Nama = Sandy Wijaya

NIM = 120450047

LAPORAN PRAKTIKUM``

PERTEMUAN 7- HIGHER ORDER FUNCTION (MAP)

1. Given List p = ['a', 'k', 'u', 'l', 'u','p','a']

we want to make list of tuples of p like this

P’ = [(1, ’a’) , (3, ‘k’), (5,’u’), (7,’l’) , (9,’u’), (11,’p’) , (13,’a’)

```
P= 'akulupa'
print(*map(lambda p :(p[0]*2+1, p[1]), enumerate (P)))
```

```
(1, 'a') (3, 'k') (5, 'u') (7, 'l') (9, 'u') (11, 'p') (13, 'a')
```

2. Terdapat bilangan B

B = 24

Petakan B menjadi list faktor nya!

B’ = [1,2,3,4,6,8,12,24]`

```
B= 24
TWO =map(lambda b:b+1 if B% (b+1)== 0 else -1, range(B))
def dua(TWO):
  x=[]
  for i in TWO:
    if i != -1:
      x.append(i)
  return x
print(dua(TWO))
```

```
[1, 2, 3, 4, 6, 8, 12, 24]
```

3. Diketahui matriks A,B,C sebagai berikut

A = [[3 , 4] , [5, 6]]

B = [[1,2] , [7 , 8]]

C = AB

Buatlah program untuk menghitung determinan matriks C menggunakan HOF map!

```
A=[[3,4],[5,6]]
B=[[1,2],[7,8]]
C= list(map(lambda ra, rb:list(map(lambda raa, rbb: raa+rbb, ra, rb)),A,B))
def dett(C):
  return C[0][0]*C[1][1]-C[0][1]*C[1][0]
print(dett(C))
```

```
-16
```

PERTEMUAN 9- HIGHER ORDER FUNTION (FILTER)

1. Buat program untuk menghitung deret bilangan prima dari 2 hingga N menggunakan HOF filter dan map

Contoh primes(100):
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97

```
FAKTOR = lambda n: list(filter(lambda i:n%i==0, range(1, n+1)))
primes = lambda n : filter(lambda i : len(FAKTOR(i))==2, range(1,n+1))
print(*primes(100))
```

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

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

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

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

```
    'Hidemi':29
}
print(employee.items())
```

```
dict_items([('Nagao', 35), ('Ishii', 30), ('Kazutomo', 20), ('Saito', 25), ('Hidemi',
```

```
filter_by_age=lambda age, employee : list(filter(lambda x:x [1]>=25, employee.items()))
print(*map(lambda d:d[0],filter_by_age(25, employee)))
```

```
Nagao Ishii Saito Hidemi
```

PERTEMUAN 10- HIGHER ORDER FUNCTION (REDUCE)

1. Buat fungsi mencari jumlah bilangan genap dari list L!

Contoh:

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

Output:

3

```
from functools import reduce as r
L=[2,1,9,10,3,90,15]
n_genap = lambda L:r(lambda a,b: a+1 if b%2 == 0 else a,L, 0)
n_genap(L)
```

```
3
```

2. Buat fungsi untuk menghitung n! Menggunakan reduce!

```
factor = lambda n:r(lambda a,b : a*b if b>1 else 1, range(1, n+1),1)
for i in range(10+1):
  print(str(i)+'!=',factor(i))
```

```
0!= 1
1!= 1
2!= 2
3!= 6
4!= 24
5!= 120
6!= 720
7!= 5040
8!= 40320
9!= 362880
10!= 3628800
```

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

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

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

```
x= [2,5,6,7,10]
y= [-2,9,2,-1,10]

euclid = lambda x,y: r(lambda a,c: a+c, map(lambda x,y: (x-y)**2, x,y))**0.5
euclid(x,y)
```

```
10.583005244258363
```

4. Terdapat dictionary employee berisi nama dan umur pegawai, lakukan reduce untuk mengetahui berapa jumlah pegawai yang berumur > 25 tahun !

