinci', -8]
```

```
print('doi' in rez1)
print('do' in rez1)
print(-8 in rez1)
print('34' not in rez1)
print(34 in rez1)
```

Rezultat

```
True
False
True
True
True
```

not in: Funcționează exact opusul la ceea ce face operatorul „in”.

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Operații cu liste

Copierea conținutului listelor – două modalități:

Copiere prin referință – indică aceeași adresă de memorie

Exemplu

```
l5 = [9,'doi',5,'m']
l6 = l5
print(l6)

l6 += ['trei']
print(l5)
print(l6)
```

Rezultat

```
[9, 'doi', 5, 'm']
[9, 'doi', 5, 'm', 'trei']
[9, 'doi', 5, 'm', 'trei']
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Operații cu liste

Copierea conținutului listelor – două modalități:

Copiere prin valoare – indică adrese de memorie diferite

Exemplu

```
l5 = [9,'doi',5,'m']
l7 = list(l5)
print(l7)

l7 += [0]
print(l7)
print(l5)
```

Rezultat

```
[9, 'doi', 5, 'm']
[9, 'doi', 5, 'm', 0]
[9, 'doi', 5, 'm']
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Inserări şi eliminări de componente dintr-o listă, slice-ing

- Selectarea unui element dintr-o listă – operatorul [ ]
- Indexarea începe de la 0

Exemple

```
rez
['doi', 34, 'cinci', -8, 'joi', '23', 7, 32]

rez[3]
-8

rez[2-6]
'joi'

rez[-1]
32
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Inserări şi eliminări de componente dintr-o listă, slice-ing

- Selectarea mai multor elemente - notaţia **slice, felierea L[start: stop: step]**
- Notaţia slice nu include indexul din dreapta.

```
rez
['doi', 34, 'cinci', -8, 'joi', '23', 7, 32]
rez[2:6]
['cinci', -8, 'joi', '23']

rez[:5]
['doi', 34, 'cinci', -8, 'joi']
rez[3:]
[-8, 'joi', '23', 7, 32]
rez[0:5:2]
['doi', 'cinci', 'joi']

rez[0:5:3]
['doi', -8]
```

L [start : stop : step]

Exemple cu rezultate

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Inserări şi eliminări de componente dintr-o listă, slice-ing

- Selectarea mai multor elemente - notaţia **slice, felierea**
- Notaţia slice nu include indexul din dreapta.

Exemple cu rezultate

L [start : stop : step]

```
rez[::]
['doi', 34, 'cinci', -8, 'joi', '23', 7, 32]

rez[::-1]
[32, 7, '23', 'joi', -8, 'cinci', 34, 'doi']

rez[len(rez)::-1]
[32, 7, '23', 'joi', -8, 'cinci', 34, 'doi']
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Inserări şi eliminări de componente dintr-o listă, slice-ing

Exemple cu rezultate

```
rez
['doi', 34, 'cinci', -8, 'joi', '23', 7, 32]

rez[0:0]= [1]

rez
[1, 'doi', 34, 'cinci', -8, 'joi', '23', 7, 32]

rez[3:3]=[5,6]

rez
[1, 'doi', 34, 5, 6, 'cinci', -8, 'joi', '23', 7, 32]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Inserări şi eliminări de componente dintr-o listă, slice-ing

Exemple cu rezultate

```
rez
[1, 'doi', 34, 5, 6, 'cinci', -8, 'joi', '23', 7, 32]

rez[0:3]=[9]

rez
[9, 5, 6, 'cinci', -8, 'joi', '23', 7, 32]

rez[6:]=[]

rez
[9, 5, 6, 'cinci', -8, 'joi']
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Inserări şi eliminări de componente dintr-o listă, slice-ing

Exemplu

```
In [63]: rez[0:0]=1
Traceback (most recent call last):

  File "<ipython-input-63-2624b5b3a0b1>", line 1, in <module>
    rez[0:0]=1

TypeError: can only assign an iterable
```

```
rez
[9, 5, 6, 'cinci', -8, 'joi']

rez[0:0]='A'

rez
['A', 9, 5, 6, 'cinci', -8, 'joi']
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Inserări şi eliminări de componente dintr-o listă, slice-ing

Exemplu

```
>>> pers = [300,'Irina', 3.12, [56,'Ion',78], 'Maria',4]
>>> pers[3][1]
'Ion'
>>> pers[4][1]
'a'
>>> pers[3][1][0]
'I'
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Ordonarea componentelor unei liste
Funcții utile len(), del, sorted(), min(), max(), sum()

Exemple

```
rez
['A', 9, 5, 6, 'cinci', -8, 'joi']

len(rez)
7

del rez[5]

rez
['A', 9, 5, 6, 'cinci', 'joi']
```

```
lista1 = ['car', 'transport', 'util', 'este', 'aratos']

sorted(lista1)
['aratos', 'car', 'este', 'transport', 'util']

sorted(lista1, reverse=True)
['util', 'transport', 'este', 'car', 'aratos']

sorted(lista1, key=len)
['car', 'util', 'este', 'aratos', 'transport']
```

```
lista1
['car', 'transport', 'util', 'este', 'aratos']
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Ordonarea componentelor unei liste
Funcții utile len(), del, sorted(), min(), max(), sum()

