Jurnal_PBF_M2_120450081_Dimas_Wahyu_Saputro

May 16, 2022

1 Tugas Pendahuluan - PBF - Modul 2

Dimas Wahyu Saputro 120450081

1.1 Soal

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

1.2 1

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

Referensi: Matrix

[ ]:
```
'''
Karena ingin mencetak bilangan skalar a, sebanyak n, digunakan map().
map() mempunyai struktur map(function, iterable(s))

1. pada function, saya menggunakan lambda. x menjadi parameter, dan a menjadi␣
↪ekspresi.
    akan selalu mengembalikan nilai a.
2. iterable, yaitu range(0,n). Artinya, akan membuat urutan angka dari 0 hingga␣
↪n.

Ketika map(lambda x: a, range(0, n)), akan mengembalikan bilangan skalar a,␣
↪sebanyak n.
'''
def ulangi_081(a, n):
    return list(map(lambda x: a, range(0, n)))

''' saya disini ingin mencetak vektor 1xn, dengan bilangan = 10, dan n = 4 '''
ulangi_081(10, 4)
```

[ ]:
```
[10, 10, 10, 10]
```

1.3 2

Buatlah deret bilangan sebagai berikut dengan input n sebagai panjang deret:

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

Referensi: Deret - RuangGuru

[ ]:
```
import functools

''' Membuat fungsi untuk menghitung nilai deret ke-n '''
def calc_seq(n):
    return pow(-1, n + 1) * (1/pow(2, n))

'''
Membuat fungsi untuk mencetak deret dari deret ke-1 sampai ke-n
Langkah awal, menggunakan map(). map() mempunyai struktur: map(function,␣
↪iterable(s))

1. pada function, saya menggunakan lambda. x menjadi parameter, dan calc_seq(x)␣
↪menjadi ekspresi.
    Artinya, dari setiap nilai iterable nanti, akan mereturn nilai calc_seq(x).
2. iterable, yaitu range(1, n+1). Artinya, akan membuat urutan angka dari 1␣
↪hingga n+1.
'''
def gen_seq(n):
    return list(map(lambda x: calc_seq(x), range(1, n+1)))

gen_seq(4)
```

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

1.4 3

Jumlahkan deret bilangan tersebut!

[ ]:
```
import functools

'''
Untuk menjumlahkan dari deret yang sudah ada, dibutuhkan reduce() function.
fungsi reduce() menghasilkan suatu nilai kumulatif dari operasi fungsi masukan␣
↪terhadap nilai pada iterable masukan.

'''
def sum_seq(n):
    seq = gen_seq(n)
    return functools.reduce(lambda x, y: x+y, seq, 0)

sum_seq(4)
```

[ ]:
```
0.3125
```

1.5 4

Sebuah DNA dimodelkan dalam sebuah string menjadi sequence TCGA dan disimpan ke dalam. Hitunglah kemunculan pola!

Referensi: Count the number of times a letter

[ ]:
```
''' Membuka file .txt '''
filename = 'DNA.txt'
dat = open(filename, 'r').read()

''' Ketika file .txt dibuka, diakhir akan terdapat string \n. Hal ini harus␣
 ↪kita cegah, karena kita hanya ingin string DNA '''
dat = dat[:-1]
#dat
```

