Exercise 13

May 16, 2022

1 1

Addku = lambda x: x + 10

Powku = lambda x: x**2

Kurku = lambda x: x - 2 * x

A. Buatlah fungsi komposisi menggunakan 3 fungsi diatas yang melakukan hal sebagai berikut secara berurut: 1. Menjumlahkan input dengan nilai 10 2. Mengurangi input dengan 2 kali input nya 3. Mengeluarkan nilai kuadrat dari input nya

B. Buatlah fungsi invers nya!

[1]:
```
addku = lambda x: x + 10
powku = lambda x: x**2
kurku = lambda x: x - 2 * x

f_komp = lambda f,g: lambda x: f(g(x))

my_f_kom = f_komp(kurku, f_komp(powku, addku))

my_f_kom(10)
```

[1]:
```
-400
```

[3]:
```
# invers
inv_addku = lambda x: x - 10
inv_powku = lambda x: x**0.5
inv_kurku = lambda x: -1 * x

my_f_kom_inv = f_komp(inv_addku, f_komp(inv_powku, inv_kurku))
my_f_kom_inv(-400)
```

[3]:
```
10.0
```

2 2

2.1 IPK

[9]:
```
from functools import reduce as r

# Define function composition
mycompose = lambda *funcs: r( lambda f, g: lambda x: f(g(x)), reversed(funcs),␣
↪lambda x:x )
```

[10]:
```
# Ketentuan jumlah tanggungan
def skor1(jtg):
    return 1 if jtg >= 5 else 5-jtg
```

[11]:
```
# Ketentuan token listrik
def skor2(X):
    def rata(X):
        return sum(X)/len(X)

    def l_cond_1(X):
        return [X, [X>100000] ]

    def l_cond_2(X):
        return [X[0], X[1] + [ X[0] >= 50000 ] ]

    def to_score2(X):
        return r( lambda a,b: a+ (1 if b == True else 0), X[1], 1)

    compose_cond = mycompose(rata, l_cond_1, l_cond_2, to_score2)
    return compose_cond(X)

# skor2([50000, 50000, 50000])
```

[12]:
```
# Ketentuan gaji

def con_1(X):
    return [X[0], 1, X[2], [ X[0] > X[2][X[1]] ] ]

def con_2_to_n(X):
    return [X[0], X[1]+1, X[2], X[3] + [ X[0] > X[2][X[1]] ] ]

def to_score(X):
    return r( lambda a,b: a+ (1 if b == True else 0), X[-1], 2)

def prep(gj):
    return [gj, 0, list(map( lambda x: x*1000000, list(range(10,3,-1)) + [3]) )]

def skor3(gaji):
```

```
    commpy = mycompose(prep, con_1, *(con_2_to_n for i in range(4)), to_score)
    return commpy(gaji)
```

[13]:
```
# Ketentuan KIP K
def skor4(X=True):
    return 1 if X else 5
```

[14]:
```
def combineskor(X):
    return X + [map( lambda f,x: f(x), X[1], X[0] )]

def boboti(X):
    return r( lambda a,b: a+b, map(lambda x,y: x*y, X[-1], [0.2, 0.3, 0.2, 0.
 ↪3]) )

def toUKT(X):
    return 750000 + X*500000
```

[15]:
```
mhs = [3,
    [120000, 75000, 50000],
    5.5 * 10**6,
    False
]

datas = [mhs, [skor1, skor2, skor3, skor4] ]
compose_fin = mycompose(combineskor, boboti, toUKT)
compose_fin(datas)
```

[15]:
```
2200000.0
```

3 3

3.1 Turunan polinom

dat = '-3x^5 + 2x^2 -4x +5'

output -> '-15.0x^4 + 4.0x - 4.0'

[16]:
```
# Turnan polinom

def split(dat):
    return dat.replace(' ', '').replace('-','+-').split('+')

def chdepan(dat):
    return dat[1:] if dat[0] == '' else dat

def eqkan(dat):
```

```
    return map( lambda x: x if '^' in x else x+ '^1' if 'x' in x else x+ 'x^0',␣
 ↪dat)

def toarr2d(dat):
    return r( lambda a, b: a + [[float(hurf) for hurf in b.split('x^')]] , dat,␣
 ↪[])

def sortdesc(dat):
    return sorted(dat, key=lambda x: x[1], reverse=True)

def calctur(dat):
    return map( lambda x: [0,0] if x[1] == 0 else [x[1]*x[0], x[1]-1], dat)

def tostr(dat):
    return map( lambda x: '0' if x[0] == 0 else str(x[0]) if x[1]==0 else␣
 ↪str(x[0]) + 'x^' + str(x[1]), dat)

def prettykan(dat):
    return r( lambda a,b: a+'+' + b if b != '0' else a, dat, '')

def prettysign(dat):
    return dat.replace('+-', ' -').replace('+', '+ ')
```

[20]:
```
dat = '-3x^5 + 2x^2 -4x +5'
fss = (split, chdepan, eqkan, toarr2d, sortdesc, calctur, tostr, prettykan,␣
 ↪prettysign)
my_turunan = mycompose(*fss)
my_turunan(dat)
```

[20]:
```
' -15.0x^4.0+ 4.0x^1.0 -4.0'
```

4 4

Buatlah fungsi untuk menghitung biaya yang harus dibayar customer pada suatu e-commerce menggunakan higher order function. Buatlah decorator untuk mengeluarkan harga sebelum pajak dan sesudah pajak (pajak = 11%) ! Gunakan decorator untuk menambahkan perhitungan waktu eksekusi!

[4]:
```
from functools import reduce as r
keranjang = [
    {'Jumlah_Barang': 5, 'Harga:': 10 },
    {'Jumlah_Barang': 7, 'Harga:': 20 },
    {'Jumlah_Barang': 20, 'Harga:': 4.5 }
]

def pajak_decorator(func):
```

```
    def inner(*args, **kwargs):
        res = func(*args, **kwargs)
        print('Sub Total: ', res)
        print('Pajak: ', res * 0.01)
        print('Total: ', res + res * 0.01)
        return res
    return inner

import time

def calc_time_decorator(func):
    def inner(*args, **kwargs):
        start = time.time()
        res = func(*args, **kwargs)
        end = time.time()
        print('Time: ', end - start)
        return res
    return inner
```

[6]:
```
@calc_time_decorator
@pajak_decorator
def hitung_pembayaran_1(keranjang):
    return r( lambda a,b: a + (b['Jumlah_Barang'] * b['Harga:']), keranjang, 0)␣
 ↪* 1000

hitung_pembayaran_1(keranjang)
```

```
Sub Total:  280000.0
Pajak:  2800.0
Total:  282800.0
Time:  0.0009989738464355469
```

[6]: 280000.0