Wyklad4

December 11, 2019

1 Zadanie domowe 9

Sprawdź dokumentację funkcji chain, a następnie napisz program, który obniży stopień zagnieżdżenia listy. Np. [[2, 5], [1, 2], [4, 4]] zamieni na [2, 5, 1, 2, 4, 4].

[1]:
```
lista_1=[[2,5],[1,2],[4,4]]
```

[2]:
```
import itertools
```

[ ]:
```
help(itertools.chain)
```

[4]:
```
list(itertools.chain(lista_1))
```

[4]:
```
[[2, 5], [1, 2], [4, 4]]
```

[5]:
```
list(itertools.chain([2,5],[1,2],[4,4]))
```

[5]:
```
[2, 5, 1, 2, 4, 4]
```

[6]:
```
list(itertools.chain(lista_1[0],lista_1[1],lista_1[2]))
```

[6]:
```
[2, 5, 1, 2, 4, 4]
```

[7]:
```
list(itertools.chain(*lista_1))
```

[7]:
```
[2, 5, 1, 2, 4, 4]
```

[8]:
```
list(itertools.chain.from_iterable(lista_1))
```

[8]:
```
[2, 5, 1, 2, 4, 4]
```

Dygresje

[9]:
```
list(itertools.chain([[2,5]],[1,2],[4,4]))
```

[9]:
```
[[2, 5], 1, 2, 4, 4]
```

[10]:
```
print(*lista_1[0])
```

```
2 5
```

2 Zadanie domowe 7

Napisz program, który uporządkuje rosnąco listę ze względu na jej drugi element. Np. [(2, 5), (1, 2), (4, 4)] zamieni na [(1, 2), (4, 4), (2, 5)].

[11]:
```
lista_2=[(2, 5), (1, 2), (4, 4)]
```

[12]:
```
def funkcja_Second(elem):
    return elem[1]
```

[13]:
```
funkcja_Second(lista_2[0])
```

[13]: 5

[14]:
```
lista_2.sort(key = funkcja_Second)
```

[15]:
```
print(lista_2)
```

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

[16]:
```
lista_3=[(2, 5), (1, 2), (4, 4)]
```

[17]:
```
lista_3.sort(key = lambda elem: elem[1])
```

[18]:
```
print(lista_3)
```

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

3 Zmienne mutowalne i niemutowalne

[19]:
```
x=3
```

[20]:
```
id(x)
```

[20]: 10853696

[21]:
```
x=5;id(x)
```

[21]: 10853760

[22]:
```
y=[3];id(y)
```

[22]: 140311211046792

[23]:
```
y[0]=5;id(y)
```

[23]:
```
140311211046792
```

Konsekwencje

[24]:
```
y1=[1,2,3]
```

[25]:
```
y2=y1
```

[26]:
```
print(id(y1), id(y2))
```

```
140310801420040 140310801420040
```

[27]:
```
y2[0]=5
```

[28]:
```
print([y1,y2])
```

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

[29]:
```
y2=y1.copy()
```

[30]:
```
print(id(y1), id(y2))
```

```
140310801420040 140310801418056
```

[31]:
```
y2[0]=7
```

[32]:
```
print([y1,y2])
```

```
[[5, 2, 3], [7, 2, 3]]
```

Dalsze konsekwencje

[33]:
```
def zmiana_wartosci(x):
    print("wewnątrz funkcji (przed) id(x) =", id(x))
    x+=1
    print("wewnątrz funkcji (po) id(x)=", id(x))
```

[34]:
```
def zmiana_wartosci2(x):
    print("wewnątrz funkcji (przed) id(x) =", id(x))
    x[0]+=1
    print("wewnątrz funkcji (po) id(x)=", id(x))
```

[35]:
```
x=0;id(x)
```

[35]:
```
10853600
```

[36]:
```
zmiana_wartosci(x)
```

```
wewnątrz funkcji (przed) id(x) = 10853600
wewnątrz funkcji (po) id(x)= 10853632
```

[37]:
```
print("x=",x,"id(x)=",id(x))
```

```
x= 0 id(x)= 10853600
```

[38]:
```
x=[0];id(x)
```

[38]: 140311146056072

[39]:
```
zmiana_wartosci2(x)
```

```
wewnątrz funkcji (przed) id(x) = 140311146056072
wewnątrz funkcji (po) id(x)= 140311146056072
```

[40]:
```
print("x=",x,"id(x)=",id(x))
```

```
x= [1] id(x)= 140311146056072
```

4 Nieznana liczba parametrów

[41]:
```
def suma(x,*param):
    s=x
    for elem in param:
        s+=elem
    return s
```

[42]:
```
suma(0)
```

[42]: 0

[43]:
```
suma(0,1)
```

[43]: 1

[44]:
```
suma(0,1,2)
```

[44]: 3

[46]:
```
suma(0,*range(1,16))
```

[46]: 120

[ ]:
```
0+1+2+3+4+5
```

[47]:
```
def funkcja(arg1,arg2,*args):
    print("arg1=",arg1,"arg2=",arg2)
    print("*args=",args)
```

[48]:
```
funkcja("a","b","c","d","e",2,3)
```

```
arg1= a arg2= b
*args= ('c', 'd', 'e', 2, 3)
```

5 Funkcja annimowa - lambda

[49]:
```
Lista = [(lambda x: x**2), (lambda x: x**3), (lambda x: x**4)]
```

[50]:
```
for f in Lista:
    print(f(2),end=",")
```

```
4,8,16,
```

[51]:
```
Lista[0](3)
```

[51]: 9

[52]:
```
def f1(x): return x ** 2
def f2(x): return x ** 3
def f3(x): return x ** 4
```

[53]:
```
Lista_def = [f1, f2, f3]
```

[54]:
```
for f in Lista_def:
    print(f(2),end=",")
```

```
4,8,16,
```

[55]:
```
Lista_def[0](3)
```

[55]: 9

[57]:
```
suma=(lambda x: (lambda y: 2*x + 3*y))
```

[58]:
```
suma(1)(2)
```

[58]: 8

[59]:
```
suma(1)(0)
```

[59]: 2

[60]:
```
suma(0)(2)
```

[60]: 6

[61]:
```
suma(1,2)
```

```
␣
↪---------------------------------------------------------------------------

TypeError                                 Traceback (most recent call last)

<ipython-input-61-be5e6a409906> in <module>
----> 1 suma(1,2)


TypeError: <lambda>() takes 1 positional argument but 2 were given
```

[62]:
```
f = lambda a=1, b=2, c=3: ([a+b+c], [a-b-c])
```

[63]:
```
f()
```

[63]: ([6], [-4])

[ ]:
```

```