

LAPORAN AKHIR PRAKTIKUM PEMOGRAMAN BERBASIS FUNGSI

1	Nama	: Alfianri Manihuruk
2	NIM	: 120450088
3	Kelas	: RB
4	Matkul	: Pemograman Berbasis Fungsi

Jurnal Modul 1

1. Seorang mahasiswa sains data ingin menyewa buku dari sebuah startup yang menyediakan layanan sewa buku, Startup 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 5 kategori buku. 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!

In []:

```
1 tanggal = input('tanggal pinjam :')
2 durasi = int( input ('durasi pin: '))
3 kategoris ={
4     1:100,
5     2:200,
6     3:250,
7     4:300,
8     5: 500
9 }
```

In [25]:

```
1 def dtl(s_tgl):
2     return [int(k) for k in s_tgl.split('-')]
3
4 def is_cm(tgl_p, d, c):
5     return tgl_p[2] + d > c
6
7 def thn_back(tgl_p, d, c):
8     return tgl_p[0] + 1 if (is_cm(tgl_p, d, c) and tgl_p[1] == 12) else tgl_p[0]
9
10 def bln_back(tgl_p, d, c):
11     return (tgl_p[1] % 12) + 1 if is_cm(tgl_p, d, c) else tgl_p[1]
12
13 def tgl_back(tgl_p, d, c):
14     return tgl_p[2] + d - c if is_cm(tgl_p, d, c) else tgl_p[2] + d
15
16 def is_awal_abad(thn):
17     return thn % 100 == 0
18
19 def kabisat(thn):
20     return (is_awal_abad(thn) and thn % 400 == 0) or (not is_awal_abad(thn) and thn %
21
22 def dec_c(t):
23     return 30 + (t[1] % 2 if t[1] <= 8 else abs((t[1] % 2) - 1)) if t[1] != 2
24
25 def wkt_kembali(tgl_p, d):
26     return [thn_back(tgl_p, d, dec_c(tgl_p)), bln_back(tgl_p, d, dec_c(tgl_p)), tgl_back
```

In [30]:

```
1 #tanggal="2022-08-1"
2 #durasi=1
3
4 tgl_p = dtl(tanggal)
5 wkt_kembali(tgl_p, durasi)
```

Out[30]:

[2022, 8, 2]

In [27]:

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

In [28]:

```
1 calc_all_biaya(kategori, sewaan_all, durasi)
```

Out[28]:

14400

Jurnal Modul 2

1. Buatlah sebuah fungsi bernama `ulangi_NIM`, ulangi memiliki input sebuah bilangan skalar `a`, dan mengeluarkan vektor `1xn` dengan seluruh elemen nya adalah `a`

In [3]:

```
1 def ulangi_088(a): # membuat fungsi untuk jumlah atau panjang sklar
2     return list(map(lambda x: a, range(a)))# akan me return nilai si x dengan panjangnya
3 ulangi_088(5)
```

Out[3]:

[5, 5, 5, 5, 5]

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

$$\frac{1}{2}, \frac{-1}{4}, \dots, (-1)^n \frac{1}{2^{n+1}}$$

In [6]:

```
1 n = 5 # inputan
2 deret = list(map(lambda x: ((-1)**(x)) * (1/(2**(x+1))), range(0,n))) # Fungsi map untuk
3 # Rumus deretnya menggunakan fungsi lambda untuk menghitung deret bilangan pecahan diatas
4
5 print(deret)
```

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

In [7]:

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

3. Jumlahkan deret bilangan tersebut!

In [8]:

```
1 from functools import reduce # Import library functools.reduce(function, iterable, initial)
```

In [9]:

```
1 print(reduce(lambda x,y: x+y, deret)) # Menyimpan nilai sebelumnya dan menambahkan nilai
```

0.34375

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

In [10]:

```
1 filename= 'DNA.txt' # kita deklarasikan filenya menjadi sebuah variabel (filename)
2 dat = open(filename, 'r').read() #membuka file, fungsi dari (r) adalah membaca file ter
3 dat=dat[:-1] # melakukan slicing pada data sehingga memotong indexnya hanya 1
4 seq = 'ACT' # membuat variabel seq dengan isi string("ACT")
5 dat #menampilkan data
```