[ ]:
```
'TGTCTTCCGGCTGAGCGGTTCCTAACCAGCAGACTGATACTGGTCGAATATCGACGGGCAAGAGCCCTGGGATTGATGC
GTTTCACCATGCGCGTCTCAGTGCAGGCAGGAATGCAGAGCTTACTTCAAACTAGTTACTGGCAAAAAATACAAATTTTT
TCGATCGACCTTGAGTTTATTCATTACCGCACAGTCTTTTACCGCACCTGTTACCGCACATCCGTAAGTTTACCGCACGT
TACCGCACTACCTCTCTATATTACCGCACTTCGTTTACCGCACGCTGAGGAACGGTTACCGCACTTACCGCACCACAAGG
TGCGTGCTCTGTTATTACCGCACCACCATTACCGCACGCACTTTTATTACCGCACCAGGGCACAGCCACGTAGGGTAGCG
TCGTTCTCACTGTATTGCGGCGACGGTCGTAATTTACCGCATTACCGCACCACTCGTTAGCTTACCGCACCTAGGGTTGT
TACCGCACGACTTACCGCACAGCCGTTACCGCACGTGTTACTTGACGCTCTAACTCCCACTCATATCAGTCTTATTACCG
CACACTGGGCTTACCGCACCCGCACCTTAAGTAGGCAGTTACCGCACGTATTACCGCACGTAATTACCGCACACCTGTAA
AGGCAGGGTAAAGTACAGACTTACCGCTTACCGCACGGTTGCACCACGACAAATCTAACGTTAGGTACGTTACCGCACGG
GAAATTACCGCACTCCAGGGTTTTACCGCACAGATATCCATTCGGGAATGTGACCCCTGGAGTGGAGTTGTGCGAAAGAT
ACGGAGTTTTCAAGGGCACACCCAGCTATGTTATTAAGCGTTACAGTGGCCGCTGCATCATGTCAATGTTCAGGTCATTC
TCTATCTTGCTATGTACGAACCCTCGTTAAGAGGGAGTAAGCGATCTTTTGACAAAATCGTATGCATGTAGGCGAGGCAA
TGCCGATTACATTGAACGGCGGGACTTTTCGTATGAGACACCGCGGTTGAAATATTTTTTTATGCAAGAGCGGGATTGGG
CGGAAGGAGACTTAACGCAGTGCCTAGCACTGTTAACTGCGGCATGGCCGGATGGACTACCTATTTTGCAGCTCCAGCGT
TTGAGTTCCACGTACTGACGGAACAGTCCCGAGATAGGCCATGTGGTCGATCCCAGTGAGAAATGAGACTCGAGATGCCG
GTACCGGTAGCATCACCACATTGCTCCAGTATGATATCAGTCTTCACTGTCAGCAATTAATGCAGCGATCTTGAAGAGAG
TTATTCATCTCTTATCACCTGACAATAAATCAATTTACCAGTCAAATTCTCTTTAACATCGTGCCGAACTGCGATGCGTC
GTAGTCTAGATTAGGATATATTTTCTTAGCTGGCTTCGATGATTGGCTGTACGCTAAGGTGATTGAATTTCGATCTGCAT
TGGAGCTGTACCCCACCTTGCATGGCATTGACAGCCTAAAGCGTGAAGAATGCAATACAGCTGACAGAAAAATAACGGGC
TCGATAACGTTCCAAGATTCTGACTTAACGACGGCTAGCGAGCGAGTCATAAATCCCGTCCACACCGGGCAATCGGGTCG
GAGTGGAAAGGGCGGGATTTTATTATTACGTGACGCAGATCTCCGTGTCACTATACTCACATCCTCTCTGTAGATAAAGT
TATACCAACCTCCATATTCTTCTTACGCTAAGTTCGGGCTATCCGAGTCTCGGCCCATAGCAGGAGCACTTTAAGGGAAG
TCCTATTGCCGAATACAGTACGTTCCCCGCAATATGTTATACTCACCCAAATATGTTAATATGTTATATGTTAAAACGCA
GTGTGGGAATATGTTAATATGTTATGTGAATATGTTAAAATATGTTAAAATATGTTAACGATGTTAGCCGTGATAAATAT
GTTATTAACGGCGTGCGTTAATATGTTAGCGACGACTGGGGGTCAATATGTTAGCCAACTTCCTCAATATGTTAACCGGT
TAATATGTTAGTTAAGATCAATAAATATGTTAGCTACGTAGACAATAAAAGCATAAGCAATATGTTATAATATGTTAGAC
AGTTCTCTAACCGATAATATGTTAAGGCATACTTAACCAGCGAATGACAGAAATATGTTAATATGTTAAATAAATATGTT
AGATAATATGTTACGATATTACCCGCACATTGCTCCGAATATGAATATGTTAAGGTGGTTCTCCGTATTTAATATTGTGA
GAGATAGCTTGTGAGAGATGTTGTTGTGAGAGAGCTGTGAGAGCTTTGCGAGCCTTTAAATATTGTGAGAGTTATGTAGT
CGGCCTGTGAGAGATGTGTGAGAGAGTTAAATATGTTAGTTAGCTGAGAGCTACGCTGTGAGAGGCGAATTGACGTAGTG
```