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

```
employee = {
    'Nagao':35,
    'Ishii':30,
    'Kazutomo':20,
    'Saito':25,
    'Hidemi':29
}
cnt_emp = lambda lim,employee :r(lambda a,b : a+1 if b[1]>lim else a, employee.items(),0)
cnt_emp(25, employee)
```

```
3
```

5. Buatlah deret fibonacci menggunakan higher order function!

```
fibo = lambda n:r(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]
```

```
for i in range(10):
```

```
  print('fibonacci',i,'->',fibo(i))
```

```
    fibonacci 0 -> [0]
    fibonacci 1 -> [0, 1]
    fibonacci 2 -> [0, 1, 1]
    fibonacci 3 -> [0, 1, 1, 2]
    fibonacci 4 -> [0, 1, 1, 2, 3]
    fibonacci 5 -> [0, 1, 1, 2, 3, 5]
    fibonacci 6 -> [0, 1, 1, 2, 3, 5, 8]
    fibonacci 7 -> [0, 1, 1, 2, 3, 5, 8, 13]
    fibonacci 8 -> [0, 1, 1, 2, 3, 5, 8, 13, 21]
    fibonacci 9 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

PERTEMUAN 11- RECUSION IN FP

Buat sebuah program untuk membuat deret fibonacci dari 0 hingga N dengan menggunakan fungsi non-rekursif dan rekursif!

Bandingkan keduanya jika nilai N = 500, Manakah yang lebih baik? Jelaskan!

```
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_rec = lambda n : list(map(lambda i : fibo_rec(i), range(n+1)))
```

```
deret_fibo_rec(10)
```

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

PERTEMUAN 12- PURITY AND IMMUTABILITY

Latihan 1

Ubah fungsiku menjadi pure function!

```
def fungsiku(L):
  def check_genap(l):
    return l % 2 == 0
  for i in range(len(L)):
    if check_genap(L[i]):
       L[i]= L[i]/2
    else:
      L[i]=L[i]*n+1
  return L
```

```
n = 3
```

```
L =[5,6,7,8]
print(fungsiku(L))
```

```
     [16, 3.0, 22, 4.0]
```

```
print(L)
```

```
     [16, 3.0, 22, 4.0]
```

```
n = 3
L =[5,6,7,8]
def fungsiku(L,n):
  return list(map(lambda x:x/2 if x%2 == 0 else x*n+1, L))
print(fungsiku(L,n))
```

```
     [16, 3.0, 22, 4.0]
```

```
print(L)
```

```
     [5, 6, 7, 8]
```

Latihan 2

Ubah fungsiku2 menjadi pure function!

```
def fungsiku2(L):
  def check_faktor(l):
    return l % n == 0
  for i in range(len(L)):
    if check_faktor(L[i]):
      L[i]= L[i]/2
    else:
      L[i]=L[i]*n+1
  return L
```

```
n = 3
L = [5,6,7,8]
print(list(fungsiku2(L)))
```

```
     [16, 3.0, 22, 25]
```

```
print(L)
```

```
     [16, 3.0, 22, 25]
```

```
n = 3
L = [5,6,7,8]
def fungsiku2(L,n):
  return list(map(lambda x: x/2 if x%n == 0 else x*n+1, L))
print(fungsiku2(L,n))
```

```
[16, 3.0, 22, 25]
```

```
print(L)
```

```
[5, 6, 7, 8]
```

Apakah isi dalam tupel tup ada yang dapat diubah?

```
tup = ([3, 4, 5], 'myname')
```

```
tup1 = ([3,4,5],'myname')
tup1[0]= "sandy"
print('tup 2 : ',tup)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-37-8fbb6e4386b7> in <module>()
      1 tup1 = ([3,4,5],'myname')
----> 2 tup1[0]= "sandy"
      3 print('tup 2 : ',tup)

TypeError: 'tuple' object does not support item assignment
```

SEARCH STACK OVERFLOW

PERTEMUAN 13- FUNCTION BUILDING FUNCTION

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!

```
keranjang = [
{'Jumlah_barang' :5, 'Harga':10}
{'Jumlah_barang' :7, 'Harga':20}
{'Jumlah_barang' :20, 'Harga':4.5}
]
```

```
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 * 0.11)
    print('total : ', res + res * 0.11)
    return res
  return inner

import time
def calc_time_decorator(fu):
  def inner(*args, **kwargs):
    waktu_awal = time.time()
    res = fu(*args, **kwargs)
    waktu_akhir = time.time()
    print('Waktu eksekusi : ', waktu_akhir - waktu_awal)
    return res
  return inner


@calc_time_decorator
@pajak_decorator
def hitung_pembayaran_1(keranjang):
  return r(lambda a,b : a + (b['Jumlah_barang'] * b['Harga']),keranjang,0)
hitung_pembayaran_1(keranjang)
```