Out[10]:

```
'TGTCTTCCGGCTGAGCGGTTCTTAACCAGCAGACTGATACTGGTCTGAATATCGACGGGCAAGAGCCCTGGGATTG
ATGCGTTTTACCATGCGCTCTCAGTGCAGGCAGGAATGCAGAGCTTACTTCAAACCTAGTTACTGGCAAAAAATAC
AAATTTTTTCGATCGACCTTGAGTTTATTATTACCGCACAGTCTTTTACCGCACCTGTTACCGCACATCCGTAAG
TTTACCGCACGTTACCGCACTACCTCTCTATATTACCGCACTTCGTTTACCGCACGCTGAGGAACGGTTACCGCAC
TTACCGCACCACAAGGTGCGTGCTCTGTTATTACCGCACCACCATTACCGCACGCACTTTTATTACCGCACCAGGG
CACAGCCACGTAGGGTAGCGTCTGTTCTCACTGTATTGCGGCGACGGTCGTAATTTACCGCATTACCGCACCACCTCG
TTAGCTTACCGCACCTAGGGTTGTTACCGCACGACTTACCGCACAGCCGTTACCGCACGTGTTACTTGACGCTCTA
ACTCCCCTCATATCAGTCTTATTACCGCACACTGGGCTTACCGCACCCGACCTTAAGTAGGCAGTTACCGCACG
TATTACCGCACGTAATTACCGCACACCTGTAAAGGCAGGGTAAAGTACAGACTTACCGCTTACCGCACGGTTGCAC
CAGACAAATCTAACGTTAGGTACGTTACCGCACGGGAAATTACCGCACTCCAGGGTTTTACCGCACAGATATCCA
TTCGGGAATGTGACCCCTGGAGTGGAGTTGTGCGAAAGATACGGAGTTTTCAAGGGCACACCCAGCTATGTTATTA
AGCGTTACAGTGGCCGCTGCATCATGTCAATGTTCAAGTCTCTCTATCTTGCTATGTACGAACCCCTCGTTAAGA
GGGAGTAAGCGATCTTTTGACAAAATCGTATGCATGTAGGCGAGGCAATGCCGATTACATTGAACGGCGGGACTTT
TCGTATGAGACACCGCGGTTGAAATATTTTTTATGCAAGAGCGGGATTGGGCGGAAGGAGACTTAACGCAGTGCC
TAGCACTGTAACTGCGGCATGGCCGGATGGACTACCTATTTTGCAGCTCCAGCGTTTGAGTTCCACGTAAGTACG
GAACAGTCCCAGATAGGCCATGTGGTGCATCCAGTGAGAAATGAGACTCGAGATGCCGGTACCGGTAGCATCAC
CACATTGCTCCAGTATGATATCAGTCTTCACTGTGAGCAATTAATGCAGCGATCTTGAAGAGAGTTATTCTCTCT
TATCACCTGACAATAAATCAATTTACCAGTCAAATCTCTTTAACATCGTGCCGAAGTGCATGCGTCTGATGCTA
GATTAGGATATATTTTCTTAGCTGGCTTCGATGATTGGCTGTACGCTAAGGTGATTGAATTTGATCTGCATTGGA
GCTGTACCCACCTTGATGGCATTGACAGCCTAAAGCGTGAAGAATGCAATACAGCTGACAGAAAAATAACGGGC
TCGATAACGTTCCAAGATTCTGACTTAACGACGGCTAGCGAGCGAGTCATAAATCCCGTCCACACCGGGCAATCGG
GTCGGAGTGGAAGGGCGGGATTTTATTATTACGTGACGAGATCTCCGTGTCACTATACTCACATCTCTCTGTA
GATAAAGTTATACCAACCTCCATATTCTTCTTACGCTAAGTTCGGGCTATCCGAGTCTCGGCCCATAGCAGGAGCA
CTTTAAGGGAAGTCTATTGCCGAATACAGTACGTTCCCCGCAATATGTTATACTCACCCAAATATGTTAATATGT
TATATGTTAAACGCAGTGTGGGAATATGTTAATATGTTATGTGAATATGTTAAATATGTTAAATATGTTAAGC