CTTTTTGTTCTGTGAGAGATGTGTGAGAGTGTGAGAGCGCGTGTGAGTGTGAGAGTGATTGTGCATGGTCCAGTAAGATG
TGAGAGTGTGAGAGTGATCTAACGCTATGTGTGAGAGAGGGTGTGAGAGGCTGCGTATGAAGCACAAATGTGAGAGTTGT
GAGAGATACGTTAAGAGCCCGGAAGCTCGGCATCATAAGCTGAGCAGATTCAATGTGAGAGGGCGAGCCGACGGTAGGCT
GTGAGAGTCATTATTGTGAGAGTCGCGTGGTGTGAGAGTCCCATTTTATGTGAGATGTGAGAGCTCTGGGGCTGTGATGT
GAGAGAGTACGCCGAAGCGTGTGAGAGTCCTGTGAGAGATTCGGAGGTCTGGATGACATTGTGAGAGCCTGCTTACGCGA
CGTGATGAACGCGACCGACTAGCGACCGCCCACTACTACTCGCAGTTGGTCTAGAGGCATTGCTTTACTGAAATACGCAG
GATGCTTATGACGCTCGCGCCAATACATCGCGCTCGCACTGTATGTCGCTTCACCTTAATCCTAAAGCTCAAATATAACG
GAAAAAGAGAAATTAGGACGACCGAGGGTCGTCCTCCGGTGGTTTTCACGACTTCGCCAATGGCGTGCTGCGTCGAAATG
TGCTCAAAGCCCCGTAAAGCTCAGACACCATGCAGGAATGGGAATGTGTACCCAGAGATCCCTAGTAAGAGAGATCCAAG
ACTTAAAGCCGTTCCGAGAGAGATCTAATCACTAGAGATCTTAACACCAATAGAGATCCTCTAAGAGATCAGTAGAGATC
GCTTTTCAGAGATAGAGATCACTCACCGAGAGATCTTACAGTTTGATATGTCAGTTCGGTTAAAAGCAGAGATCGTCTGC
AGAGATCGGTAGCGTAGAGATCCCGTGTCGTACAAAAACTTAGAGATCAGATCGCGCCTCGAACTGTACTTAGAGATCTA
CATTATCTAAGAGAGAGATCAGAGATCACAAGGCCACACACGACAAAGTTAGAGATCTACACACGATAGGTGGTGCCGAA
CCTGAGAGATCCGGGTTTTGAGAGAGATCAAGAGATAGAGATCGTTAGAGAAGAGATCTAGAGATCGCACGGGTTTTGGA
GAGATCGTTCGGGTTTTGTCGGGAAGAGATTAATGCGGTGAGTTAATGCGGGTATAATGCGGCAGATAATGTAATGCGGT
CTAATGTAATGCGGGAGATAATGCGGTGATGAAACTTAATGCGGCTAATGCGGTTAATGCGGTCGAACGCTAATGCGGAG
CTAATGCGGGCGTAACATAATGTAATGCGGTTGTCAATATTGTTTTCAATATTAATATTCAATTACAATATTCAAACGCA
ATATTAAACGGCCGGTAATGCGGGGTCAATCAATATTTTCGTAATGCGGGGTTAATGCGGTTTTCAATATTATCGGTAAT
GCGGGAGCTGGCAATATTGGTTTTGGTAATGCGGTTTAATAATGCGGGGCGACAATATTGGGTAATGCGGATATTTATCA
ATATTGTGTTTCAATATTTAACACAATATTTGCCGTAGGTATGACCTAATTAATGCGGATATTAGGGGCCAATTAATGCG
GATCGTAATGCGGGTCGGGCTTATAACAATTAATGCGGTCAATATTACTAATGCTAATGCGGCGGACTACAATATTTACA
AAAGACTACCAATAATGCGGATAATGCGGTCAATAATGCGGAAGATAACGCGGCAATATTGCCCGACAATATTTGACTAC
ACAAGACTACACAATATTCCGTTATTCTGTGCCAACGCCAGGTCAATGCGTCGAACCAATATTCTTGATTGTGATGCAGA
CTACACGACTACAATATTTACCCCCGGGACTACATATCCACGACTACAGGGCGAGACTACATAGGGACTACAGACTACAA
CAATTATGGTCACATTAACTCTGCCCGGCGGCTCTTCCCTAAATCTCACGTGATGGACTAGACTACACCGACTACAACAT
ACTTTGCAACGACTACAGTACGTTAAGACTACAGGATTCAGACTACACTTGATTTCCTGACTACACTTCTGACAACCCGC
ACATTGCCCGCTAACTCTGATGGCCCCCAGAGACTACATACCATCGAGCGCGACTACAGGACTACAGCCGTAGACCCTTT
AGACTACACGCCAGGGCCAACTGACACGGATAAGGTCTTTGCCCCGCAAGTGCTCGCCGAATGTGATTAATCTCAACATT
CCGACCTGCAAGAGCACACGCATTTGATATGGGTATAAGGAAGATCTCGTCCAGCTATAATGTACAACATTTCCCCGTCA
TGACTTGCTACATAAACAGAATAAGACGTGACGTCGCCAATATAAGACGTACTCGATTGACCGTAAAATTTTTCTAAGAA
CCAAAATAAGACGTAAGACGTTCACTTAAATAAGACGTAGGGCGTTACCGATAAGACGTTAAGACGTGGATCGCCATCGC
CCGTGAGTCGCTCTCCCGCATAAGACGTTAAGACGTCCCAATAGTGCTCCCTACACTTTACCGGTGGTAGATAAGACGTA
GACGTTATAAGACGTCGGGTAAATATAAGACGTTATTCCCAATAAATAAGACGTAATCCCTGTAACACTGGAAGTGATAA
GACGTTTGTTCTAACATAAGACGTTGTAACTGCCCTAACCCTGATAAGACGTTTTAAAAAGTACTATAAGACGTTCGAGG
AATGAGACCATAAGACGTCGTCCCTCCCTCAGCACTGAATTTTTTCGAAGATAAGACGTAAGACGTGTTGGTTTATCGTT
AGAAATAAGACGTACGTTTATAAGACGTAATGGTCATAAGACGTACGTTAAGACGTAATAAGACGTTATCCATCCCAAAA
TTACACGTCAGAAATCATGGCAACCGCCGTGATGGAAGAGAGTAGCAACCGACTACATACAGTATACTGTGGGCAGACTC
GTTTGTACACCAACACTTCCGCCGCCATTATTAAATACGATTGGTGCTTTACGCATCTTGATGACCATGGTTACTCACCT
CGGGTGCTGACCCGCCTGTCTCCTATGACGTCGGGCTCCACTACGGCCCCGTTTCGACAGATAGGGGGGAGTTGACCTCG
AATGCGGGTTACTTCGCCTGCCTTTCGACGAATCGGTATGGCTAGCTTGGACAAGTATAGGATTGGTCTTTCAAGCTGCA
CTGTTTTGCAGCTTCTAGCGAGATAAGGCTGAAGCCTCCAGCGATATTGTCCAGTTGGAAAAAAGTTGGAAAAATGGGGG
TTTGGAAAAAAGAAAACGCCCGGGTTACACCGGGGACATAATTGGAAAAAACCAGTTGGAAAAAGCTTAGAAAGCTTGGA
AAAAGTCTTGGGAACAATTATTGGAAAAACGATGGGCGACTGAGAGTTGGAAAAAAAAATTGGAAAAATTGGAAAAACGT
GGCTTTGGAAAAATGGAAAAAGATTGGAAAAATTGGAAAAACTTTTGGAATTGGAAAAAAAAGCCACTGCGGGTGCTTTG
GAAAAAATATTTGGAAAAAACTAGCAAAGCGGCATTCTGAGAGATTGGAAAAACGTGCTAAGCTTCTTTGGAAAAAGAAT
TGGAAAAAAAAAAGCGCACCACTCAGGAAGACATGTCTGGCACTTTAGCGTTAAAGTTTGGAAAAAACTCCTCCCACATT
TGGAAAAATGGAAAAAGAATCGGTTAGAGCGGCACGTGTCATATTGGAAAAATACTCAGCGCGTTAGCAGTTGGAAAAAA

```
ATGATGACTATGTTTGGAAGACAAGGAGAAAGTCTCCGAACAACATCCATGACAAGGAGGAGGCTGGACAAGGATTCAGG
CTGTTCAGACAAGGAGGGACGACAAGGAAGGACTGTTCAGGCTGGACAAGACAAGGACTGTTGACAAGGACAGGACAAGG
ACGAAAGGCTGTTCAGGGACAAGGAAGGACAGGCTGTTCAGAGGACGAGGACGACAAGGAAGGCTGTTCAGGCTGTTCAG
GAGGACGAGGAAGGATGTTCGACAAGAGGACGACAGGCTAGGACGACGAAGAGGACGACTGTTCAGGCTGTAGGACGAGG
ACGAAAGGATGTTGACAAGGAGGAGAGGAGGACGAGGAAGGACGAAAGGAGACAAGGAGACAAGGAGACAAGGAAGGACG
AAGGACGAAGGACGAAGGACGAAGGACGAAGGACGAAGGACAGAGGACAGGACGACGAAGGACGAAGGACGAAGGACAGG
ACGAAGGACGAAGGACGAAGGACGACAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCA
TCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCA
ATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCATCAATC
ATCAATCATCAATCATCAATCATCAATCATCAATCATCAATCAT'
```

