Lambda/Map/Filter/Zip/Reduce

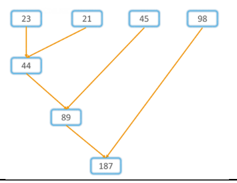
```
# lambda x,y,z, ...: function of(x,y,z,...)

double = lambda x: x*2
print("double: ", double(4))

8

# example
y = lambda m,x,c: m*x + c
print("y=mx+b: ", y(1,2,3))

# example
convert_temp = {'f2k': lambda deg_f: 273.15 + (deg_f - 32) * 5/9,
                'c2k': lambda deg_c: 273.15 + deg_c }

print("f2k: ", convert_temp['f2k'](32))
print("c2k: ", convert_temp['c2k'](50))

f2k:  273.15
c2k:  323.15
```

```
# map(func, *iterable), it transforms lije map in perl. map can
take 3 arguments also

l1 = ['a1', 'b1', 'c1']
upper1 = []

for i in l1:
    upper1.append(i.upper())
print("upper1: ", upper1)


upper2 = list(map(lambda i: i.upper(), l1))
print("upper2: ", upper2)

# example

# zip, implement zip with map

l1 = ['a', 'b', 'c', 'd', 'e', 'f']
n1 = [1, 2, 3, 4]

res1 = list(zip(l1,n1))
print("zip result: ", res1)

res2 = list(map(lambda x, y: (x,y), l1, n1))
print("zip using map: ", res2)
```

```
# example

tup = (5, 7, 22, 97, 54, 62, 77, 23, 73, 61)
newtuple = tuple(map(lambda x: x+3 , tup))
print(newtuple)

O/P:
(8, 10, 25, 100, 57, 65, 80, 26, 76, 64)
```

```
import re

row = """172.16.0.3 - - [25/Sep/2002:14:04:19 +0200] "GET / HTTP/1.1"
401 - "" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.1)
Gecko/20020827" """

print(list(map(''.join, re.findall(r'\"(.*?)\"|\[(.*?)\]|(\S+)', row))))

#['172.16.0.3', '-', '-', '25/Sep/2002:14:04:19 +0200', 'GET /
HTTP/1.1', '401', '-', '', 'Mozilla/5.0 (X11; U; Linux i686; en-US;
rv:1.1) Gecko/20020827']
```

```
# reduce(func, iterable[, initial])

from functools import reduce
reduce(lambda a,b: a+b,[23,21,45,98])
O/P: 187
```



```
from functools import reduce

numbers = [3,4,6,9,34,12]
def custom_sum(first, second):
    return first+second

result = reduce(custom_sum, numbers)
print("Result is: ", result)

Result is:  68
```

```
# return Palindrome words from a list of words

words = ['demigod', 'rewire', 'madam']
p_words = list(filter(lambda word: word == word[::-1], words))

print("p_words: ", p_words)

O/P:
p_words:  []
```

```
# I have a list of circle areas with 5 decimal precision. Round each
element in the list up to its decimal position places. 1st element ->
decimal 1, 2nd -> decimal 2 ..

ca = [ 3.56773, 5.57668, 4.00914, 5.77213, 6.11932]
result = list(map(round, ca, range(1,7)))

print("circle areas: ", result)

O/P: circle areas:  [3.6, 5.58, 4.009, 5.7721, 6.11932]

# explanation: 1st elm: round(3.56773,1) -> 2nd elm: round(5.57668,2)
```

```
# example

l1=[[1, 3, 5], [7, 9], [11, 13, 15]]
from functools import reduce
reduce(list.__add__, [[1, 3, 5], [7, 9], [11, 13, 15]], [])

O/P:
[1, 3, 5, 7, 9, 11, 13, 15]
```

```
# example

num = [[5, 7, 8, 10, 3], [5, 12, 45, 8, 9], [8, 39, 90,
5, 12]]
res = reduce(set.intersection, map(set, num))
print(res)

O/P:
{8, 5}
```