ATGTTAGCCGTGATAAATATGTTAATACGGCGTGCCTTAATATGTTAGCGACGACTGGGGGTCAATATGTTAGCC
AATTCCTCAATATGTTAACCAGTAAATATGTTAGTTAAGATCAATAAATATGTTAGCTACGTAGACAATAAAGC
ATAAGCAATATGTTAATATGTTAGACAGTTCTCTAACCAGATAATATGTTAAGGCATACTTAACCAGCGAATGAC
AGAAATATGTTAATATGTTAAATAAATATGTTAGATAATATGTTACGATATTACCCGCACATTGCTCCGAATATGA
ATATGTTAAGGTGGTCTCCGTATTTAATATTGTGAGAGATAGCTTGTGAGAGATGTTGTTGTGAGAGAGCTGTGA
GAGCTTTGCGAGCCTTTAAATATTGTGAGAGTTATGTAGTCGGCTGTGAGAGATGTGTGAGAGAGTTAAATATGT
TAGTTAGCTGAGAGCTACGCTGTGAGAGGCGAATTGACGTAGTGCTTTTTGTTCTGTGAGAGATGTGTGAGAGTGT
GAGAGCGCGTGTGAGTGTGAGAGTGATTGTGCATGGTCCAGTAAGATGTGAGAGTGTGAGAGTGATCTAACGCTAT
GTGTGAGAGAGGGTGTGAGAGGCTGCGTATGAAGCACAAATGTGAGAGTTGTGAGAGATACGTTAAGAGCCCGGAA
GCTCGGCATCATAAGCTGAGCAGATTCAATGTGAGAGGGCGAGCCGACGGTAGGCTGTGAGAGTCATTATTGTGAG
AGTCGCGTGGTGTGAGAGTCCATTTTATGTGAGATGTGAGAGCTCTGGGGCTGTGATGTGAGAGAGTACGCCGAA
GCGTGTGAGAGTCTGTGAGAGATTCGGAGGTCTGGATGACATTGTGAGAGCCTGCTTACGCGACGTGATGAACGC
GACCGACTAGCGACCGCCCACTACTACTCGCAGTTGGTCTAGAGGCATTGCTTTACTGAAATACGCAGGATGCTTA
TGACGCTCGCGCCAATACATCGCGCTCGCACTGTATGTGCTTACCTTAATCCTAAAGCTCAAATATAACGGAAA
AAGAGAAATTAGGACGACCGAGGGTCTCCTCCGGTGGTTTTTACGACTTCGCCAATGGCGTGCTGCGTCGAAATG
TGCTCAAAGCCCCGTAAAGCTCAGACACCATGCAGGAATGGGAATGTGTACCCAGAGATCCCTAGTAAGAGAGATC
CAAGACTTAAAGCCGTTCCGAGAGAGATCTAATCACTAGAGATCTTAACACCAATAGAGATCCTCTAAGAGATCAG
TAGAGATCGCTTTTTCAGAGATAGAGATCACTACCGAGAGATCTTACAGTTTGATATGTCAGTTCGGTTAAAAGCA
GAGATCGTCTGCAGAGATCGGTAGCGTAGAGATCCCGTGTGCTACAAAACCTTAGAGATCAGATCGCGCTCGAAC
TGTACTTAGAGATCTACATTATCTAAGAGAGAGATCAGAGATCACAAGGCCACACACGACAAAGTTAGAGATCTAC
ACACGATAGGTGGTGCCGAACCTGAGAGATCCGGGTTTTGAGAGAGATCAAGAGATAGAGATCGTTAGAGAAGAGA
TCTAGAGATCGCACGGGTTTTGGAGAGATCGTTCCGGGTTTTGTGCGGAAGAGATTAATGCCGTGAGTTAATGCCGG
TATAATGCGGCAGATAATGTAATGCCGTCTAATGTAATGCCGGGAGATAATGCCGTGATGAACTTAATGCCGGCTAA
TGCGGTTAATGCCGTGCAACGCTAATGCCGAGCTAATGCCGGGCGTAACATAATGTAATGCCGTGTCAATATTGTT
TTCAATATTAATATTCAATTACAATATTCAAACGCAATATTAACGGCCGGTAATGCCGGGTCAATCAATATTTTC
GTAATGCCGGGTTAATGCCGTTTTCAATATTATCGGTAATGCCGGGAGCTGGCAATATTGGTTTTGTAATGCCGTT
```

