Exercise 10

May 16, 2022

1 1

Buat fungsi mencari jumlah bilangan genap dari list L.

Contoh:

L = [2,1,9,10,3,90,15] -> 3

[7]:
```
import functools as ft
L = [2,1,9,10,3,90,15]

# function count the even numbers from a list L using reduce without for
def even_count(L):
    return ft.reduce(lambda x,y: x+1 if y%2==0 else x, L, 0)

print(even_count(L))
```

```
3
```

2 2

Buat fungsi untuk menghitung n! menggunakan reduce

[15]:
```
facto = lambda n: ft.reduce( lambda a, b: a*b if b > 1 else 1, range(1,n+1), 1)
print(facto(5))
```

```
120
```

[18]:
```
n = 10
for i in range(0, n+1):
    print( str(i) + '! = ' + str(facto(i)) )
```

```
0! = 1
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
```

```
7! = 5040
8! = 40320
9! = 362880
10! = 3628800
```

3 3

Hitung euclidian distance dari dua vektor berikut menggunakan higher order function!

X = [2,5,6,7,10]

Y = [-2,9,2,-1,10]

[23]:
```
X = [2,5,6,7,10]
Y = [-2,9,2,-1,10]

euclid = lambda X, Y: ft.reduce( lambda a,c: a + c, map(lambda x,y: (x-y)**2,␣
 ↪X, Y) )**0.5
print(euclid(X,Y))
```

```
10.583005244258363
```

3.1 4

```
employee = {
    'Nagao':35,
    'Ishii':30,
    'Kazutomo':20,
    'Saito':25,
    'Hidemi':29
}
```

Terdapat dictionary employee berisi nama dan umur pegawai, lakukan filter untuk mengetahui pegawai yang berumur > 25 tahun!

[27]:
```
employee = {
    'Nagao':35,
    'Ishii':30,
    'Kazutomo':20,
    'Saito':25,
    'Hidemi':29
}

cnt_emp = lambda lim, employee: ft.reduce(lambda x,y: x+1 if y[1]> lim else x,␣
 ↪employee.items(), 0)
cnt_emp(25, employee)
```

[27]:
```
3
```

4 5

Buatlah deret fibonacci menggunakan higher order function!

[32]:
```
fibo = lambda n: ft.reduce( lambda a, b: a if b[0] <= 1 else a + [ a[ b[0]-1 ]␣
↪+ a[ b[0]-2 ] ] ,
                    enumerate( [0,1] + list(range(1, n))) , [0,1] ) if n > 0␣
↪else [0]
```

[34]:
```
for i in range(10):
    print('Fibonacci of ' + str(i) + ' = ' + str(fibo(i)))
```

```
Fibonacci of 0 = [0]
Fibonacci of 1 = [0, 1]
Fibonacci of 2 = [0, 1, 1]
Fibonacci of 3 = [0, 1, 1, 2]
Fibonacci of 4 = [0, 1, 1, 2, 3]
Fibonacci of 5 = [0, 1, 1, 2, 3, 5]
Fibonacci of 6 = [0, 1, 1, 2, 3, 5, 8]
Fibonacci of 7 = [0, 1, 1, 2, 3, 5, 8, 13]
Fibonacci of 8 = [0, 1, 1, 2, 3, 5, 8, 13, 21]
Fibonacci of 9 = [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

[36]:
```
# Recursive fibonacci using lambda
fibo_rec = lambda n: 0 if n == 0 else 1 if (n == 1 or n == 2) else␣
↪fibo_rec(n-1) + fibo_rec(n-2)
deret_fibo = lambda n: list( map( lambda x: fibo_rec(x), range(n+1) ) )
deret_fibo(10)
```

[36]:
```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
```