[ ]:
```python
import functools as ft

'''
Karena harus dimodularkan, fungsi append_n berfungsi untuk mendapatkan string terpecah dari data, mirip seperti split().
Pada fungsi append_n digunakan reduce, supaya mengembalikan satu nilai.
Misalkan append_n(dat, 0, 3), maka akan mengembalikan nilai 'TGT'
'''
def append_n(dat, i, n):
    return ft.reduce( lambda a,b:a+b , dat[i:i+n] )

'''
fungsi remap() berguna untuk membuat list yang berisi string terpecah dari data.

Digunakan map, dengan fungsi lambda yang akan memproses setiap item dari iterable,
dengan iterable range(len(dat) - len(seq)), artinya panjang data - panjang seq.

Misalkan list(remap(dat, 'ACT')), akan mengembalikan ['TGT', 'GTC', 'TCT', ....]
'''
def remap(dat, seq):
    return map( lambda x: append_n(dat,x,len(seq)) , range(len(dat) - len(seq)+ 1 ) )

'''
Setelah didapatkan list, dibuat fungsi count_mer() untuk menghitung berapa kali seq tertentu muncul.
digunakan reduce() akan menghasilkan suatu nilai kumulatif dari operasi fungsi masukan.
'''
def count_mer(dat, seq):
    return ft.reduce( lambda a, b: a + (1 if b == seq else 0) , remap(dat, seq), 0 )
```