TAATAATGCGGGGCGACAATATTGGGTAATGCGGATATTTATCAATATTGTGTTTCAATATTTAACACAATATTTG
 CCGTAGGTATGACCTAATTAATGCGGATATTAGGGGCCAATTAATGCGGATCGTAATGCGGGTCGGGCTTATAACA
 ATTAATGCGGTCAATATTACTAATGCTAATGCGGCGGACTACAATATTTACAAAAGACTACCAATAATGCGGATAA
 TGCGGTCAATAATGCGGAAGATAACGCGGCAATATTGCCCCGACAATATTTGACTACACAAGACTACACAATATTCC
 GTTATTCTGTGCCAACGCCAGGTCAATGCGTCGAACCAATATTCTTGATTGTGATGCAGACTACACGACTACAATA
 TTTACCCCCGGGACTACATATCCACGACTACAGGGCGAGACTACATAGGGACTACAGACTACAACAATTATGGTCA
 CATTAACTCTGCCCCGGCGGCTCTTCCCTAAATCTCACGTGATGGACTAGACTACACCGACTACAACATACTTTGCA
 ACGACTACAGTACGTTAAGACTACAGGATTACAGACTACACTTGATTTCTGACTACACTTCTGACAACCCGCACAT
 TGCCCGCTAACTCTGATGGCCCCCAGAGACTACATACCATCGAGCGCGACTACAGGACTACAGCCGTAGACCCTTT
 AGACTACACGCCAGGGCCAAGTACACGGATAAGGTCTTTGCCCGCAAGTGCTCGCCGAATGTGATTAATCTCAA
 CATTCCGACCTGCAAGAGCACACGCATTTGATATGGGTATAAGGAAGATCTCGTCCAGCTATAATGTACAACATTT
 CCCCCTCATGACTTGCTACATAAACAGAATAAGACGTGACGTGCGCAATATAAGACGTACTCGATTGACCGTAAAA
 TTTTTCTAAGAACCAAAATAAGACGTAAGACGTTCACTTAAATAAGACGTAGGGCGTTACCGATAAGACGTTAAGA
 CGTGATCGCCATCGCCCCGTGAGTCGCTCTCCCGCATAAGACGTTAAGACGTCCCAATAGTGCTCCCTACACTTTA
 CCGGTGGTAGATAAGACGTAGACGTTATAAGACGTGCGGTAAATATAAGACGTTATTCCCAATAAATAAGACGTAA
 TCCCTGTAACACTGGAAGTGATAAGACGTTTGTCTAACATAAGACGTTGTAAGTGCCTAACCTGATAAGACGT
 TTTAAAAAGTACTATAAGACGTTTCGAGGAATGAGACCATAAGACGTCGTCCCTCCCTCAGCACTGAATTTTTTCGA
 AGATAAGACGTAAGACGTGTTGGTTTATCGTTAGAAATAAGACGTACGTTTATAAGACGTAATGGTCATAAGACGT
 ACGTTAAGACGTAATAAGACGTTATCCATCCCAAATTACACGTCAGAAATCATGGCAACCGCCGTGATGGAAGAG
 AGTAGCAACCGACTACATACAGTATACTGTGGGCAGACTCGTTTGACACCAACACTTCCGCCGCCATTATTAAT
 ACGATTGGTGCTTTACGCATCTTGATGACCATGGTTACTCACCTCGGGTGCTGACCCGCCTGTCTCCTATGACGTC
 GGGCTCCACTACGGCCCCGTTTTCGACAGATAGGGGGGAGTTGACCTCGAATGCGGGTACTTCGCTGCCTTTTCGA
 CGAATCGGTATGGCTAGCTTGGACAAGTATAGGATTGGTCTTTCAAGCTGCACTGTTTTGCAGCTTCTAGCGAGAT
 AAGGCTGAAGCCTCCAGCGATATTGTCCAGTTGGAAAAAAGTTGGAAAAATGGGGGTTTTGAAAAAAGAAAACGCC
 CGGGTTACACCGGGGACATAATTGAAAAAACAGTTGGAAAAAGCTTAGAAAGCTTGAAAAAGTCTTGGAACA
 ATTATTGAAAAACGATGGGCGACTGAGAGTTGAAAAAATTTGAAAAATTTGAAAAACGTGGCTTTGAAAAA
 ATGAAAAAAGATTGAAAAATTTGAAAAACTTTTGAATTGAAAAAAGCCACTGCGGGTGCTTTGAAAAAAT
 ATTTGAAAAAACTAGCAAAGCGGCATTCTGAGAGATTGAAAAACGTGCTAAGCTTCTTTGAAAAAGAATTGGA
 AAAAAAAGCGCACCACTCAGGAAGACATGTCTGGCACTTTAGCGTTAAAGTTTGGAAAAAATCCTCCACATT
 TGGAAAAATGAAAAAGAATCGGTTAGAGCGGCACGTGTCATATTGAAAAAATACTCAGCGCGTTAGCAGTTGGAA
 AAAAATGATGACTATGTTTGAAGACAAGGAGAAAGTCTCCGAACAACATCCATGACAAGGAGGAGGCTGGACAAG
 GATTCAGGCTGTTTCAGACAAGGAGGACGACAAGGAAGGACTGTTTCAGGCTGGACAAGACAAGGACTGTTGACAAG
 GACAGGACAAGGACGAAAGGCTGTTTCAGGGACAAGGAAGGACAGGCTGTTTCAGAGGACGAGGACGACAAGGAAGGC
 TGTTTCAGGCTGTTTCAGGAGGACGAGGAAGGATGTTTCAGACAAGGAGGACGACAGGCTAGGACGACGAAGAGGACGACT
 GTTCAGGCTGTAGGACGAGGACGAAAGGATGTTGACAAGGAGGAGGAGGACGAGGAAGGACGAAAGGAGACAAG
 GAGACAAGGAGACAAGGAAGGACGAAGGACGAAGGACGAAGGACGAAGGACGAAGGACGAAGGACGAGGACAGGA
 CGACGAAGGACGAAGGACGAAGGACAGGACGAAGGACGAAGGACGAAGGACGAATCATCAATCATCAATCATCA
 ATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCAT
 CAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCA
 TCAATCATCAATCAT'