```
sub total :  280.0
pajak :  30.8
total :  310.8
Waktu eksekusi :  0.0031380653381347656
280.0
```

```
@calc_time_decorator
@pajak_decorator
def hitung_pembayaran_2(keranjang):
  s= 0
  for k in keranjang:
    s=s+k['Jumlah_barang']*k['Harga']
  return s*1000
hitung_pembayaran_2(keranjang)
```

```
sub total :  280000.0
pajak :  30800.0
total :  310800.0
Waktu eksekusi :  0.00032067298889160156
280000.0
```

LATIHAN UAS

```
import numpy as np
from random import randint
from functools import reduce as r


def fungsi_W(a,b):
  W = np.random.uniform(a,b,size=(3,2))
  return W
W =fungsi_W(-1,1)
W

     array([[ 0.96385914, -0.30166764],
            [ 0.68821209, -0.49218577],
            [ 0.7354844 , -0.49704018]])


def fungsi_M(a,b):
  M= np.random.uniform(a,b,size=(2,2))
  return M
M=fungsi_M(-1,1)
M

     array([[-0.34724454,  0.05519374],
            [ 0.07093356, -0.58435506]])


def fungsi_H(a,b):
  H= np.random.uniform(a, b, size=(2, 1))
  return H
H = fungsi_H(-1,1)
H

     array([[-0.84104   ],
            [-0.72503209]])


X=[0,3,4]


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


import math
def aktivasi(x):
  return (math.exp(x)-math.exp(-x))/(math.exp(x)+ math.exp(-x))
def WTi(W,i):
  return list(map(lambda w: w [i], W))
def WT(W):
```

```
  return list(map(lambda i : WTi(W,i),range(0,len(W[0]))))
def XW(X,W):
  return map(lambda w:r (lambda x,y : x+y, map(lambda xx,ww : xx*ww, X, w)),WT(W))

def input_to_hidden(X,W):
  return map(lambda x: aktivasi(x), XW(X,W))

A = list(input_to_hidden(X,W))

def MTi(M,i):
  return list(map(lambda m: m[i], M))
def MT(M):
  return list(map(lambda i : MTi(M,i), range(0,len(M[0]))))
def AM(A,M):
  return map(lambda m:r(lambda x,y : x+y, map(lambda aa,mm : aa*mm, A,m)),MT(M))
def hidden_to_hidden(A,M):
  return map(lambda a:a, AM(A,M))

@calc_time_decorator
def feed_forward(A,M,H):
  return hidden_to_hidden(hidden_to_hidden(A,M),H)


print(list(feed_forward(A,M,H)))
```

```
     Time :  5.221366882324219e-05
     [-0.11129975037578121]
```

JURNAL MODUL 1

Seorang mahasiswa sains data ingin menyewa buku dari sebuah startup yang menyediakan layanan sewa
tersebut memiliki ketentuan sewa dengan aturan sebagai berikut:
a. Harga sewa buku berbeda-beda sesuai dengan kategorinya
b. Harga sewa buku dihitung berdasarkan jumlah halaman nya
c. Harga sewa buku dihitung per hari nya
d. Maksimal durasi sewa adalah 26 hari
Startup tersebut masih dalam tahap awal pengembangan, sehingga ingin melakukan uji coba penyewaan
Berikut rincian kategori nya:
- Kategori 1 : 100 rupiah per lembar per hari
- Kategori 2 : 200 rupiah per lembar per hari
- Kategori 3 : 250 rupiah per lembar per hari
- Kategori 4 : 300 rupiah per lembar per hari
- Kategori 5 : 500 rupiah per lembar per hari
Startup tersebut memerlukan sebuah program untuk:
• menghitung total biaya dari customer
• mencatat tanggal awal sewa, dan durasi hari
• menampilkan informasi kapan tanggal pengembalian buku dari customer
Format input tanggal adalah yyyy-mm-dd
Bantulah startup tersebut membuat program tersebut dengan menggunakan konsep modularisasi!