[ ]:
```
append_n(dat, 1, 4)
```

[ ]:
```
'GTCT'
```

[ ]:
```
sequences = [ 'A', 'AT', 'GGT', 'AAGC', 'AGCTA' ]

''' fungsi untuk menghitung kemunculan sequences, menggunakan fungsi yang sudah␣
 ↪dicoba di atas '''
def count_all(dat, sequences):
    return map ( lambda x: count_mer(dat,x), sequences )

res = count_all(dat, sequences)
print(*res)
```

```
2112 557 77 22 5
```

1.6 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 !

[ ]:
```
''' Metode get() mengembalikan nilai item dengan kunci yang ditentukan. '''
def komplemen(x):
    return {'A':'T', 'T':'A', 'C':'G', 'G':'C' }.get(x)

''' Mereverse komplemen data menggunakan map '''
def reverse_komplemen(dat):
    return map( lambda x: komplemen(x), dat)
```

[ ]:
```
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 A G C T T A T A G C T G C C C G T T C T C G G G A C C C T A A C T A C G C
A A A G T G G T A C G C G C A G A G T C A C G T C C G T C C T T A C G T C T C G
A A T G A A G T T T G A T C A A T G A C C G T T T T T T A T G T T T A A A A A A
G C T A G C T G G A A C T C A A A T A A G T A A T G G C G T G T C A G A A A A T
G G C G T G G A C A A T G G C G T G T A G G C A T T C A A A T G G C G T G C A A
T G G C G T G A T G G A G A G A T A T A A T G G C G T G A A G C A A A T G G C G
T G C G A C T C C T T G C C C A A T G G C G T G A A T G G C G T G G T G T T C C A
```

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

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

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

1.7 6

Buatlah fungsi feed-forward!

[ ]:
```
import math

''' Fungsi aktivasi '''
def aktivasi(x):
    return 1/ (1+ math.exp(-x))

'''
Mendapatkan nilai W di setiap list sesuai dengan index.
Mapping W menjadi satu dimensi
'''
def WTi(W, i):
    return list(map( lambda w:w[i], W))

'''
Menampung WTi sesuai index dan membuat menjadi satu list
'''
def WT(W):
    return list( map( lambda i : WTi(W, i), range(len(W[0])) ) )

'''
```

```
Nilai yang masuk ke neuron di hidden layer adalah penjumlahan antara perkalian␣
 ↪weight dengan
nilai yang masuk pada input neuron.
'''
def XW(X,W):
    return map( lambda w: ft.reduce( lambda a,b:a+b, map( lambda xx,vw: xx*vw,␣
 ↪X, w), 0), WT(W) )

''' Mengaktivasi nilai yang didapat pada XW '''
def input_to_hidden(X, W):
    return list( map( lambda x:aktivasi(x) , XW(X, W) ) )

''' membuat feed-forward dari fungsi yang sudah dibuat di atas, supaya modular␣
 ↪'''
def feed_forward(X, W, M):
    return input_to_hidden(input_to_hidden(X, W), M)
```

[ ]:
```
X = [ 9, 10, -4 ]
W = [ [ 0.5, 0.4 ] , [ 0.3, 0.7 ] , [ 0.25, 0.9 ] ]
M = [ [ 0.34 ], [0.45] ]

feed_forward(X, W, M)
```

[ ]:
```
[0.6876336740661236]
```