In [11]:

```

1 #memodulkan setiap fungsi yang akan digunakan
2 """ Fungsi append_n adalah menambahkan nilai array pada nilai akhir dan membagi data DNA
3 Rumusnya a nilai sebelumnya data DNA, b nilai yang baru dari data DNA dengan batasan ir
4 def append_n(dat,i,n):
5     return reduce(lambda a,b: a+b, dat [i: i+n])
6
7 """ Fungsi remap adalah memetakan kembali dari sequence yang baru dan anyaknya urutan c
8 def remap(dat, seq):
9     return map( lambda x: append_n(dat, x, len(seq)), range (len(dat) - len(seq) + 1))
10
11 """pada fungsi ke-2 remap dihitung jumlah sequence yang muncul """
12 def count_mer(dat, seq):
13     return reduce (lambda a,b : a + (1 if b == seq else 0), remap(dat,seq), 0)
14

```

In [12]:

```
1 list(remap(dat, 'ACT'))[-1] # mengubah dna "act" ke dna yang lain
```

Out[12]:

'CAT'

In [13]:

```
1 len (seq) # mencari panjang seq
```

Out[13]:

3

In [14]:

```
1 sequences=['A', 'AT', 'GGT', 'AAGC', ' AGCTA']
2 def count_all(dat, sequences):
3     return map(lambda x: count_mer(dat, x), sequences)#melakukan perhitungan mer pada s
4 res = count_all(dat, sequences) # merupakan jumla pengembalian berapa kali res terjadi
5 print(*res)
```

2112 557 77 22 0

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

In [15]:

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

In [16]:

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

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

6. Buatlah fungsi feed-forward

In [17]:

```
1 import math # library yang digunakan
```

In [19]:

```
1 #memdulkan setiap fungsi yang ada
2 def aktivasi (x):
3     return 1/ (1+math.exp(-x))
4 '''# untuk fungsi ini merupakan fungsi aktivasi yang telah ada pada soal yang dimana
5     kita hanya menuliskan saya, dengan parameternya adalah x'''
6
7 def WTi(W,i): # Index yang tercantum dalam W digunakan didalam List
8     return list(map( lambda w:w[i] , W))
9
10 def WT(W):
11     return list( map( lambda i : WTi(W,i) , range(len(W[0])))) )
12
13 def XW(X,W):
14     return map( lambda w: reduce( lambda a,b:a+b ,map (lambda xx,ww: xx * ww , X,w),0)
15
16 def input_to_hidden(X,W):
17     return list( map( lambda x:aktivasi(x) , XW(X,W) ))# menginput aktivasi tersembunyi
18
19 def feed_forward(X,W,M):
20     return input_to_hidden(input_to_hidden(X,W),M)#membuat fungsi feed forward
```


In [20]:

```
1 """pada bagian ini kita menentukan setiap nilai dari si x, w, dan m"""
2 x= [9, 10, -1]
3 w= [[0.5, 0.4], [0.3, 0.7], [0.25, 0.9]]
4 m= [[0.34], [0.45]]
```

In [21]:

```
1 feed_forward(x, w, m)
2 """kita memanggil fungsi feed_forward, diaman Nilai yang masuk ke neuron di hidden layer
3 adalah penjumlahan antara perkalian weight dengan nilai yang masuk pada input neuron setelah
4 fungsi aktivasi(yang tela dimodulkan di atas ). """
```

Out[21]:

'kita memanggil fungsi feed_forward, diaman Nilai yang masuk ke neuron di hidden layer\nadalah penjumlahan antara perkalian weight dengan nilai yang masuk pada input neuron setelah itu diaktifkan dengan\nfungsi aktivasi(yang tela dimodulkan di atas). '

In [22]:

```
1 WTi(w,0) # memanggil fungsi WTI
```

Out[22]:

[0.5, 0.3, 0.25]

In [23]:

```
1 WT(w) #Memanggil fungsi WT
```

Out[23]:

[[0.5, 0.3, 0.25], [0.4, 0.7, 0.9]]

Kumpulan Latihan dan Pembahasan

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')]

In [50]:

```
1 P = 'akulupa'
2 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]

In [52]:

```
1 B = 24
2 two = map( lambda b: b+1 if B % (b+1) == 0 else -1, range(B))
3 def dua(two):
4     x = []
5     for t in two:
6         if t != -1:
7             x.append(t)
8     return x
9 print( dua(two))
10
```

[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!

In [53]:

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

-16

Pertemuan 9 Higher order Function Filter

Latihan filter, 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

In [16]:

```
1 # cara 1
2 faktor = lambda n: list(filter(lambda i : n%i == 0, range (1, n+1)))
3 primes = lambda n: filter(lambda i: len (faktor(i))==2, range(1, n+1))
4 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

In [17]:

```
1 # cara 2
2 L = lambda n: range(2, n)
3 faktor = lambda n: list(filter( lambda i: n%i == 0, L(n)))
4 primes = lambda n: filter( lambda i : len(faktor(i))==0, range(2,n+1))
5
6 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

In [18]:

```
1 # cara 3
2 L = lambda n: range(2, n)
3 s_factor = lambda n: filter(lambda i: len(factor(i))>0, range (2, n+1))
4 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

In [22]:

```
1 # cara 4
2 L = lambda n : range(2, round(n**0.5)+1)
3 s_factors = lambda n: map(lambda i: i if n%i == 0 else 0, L(n))
4 primes = lambda n: filter( lambda i: sum(s_factor(i))== 0, range (2, n+1))
5
6 print(*primes(100))
```

In [23]:

```
1 s_factor = lambda n: map(lambda i: i if n%i==0 else 0, range (2, round(n**0.5)+1))
2 gjl = lambda n : n%2!=0
3 L = lambda n:filter( lambda i: sum(s_faktors(i))==0, L(n))
4 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

In [24]:

```
1 #Latihan Filter
2 employee= {'Nagao': 35,
3            'ishii': 30,
4            'Kazutomo':20,
5            'Saito': 25,
6            'Hidemi': 29}
7 print(employee.items())
```

dict_items([('Nagao', 35), ('ishii', 30), ('Kazutomo', 20), ('Saito', 25), ('Hidemi', 29)])

In [25]:

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

Nagao ishii 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]

In [41]:

```
1 L= [2, 1, 9, 10, 11, 90, 3, 15]
2 r(lambda a, b: a+ (1 if b % 2 == 0 else 0), L, 0)
```

Out[41]:

3

2. Buat fungsi untuk menghitung n! Menggunakan reduce!

In [54]:

```
1 facto = lambda n: r(lambda a,b: a*b if b>1 else 1, range(1, n+1), 1)
2 for i in range(10+1):
3     print( str(i) + '!=', facto(i) )
4
```

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

In [42]:

```
1 from functools import reduce
2 n= 4
3 print(reduce(lambda x, y: x*y, range(1, n+1)))
```

24

3. Hitung euclidian distance dari dua vektor berikut menggunakan higher order function! X = [2,5,6,7,10] Y = [-2,9,2,-1,10]

In [44]:

```
1 X = [2,5,6,7,10]
2 Y = [-2,9,2,-1,10]
3
4 eclid = lambda X, Y: r(lambda a, c: a+c, map(lambda x, y: (x-y)**2, X, Y))**0.5
5 eclid(X,Y)
```

Out[44]:

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 }

In [48]:

```
1 #Latihan Filter
2 employee= {'Nagao': 35,
3            'ishii': 30,
4            'Kazutomo':20,
5            'Saito': 25,
6            'Hidemi': 29}
7
8 cnt_emp = lambda lim, employee: r(lambda a, b:a+1 if b[1] > lim else a, employee.items()
9 cnt_emp(25, employee)
```

Out[48]:

3

5. Buatlah deret fibonacchi menggunakan higher order function!

In [3]:

```
1 # NO Recursive menggunakan reduce
2 fibo = lambda n : r(lambda a, b: a if b[0] <=1 else a + [a [ b[0]-1] + a[ b[0] -2 ]],
3 enumerate( [0,1] + list(range(1, n) )), [0,1]) if n>0 else[0]
```

In [7]:

```
1 for i in range(20):
2     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]
Fibonacci 10 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
Fibonacci 11 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
Fibonacci 12 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144]
Fibonacci 13 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233]
Fibonacci 14 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377]
Fibonacci 15 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610]
Fibonacci 16 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987]
Fibonacci 17 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597]
Fibonacci 18 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584]
Fibonacci 19 -> [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181]
```

In [8]:

```
1 fibo(10)
```

Out[8]:

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

In [1]:

```
1 # dengan menggunakan rekursif
2 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)
```

Pertemuan 11 Recursionin FP

In [39]:

```
1 #contohh
2 def factorial(x):
3     if x == 1:
4         return 1
5     else:
6         return (x * factorial(x-1))
7 num = 3
8 print("The factorial of", num, "is", factorial(num))
```

The factorial of 3 is 6

In [40]:

```
1 # Recursive Lambda
2 lambda_factorial = lambda i: 1 if i == 0 else i * lambda_factorial(i-1)
3
4 print(lambda_factorial(4))
```

24

1. 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!

In [55]:

```
1 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)
2 deret_fibo_rec = lambda n: list( map(lambda i: fibo_rec(i), range(n+1)))
3 deret_fibo_rec(10)
```

Out[55]:

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

```
1 Jadi kesimpulannya dalam membuat fibonacci, lebih baik menggunakan non recursive
2 terlihat dari hasil running yang lebih
3 cepat
```

Pertemuan 12 Purity and Immutability

In [43]:

```
1 # Contoh pure function
2 def min(x, y):
3     if x < y:
4         return x
5     else:
6         return y
```

In [46]:

```
1 # Contoh yang bukan pure function
2 def pows(L):
3     for i in range(len(L)):
4         L[i] = L[i]**exponent
5     return L
```

In [48]:

```
1 # mengubah menjadi pure function
2 exponent = 2
3 L = [1,2,3,4,5]
4 print( 'pows(L): ' , pows( L ) )
5 print( 'L : ' , L )
6 Res = pows( L )
7 print( 'Res: ', Res )
8 print( 'L:', L )
9
```

```
pows(L): [1, 4, 9, 16, 25]
L : [1, 4, 9, 16, 25]
Res: [1, 16, 81, 256, 625]
L: [1, 16, 81, 256, 625]
```

1. Ubah fungsiku menjadi pure function!

In [9]:

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

In [10]:

```
1 n = 3
2 L = [3, 5, 7, 8]
3 print (fungsiku(L))
```

```
[10, 16, 22, 4.0]
```

In [11]:

```
1 print(L)
```

[10, 16, 22, 4.0]

2. Ubah fungsiku2 menjadi Pure function

In [12]:

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

In [13]:

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

[16, 3.0, 22, 25]

[16, 3.0, 22, 25]

3. Apakah isi dalam tuple tup ada yang dapat diubah? tup = ([3, 4, 5], 'myname')

In [57]:

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