```
tanggal = input('Tanggal Pinjam :') # 2020-01-02
durasi = int(input('Durasi Pinjam :')) # 25
kategoris = {
 1 : 100,
 2 : 200,
 3 : 250,
 4 : 300,
 5 : 500
}
```

```
     Tanggal Pinjam :2020-01-02
     Durasi Pinjam :25
```

```
def dtl(s_tgl):
 return [int(k) for k in s_tgl.split('-')]
def is_cm(tgl_p,d,c):
 return tgl_p[2] + 2 > c
def thn_back(tgl_p,d,c):
 return tgl_p[0]+1 if (is_cm(tgl_p,d,c) and tgl_p[1] == 12) else tgl_p[0]
def bln_back(tgl_p,d,c):
 return(tgl_p[1] % 12)+1 if is_cm(tgl_p,d,c) else tgl_p[1]
def tgl_back(tgl_p,d,c):
 return tgl_p[2] + d - c if is_cm(tgl_p,d,c) else tgl_p[2] + d
def is_awal_abad(thn):
 return thn % 100 == 0
def kabisat(thn):
 return(is_awal_abad(thn) and thn % 400 == 0) or (not is_awal_abad(thn) and thn % 4 == 0)
def dec_c(t):
 return 30 + (t[1]%2 if t[1] <= 8 else abs ((t[1] % 2) - 1)) if t[1]!= 2 else (29 if kabis
def wkt_kembali(tgl_p,d):
 return [ thn_back(tgl_p,d, dec_c(tgl_p)), bln_back(tgl_p,d,dec_c(tgl_p)), tgl_back(tgl_p,
```

```
# tanggal = '2022-2-27'
# durasi = 1
tgl_p = dtl(tanggal)
wkt_kembali(tgl_p,durasi)
```

```
     [2020, 1, 27]
```

```
sewaan_all = [ [1,5], [2,3], [3,0], [4,1], [5,2]]
def calc_biaya_per_kategori(kategoris,sewaan):
 return sewaan[1] * kategoris.get( sewaan[0] )
def calc_all_biaya(kategoris,sewaan_all,durasi):
 return sum( [calc_biaya_per_kategori(kategoris,sewaan) for sewaan in sewaan_all]) * duras
```

```
calc_all_biaya(kategoris,sewaan_all,durasi)
```

```
     60000
```

JURNAL MODUL 2

Kerjakan seluruh soal berikut dengan menggunakan higher order function map,filter dan reduce!

1. Buatlah sebuah fungsi bernama ulangi_NIM, ulangi memiliki input sebuah bilangan skalar a, dan r

1xn dengan seluruh elemen nya adalah a !

```
def ulangi_120450047(a): # fungsi ulangi_NIM memiliki input parameter/argumen yaitu variab
 return list(map(lambda x:a, range(a))) # Rumusnya adalah memanggil variabel a sebanyak 1
ulangi_120450047(5) # Output dari fungsi ulangi_NIM adalah map dengan menggunakan fungsi l
```

```
[5, 5, 5, 5, 5]
```

```
# Cara_2
a = 1 # Dideklarasikan variabel a = 1 (sebagai bilangan skalar)
n = 5 # Dideklarasikan banyaknya output pada variabel a (sebagai dimensi vektor 1 x n)
print(list(map(lambda x:a, range(n)))) # Output dari fungsi ulangi_NIM adalah map dengan m
```

```
[1, 1, 1, 1, 1]
```

`2. Buatlah deret bilangan sebagai berikut dengan input n sebagai panjang deret:

1/2, -1/4, ...,(-1)^n 1/ 2^n+1

```
n = 5 # inputan
deret = list(map(lambda x: ((-1)**(x)) * (1/(2**(x+1))), range(0,n))) # Fungsi map untuk i
# Rumus deretnya menggunakan fungsi lambda untuk menghitung deret bilangan pecahan diatas
print(deret)
```

```
[0.5, -0.25, 0.125, -0.0625, 0.03125]
```

```
b = range(1,n+1)
def pola_deret(x):
 return (((-1)**(x+1)) * (1/(2**x)))
```

3. Jumlahkan deret bilangan tersebut! (deret pada soal 2)

```
from functools import reduce # Import library functools.reduce(function, iterable, initial
print(reduce(lambda x,y: x+y, deret)) # Menyimpan nilai sebelumnya dan menambahkan nilai y
```

```
0.34375
```

4. Sebuah DNA dimodelkan dalam sebuah string menjadi sequence TCGA dan disimpan kedalam data :

https://drive.google.com/file/d/18C1ylsTXrY9pglqqlhijoS8LYmcxdIjM/view?usp=sharing

(https://drive.google.com/file/d/18C1ylsTXrY9pglqqlhijoS8LYmcxdIjM/view?usp=sharing)

hitunglah jumlah kemunculan pola berikut pada data tersebut:

a. A

b. AT

c. GGT

d. AAGC

e. AGCTA