Exemple

```
lista = [3,7,5,2,9,6,23,1]

sorted(lista)
[1, 2, 3, 5, 6, 7, 9, 23]

sorted(lista,reverse=True)
[23, 9, 7, 6, 5, 3, 2, 1]
```

```
max(lista)
23

min(lista)
1

sum(lista)
56
```

```
len(lista)
8

del lista[7]

lista
[3, 7, 5, 2, 9, 6, 23]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

```
print(dir(lista))
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__delitem__', '__dir__',
'__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__gt__',
'__hash__', '__iadd__', '__imul__', '__init__', '__init_subclass__', '__iter__', '__le__',
'__len__', '__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__',
'__repr__', '__reversed__', '__rmul__', '__setattr__', '__setitem__', '__sizeof__',
'__str__', '__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index',
'insert', 'pop', 'remove', 'reverse', 'sort']
```

'append’, 'clear’ , 'copy', 'count’, 'extend', 'index', 'insert’, 'pop',
'remove’, 'reverse', 'sort’

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.append(x) – adaugă elementul **x** la sfârșitul listei **list**

Exemple

```
lista
[3, 7, 5, 2, 9, 6, 23]

lista.append(564)

lista
[3, 7, 5, 2, 9, 6, 23, 564]

lista.append([88,99])

lista
[3, 7, 5, 2, 9, 6, 23, 564, [88, 99]]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.extend(L) – extinde lista **list** cu elementele listei **L**

Exemple

```
lista
[3, 7, 5, 2, 9, 6, 23, 1]

L2 = [3,2,1]

lista.extend(L2)

lista
[3, 7, 5, 2, 9, 6, 23, 1, 3, 2, 1]
```

```
lista
[3, 7, 5, 2, 9, 6, 23, 564]

lista.extend([8,9,7,1])

lista
[3, 7, 5, 2, 9, 6, 23, 564, 8, 9, 7, 1]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.insert(i,x) – inserează **x** în poziția **i** a listei **list** sau la sfârșitul listei

Exemple

```
lista
[3, 7, 5, 2, 9, 6, 23, 1]

lista.insert(3,77)

lista
[3, 7, 5, 77, 2, 9, 6, 23, 1]
```

```
lista.insert(20,99)

lista
[3, 7, 5, 77, 2, 9, 6, 23, 1, 99]

lista.insert(len(lista), 88)

lista
[3, 7, 5, 77, 2, 9, 6, 23, 1, 99, 88]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.remove(x) – șterge prima apariție a lui **x. Eroare** dacă elementul nu se află în listă

```
lista
[3, 7, 5, 2, 9, 6, 23, 1]

lista.remove(5)

lista
[3, 7, 2, 9, 6, 23, 1]
```

```
In [153]: lista.remove(5)
Traceback (most recent call last):

  File "<ipython-input-153-a9caf582abd6>", line 1, in <module>
    lista.remove(5)

ValueError: list.remove(x): x not in list
```

```
lista
[3, 7, 2, 7, 9, 2, 9, 6, 23, 1]

lista.remove(7)

lista
[3, 2, 7, 9, 2, 9, 6, 23, 1]
```

Exemple

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.pop(i) – șterge elementul de pe poziția **i** sau ultimul element și totodată îl returnează

```
lista
[3, 2, 7, 9, 2, 9, 6, 23, 1]

lista.pop(4)
2

lista
[3, 2, 7, 9, 9, 6, 23, 1]

lista.pop()
1

lista
[3, 2, 7, 9, 9, 6, 23]
```

Exemple

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.clear() – elimină toate elementele din listă

Exemplu

```
lista
[3, 2, 7, 9, 9, 6, 23]

lista.clear()

lista
[]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.count(x) – returnează numărul de apariții ale unui element **x**

Exemple

```
lista
[3, 7, 5, 2, 9, 6, 23, 1, 2, 4, 3, 3]

lista.count(3)
3

lista.count(2)
2
 lista.count(0)
 0
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.index(x,[start[, end]]) – returnează indexul primei apariții a unui element **x** în intervalul **start - end**

Exemple

```
lista
[3, 7, 5, 2, 9, 6, 23, 1, 2, 6]

lista.index(5)
2

lista.index(2)
3

lista.index(2,4)
8
```

```
In [184]: lista.index(10)
Traceback (most recent call last):

  File "<ipython-input-184-cecb1ed8a
    lista.index(10)

ValueError: 10 is not in list
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.reverse() – inversează lista **list**

Exemplu