```
-----
TypeError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_6500\899989142.py in <module>
      1 tup = ([3, 4, 5], 'myname')
----> 2 tup[0]=[3,2,3]
      3 print("tup 2 :", tup)
```

TypeError: 'tuple' object does not support item assignment

```
1 kesimpulan: type data tuple tidak dapat di ubah datanya, kecuali langsung mengubah
dari fungsinya/kodenya
```

Pertemuan 13 Function Building Function

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!

In [15]:

```
1 #a. fungsi komposisi
2 addku = lambda x:x +10
3 powku = lambda x: x**2
4 kurku = lambda x: x - 2 * x
5
6 f_com= lambda f, g: lambda x: f(g(x))
7 my_f_com = f_com(kurku, f_com(powku, addku))
8
9 my_f_com(10)
```

Out[15]:

-400

In [16]:

```
1 #b. fungsi invers
2 inv_addku = lambda x:x +10
3 inv_powku = lambda x: x**2
4 inv_kurku = lambda x: x - 2 * x
5
6 my_f_kom_inv= f_com(inv_addku, f_com(inv_powku, inv_kurku))
7 my_f_kom_inv(-200)
```

Out[16]:

40010

In [50]:

```
1 Latihan Penentuan UKT Mahasiswa
2
3 Universitas di Lampung ITARE, ingin memiliki sistem penentuan golongan UKT dan
4 jumlah biaya UKT yang dibayarkan oleh Mahasiswa berdasarkan Kriteria berikut:
5
6 1. Jumlah tanggungan
7 2. Jumlah token listrik selama 3 bulan terakhir
8 3. Gaji Orang tua / Penanggung jawab
9 4. Penerima program KIP-K atau bukan
10
11 1. Ketentuan Jumlah Tanggungan :
12 Jika lebih >= 5, maka skor = 1
13 Jika < 5 , maka skor = 5 - jumlah tanggungan
14
15 2. Ketentuan token listrik:
16 Jika rata-rata lebih dari 100 ribu per bulan , maka skor = 3
17 Jika diantara 50 ribu - 100 ribu per bulan , maka skor = 2
18 Jika dibawah 50 ribu , maka skor = 1
19
20 3. Ketentuan Gaji :
21 Jika gaji penanggung jawab > 10 juta maka skor = 7
22 Jika 8 < gaji <= 10 juta, maka skor = 6
23 Jika 6 < gaji <= 8 juta , maka skor = 5
24 Jika 4 < gaji <= 6 juta , maka skor = 4
25 Jika 3 < gaji <= 4 juta , maka skor = 3
26 Jika gaji < 3 juta , maka skor = 2
27
28 4. Jika mahasiswa memiliki KIP-K , maka skor = 1, jika tidak maka skor = 5
29 Skor_total = 20 % * skor_1 + 30 % * skor_2 + 20% skor_3 + 30% skor_4
30 Jumlah bayar UKT = biaya pokok + skor_total * 500 ribu
31 Uang Pokok = 750 ribu
32
33 Gunakan fungsi komposisi untuk menyelesaikan masalah tersebut!
34 Hitung berapa biaya UKT yang harus dibayarkan dengan input sebagai berikut:
35 1. Jumlah tanggungan = 3
36 2. Listrik 3 bulan terakhir = 120 ribu , 75 ribu , 50 ribu
37 3. Gaji Penanggung jawab = 5.5 juta per bulan
38 4. Peserta KIP-K = Tidak
39
```

In [22]:

```
1 from functools import reduce as r
2
3 mycompose = lambda * funcs: r( lambda f, g:lambda x:f(g(x)), reversed(funcs), lambda x:
```

In [23]:

```
1 # ketentuan jumlah tanggungan
2 def skor1(jtg):
3     return 1 if jtg >= 5 else 5-jtg
```

In [24]:

```
1 # ketentuan token listrik
2 def skor2(X):
3     def rata(X):
4         return sum(X) / len(X)
5
6     def l_cond_1(X):
7         return [X, [X > 100000] ]
8
9     def l_cond_2(X):
10        return [X[0], X[1] + [ X[0] >= 50000] ]
11
12    def to_score2(X):
13        return r( lambda a,b: a + (1 if b==True else 0), X[1], 1)
14
15    compose_cond = mycompose(rata, l_cond_1, l_cond_2, to_score2)
16    return compose_cond(X)
```

In [25]:

```
1 # ketentuan gaji
2 def con_1(X):
3     return [X[0], 1, X[2], [ X[0] > X[2] [X[1] ] ] ]
4
5 def con_2_to_n(X):
6     return [X[0], X[1] + 1, X[2], X[3] + [X[0] > X[2] [X[1]] ]]
7
8 def to_score(X):
9     return r( lambda a,b: a + (1 if b==True else 0), X[-1], 2)
10
11 def prep(gj):
12     return [gj, 0, list(map( lambda x: x*100000, list(range(10,3,-2)) + [3]) )]
13
14 def skor3(gaji):
15     commpy = mycompose(prepare, con_1, *(con_2_to_n for i in range(4)), to_score )
16     return commpy(gaji)
```

In [26]:

```
1 # ketentuan KIP-K
2 def skor4(X=True):
3     return 1 if X else 5
```

In [27]:

```
1 def combineskor(X):
2     return X + [map(lambda f,x: f(x), X[1], X[0] )]
3
4 def boboti(X):
5     return r( lambda a,b:a+b, map(lambda x,y:x*y, X[-1], [0.2, 0.3, 0.2, 0.3]) )
6
7 def toUKT(X):
8     return 750000 + X*500000
```

In [28]:

```
1 mhs = [3,
2       [120000, 75000, 50000],
3       5.5 * 10**6,
4       False
5       ]
6
7 datas = [mhs, [skor1, skor2, skor3, skor4] ]
8 compose_fin = mycompose(combineskor, boboti, toUKT)
9 compose_fin(datas) /10**6
```

Out[28]:

2.4

```
1 Turunan Polinom
2 Contoh input: "-3x^5 + 2x^2 + 4x + 5"
3
4 output : " 15.0x^4 + 4x -4 "
5
```

In [15]:

```
1 #Turunan Polinom
2
3 def splt(dat):
4     return dat.replace(' ','').replace('-','+-').split('+')
5
6 def chdepan(dat):
7     return dat[1:] if dat[0]=='-' else dat
8
9 def eqkan(dat):
10    return map(lambda x: x if '^' in x else x+'^1' if 'x' in x else 'x^0', dat)
11
12 def toarr2d(dat):
13    return r(lambda a,b:a + [ [ float(hurf) for hurf in b.split('x^')] ], dat,[])
14
15 def sortdesc(dat):
16    return sorted(dat,key = lambda x:x[1], reverse=True)
17
18 def calctur(dat):
19    return map( lambda x: [0,0] if x[1]==0 else [ x[1] * x[0], x[1]-1 ], dat)
20
21 def tostr(dat):
22    return map(lambda x: '0' if x[0]==0 else str(x[0]) if x[1]==0 else str(x[0]) + 'x'
23
24 def prettykan(dat):
25    return r(lambda a,b : a+'+' + b if b != '0' else a, dat, '')
26
27 def prettysign(dat):
28    return dat.replace('+-',' -').replace('+','+')
```

In [17]:

```
1 dat = '-3x^5 + 2x^2 - 4x + 5'
2 fss = (splt,chdepan,eqkan,toarr2d,sortdesc,calctur,tostr,prettykan,prettysign)
3 my_turunan = mycompose(*fss)
```

In [16]:

```
1 splt(dat)
```

Out[16]:

```
['', '-3x^5', '2x^2', '-4x', '5']
```

c. 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!

In [64]:

```
1 from functools import reduce as r
2
3 keranjang = [
4     {'Jumlah_Barang': 5, 'Harga': 10},
5     {'Jumlah_Barang': 7, 'Harga': 20},
6     {'Jumlah_Barang': 20, 'Harga': 4.5}
7 ]
8 def pajak_decorator(func):
9     def inner(*args, **kwargs):
10         res = func(*args, **kwargs)
11         print('Sub Total: ', res)
12         print('Pajak: ', res * 0.11)
13         print('Total: ', res + res * 0.11)
14         return res
15     return inner
16
17 import time
18
19 def calc_time_decorator(fu):
20     def inner(*args, **kwargs):
21         waktu_awal = time.time()
22         res = fu(*args, **kwargs)
23         waktu_akhir = time.time()
24         print('Waktu eksekusi: ', waktu_akhir - waktu_awal)
25         return res
26     return inner
```

In [65]:

```
1 @calc_time_decorator
2 @pajak_decorator
3 def hitung_pembayaran_1(keranjang):
4     return r(lambda a,b: a + (b['Jumlah_Barang'] * b['Harga']), keranjang, 0)
5 hitung_pembayaran_1(keranjang)
```

Sub Total: 280.0

Pajak: 30.8

Total: 310.8

Waktu eksekusi: 0.0010008811950683594

Out[65]:

280.0

In [68]:

```
1 @calc_time_decorator
2 @pajak_decorator
3 def hitung_pembayaran_2(keranjang):
4     s = 0
5     for k in keranjang:
6         s = s + k['Jumlah_Barang'] * k['Harga']
7     return s * 1000
8 hitung_pembayaran_2(keranjang)
```

Sub Total: 280000.0

Pajak: 30800.0

Total: 310800.0

Waktu eksekusi: 0.010005474090576172

Out[68]:

280000.0

Latihan UAS

In [69]:

```
1 import numpy as np
2 from random import randint
3 from functools import reduce as r
```

In [70]:

```
1 def fungsiw(a, b):
2     W= np.random.uniform(a, b, size=(3, 2))
3     return W
4 W=fungsiw(-1, 1)
5 W
```

Out[70]:

```
array([[ 0.54260981,  0.07212758],
       [-0.01271604,  0.39396045],
       [ 0.97375111,  0.13003532]])
```

In []:

```
1
```