```
data = open('DNA.txt','r') # Open source text.txt
data = data.read() # Membaca source text.txt
# print(data) # Output source text.txt


filename = 'DNA.txt' # File source text.txt
dat = open(filename,'r').read() # Open file source text.txt
dat = dat[:-1] # Batas
seq = 'ACT' # Mengurutkan ACT dibaris akhir batas data DNA
# data


def append_n(dat,i,n): # Fungsi append_n adalah menambahkan nilai array pada nilai akhir
 return reduce(lambda a,b: a+b, dat[i:i+n]) # Rumusnya a nilai sebelumnya data DNA, b nila
def remap(dat,seq): # Fungsi remap adalah memetakan kembali dari sequence yang baru
 return map(lambda x: append_n(dat,x,len(seq)), range(len(dat) - len(seq))) # Banyaknya ur
def count_mer(dat,seq): # Fungsi count_mer
 return reduce(lambda a,b: a + (1 if b==seq else 0), remap(dat,seq), 0) # Pada fungsi ke-2


# data[0:3]
# list(remap(dat,'ACT'))
# append_n(dat,0,3)

sequences = ['A','AT','GGT','AAGC','AGCTA'] # Terdapat list dari sequence yang akan kita c
def count_all(dat,sequences):
 return map(lambda x: count_mer(dat,x), sequences) # Melakukan perhitungan mer pada setiap
res = count_all(dat,sequences) # Memanggil semua jumlah sequence yang dicari
print(* res)
```

```
     2112 556 77 22 5
```

5. Reverse complement dari suatu sequence string DNA memiliki aturan sebagai berikut:

A adalah komplemen dari T

C adalah komplemen dari G

Contoh reverse complement:

input DNA : ACTGA

Reverse complmenet : TGACT

Buatlah fungsi untuk mencari inverse komplemen dari data pada nomor 4 !

```
def komplemen(x):
 return {'A':'T', 'T':'A', 'C':'G', 'G':'C'}.get(x) # Kamus(dictionary) pengubahan huruf d
def reverse_komplemen(dat):
 return map(lambda x:komplemen(x), dat) # Setiap data yang ada diubah sesuai dengan dictio


res = reverse_komplemen(dat)
print(*res)
```

```
A C A G A A G G C C G A C T C G C C A A G G A T T G G T C G T C T G A C T A T G A C C
```

6. Buatlah fungsi feed-forward dengan input berikut:

W11 = 0.5

W12 = 0.4

W21 = 0.3

W22 = 0.7

W31 = 0.25

W32 = 0.9

M11 = 0.34

M21 = 0.45

dan X1 = 9 , X2 = 10 , X3 = -4

```
import math # Merupakan module yang akan digunakan dalam case ini
def aktivasi(x):
 return 1/ (1+ math.exp(-x)) # x dijadikan bentuk yang bisa digunakan dalam case ini denga
def WTi(W,i):
 return list(map(lambda w:w[i], W)) # Index yang tercantum dalam W digunakan didalam list
def WT(W):
 return list(map(lambda i : WTi(W, i), range(len(W[0]))))
def XW(X,W):
 return map( lambda w: reduce (lambda a,b:a+b, map(lambda xx, ww: xx * ww, X, w), 0), WT(W
def input_to_hidden(X,W):
 return list(map(lambda x:aktivasi(x), XW(X,W))) # Menginput aktivasi tersembunyi
def feed_forward(X,W,M):
 return input_to_hidden(input_to_hidden(X,W), M) # Membuat fungsi feed forward


X = [9, 10, -4]
W = [[0.5, 0.4], [0.3, 0.7], [0.25, 0.9]]
M = [[0.34], [0.45]]
# Supaya pola sesuai dengan apa yang dibuat, bentuk W harus sama dengan M
feed_forward(X,W,M) # Output fungsi feed_forward
```

```
Time :  3.075599670410156e-05
<map at 0x7f824d4c6750>
```

```
 WTi(W,0)
```

```
    [0.5, 0.3, 0.25]
```

```
x
```

```
[9, 10, -4]
```

✓ 0s completed at 9:30 PM

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.