```
lista
[3, 7, 5, 2, 9, 6, 23, 1, 2, 6]

lista.reverse()

lista
[6, 2, 1, 23, 6, 9, 2, 5, 7, 3]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.copy() – crează o copie a listei **list**

```
lista
[6, 2, 1, 23, 6, 9, 2, 5, 7, 3]

l2 = lista.copy()

l2
[6, 2, 1, 23, 6, 9, 2, 5, 7, 3]

l2.append(99)

l2
[6, 2, 1, 23, 6, 9, 2, 5, 7, 3, 99]

lista
[6, 2, 1, 23, 6, 9, 2, 5, 7, 3]
```

Exemple

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele aplicabile listelor

list.sort() – sortează elementele listei **list**

Exemple

```
lista
[6, 2, 1, 23, 6, 9, 2, 5, 7, 3]

lista.sort()

lista
[1, 2, 2, 3, 5, 6, 6, 7, 9, 23]
```

```
l2
[6, 2, 1, 23, 6, 9, 2, 5, 7, 3, 99]

l2.sort(reverse=True)

l2
[99, 23, 9, 7, 6, 6, 5, 3, 2, 2, 1]
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Metodele split() , join()

Funcții cu scopuri opuse care funcționează pe șiruri de caractere, dar care se referă și la liste.

split() – returnează o listă dintr-un șir și un separator definit (implicit este un spațiu)

join() – returnează un șir dintr-o listă cu elementele listei împărțite de un separator dat.

```
L1 = ['unu', 'doi', 'trei', 'cinci','sapte']

s1 = ' : '.join(L1)

s1
'unu : doi : trei : cinci : sapte'
```

Exemple

```
s1
'unu : doi : trei : cinci : sapte'

l1 = s1.split(' : ')

l1
['unu', 'doi', 'trei', 'cinci', 'sapte']
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Parcurgerea listelor

```
for VARIABLE in LIST:
    BODY
```

```
i = 0
while i < len(LIST):
    VARIABLE = LIST[i]
    BODY
    i += 1
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Parcurgerea listelor

Exemplu

```
numere = [34,56,23,11,90]
i=0
lungimea = len(numere)
while i < lungimea:
    print(numere[i])
    i = i+1
```

Rezultat

```
34
56
23
11
90
```

Exemplu

```
numere = [25,56,23,18,95]
for i in numere:
    print(i)
```

Rezultat

```
25
56
23
18
95
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Parcurgerea listelor

Exemplu

```
L2 = ['trei', 8, 5.6, 31, ['sapun', 1, 'abac'], -6 ]
for i in range(len(L2)):
    print(i, ' ',L2[i])
```

Rezultat

```
0   trei
1   8
2   5.6
3   31
4   ['sapun', 1, 'abac']
5   -6
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Parcurgerea listelor folosind funcția enumerate()

Folosim funcția **enumerate()** când este necesar de manipulat simultan indecșii unei liste și elementele asociate.

Exemplu

```
L2 = ['trei', 8, 5.6, 31, ['sapun', 1, 'abac'], -6 ]
for i , elem in enumerate(L2):
    print(i, ' ', elem)
```

Rezultat

```
0   trei
1   8
2   5.6
3   31
4   ['sapun', 1, 'abac']
5   -6
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Exemple de probleme cu liste

```
#Felicitari prieteni
prieteni = ['Radu', 'Maria','Eduard', 'Ana']

for prieten in prieteni :
    print ('La multi ani:',  prieten)

for i in range(len(prieteni)) :
    prieten = prieteni[i]
    print ('La multi ani:',  prieten)
```

Rezultat

```
La multi ani: Radu
La multi ani: Maria
La multi ani: Eduard
La multi ani: Ana
La multi ani: Radu
La multi ani: Maria
La multi ani: Eduard
La multi ani: Ana
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Exemple de probleme cu liste

```
#Lungimea fiecarui element din lista
a = ["spam!",[1],["Brie","Roquefort","Camembert"],[1,2,3]]
for i in range(len(a)):
    print("ELEMENT", i,"LEN = ", len(a[i]))
```

Rezultat

```
ELEMENT 0 LEN =  5
ELEMENT 1 LEN =  1
ELEMENT 2 LEN =  3
ELEMENT 3 LEN =  3
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Exemple de probleme cu liste

```
# Suma produselor elementelor a doua liste
a=[1,2,5,6]
b=[8,5,4,9]
c=[]
sum=0
num=len(a)
for i in range(0,num):
    c[i:i]= [ a[i]*b[i] ]
for i in range(0,num):
    sum = sum + c[i]
print(c, sum)
```

Rezultat

```
[8, 10, 20, 54] 92
```

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Propuneri pentru lucrul individual

1. Revizuiți conținutul cursului

2. Scrieți o funcție adunaElementeLista(a, b) care ia două liste de numere de aceeași lungime și returnează o nouă listă care conține sumele elementelor corespunzătoare din fiecare. Listele originale ar trebui să rămână neschimbate.

3. Se dă o listă de numere întregi. Aranjați ascendent numai: a) numerele pozitive; b) elementele cu poziția pară din listă.

4. Să se scrie un program care schimbă valorile între primul și ultimul element într-o listă creată cu funcția range.

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Mulțumesc de atenție!