Membuat list

```
import datetime
import os
import time


# while(True):
tanggal_hari_ini = datetime.datetime.now()


Mahasiswa = [
        "Revo Rahmat",
        "2022071047",
        "Informatika",
        "Desain Analisis Algoritma",
        tanggal_hari_ini.strftime("%x"),
        "Universitas Pembangunan Jaya",
        tanggal_hari_ini.strftime("%X"),
        12,
        True,
        1j,
]


for i in Mahasiswa:
    print(i)

time.sleep(1)
os.system("Cls")
```

```
Revo Rahmat
2022071047
Informatika
Desain Analisis Algoritma
09/17/23
Universitas Pembangunan Jaya
10:58:26
12
True
1j

0
```

Mengeluarkan Isi List

```
bin_colors = ['Red','Green','Blue','Yellow']
for i in bin_colors:
    print(i)

print()
for _ in range(-1,-1 * (1+ len(bin_colors)), -1):
    print(bin_colors[_])
```

```
Red
Green
Blue
Yellow

Yellow
Blue
Green
Red
```

Mencetak Nim

```
print(Mahasiswa[1])
```

```
2022071047
```

```
print(Mahasiswa[len(Mahasiswa) - 2])
```

```
True
```

List Slicing

```
for i in range (0, len(bin_colors) + 1):
    for j in range(0, i):
        print(f"{[j, i]}")
        print(f"{bin_colors[j:i]}")
        print()
```

```
[0, 1]
['Red']

[0, 2]
['Red', 'Green']

[1, 2]
['Green']
```

```
[0, 3]
['Red', 'Green', 'Blue']

[1, 3]
['Green', 'Blue']

[2, 3]
['Blue']

[0, 4]
['Red', 'Green', 'Blue', 'Yellow']

[1, 4]
['Green', 'Blue', 'Yellow']

[2, 4]
['Blue', 'Yellow']

[3, 4]
['Yellow']
```

Menambahkan isi list dengan string

```
for aColor in Mahasiswa:
    print(aColor + " UPJ")
```

```
Revo Rahmat UPJ
2022071047 UPJ
Informatika UPJ
Desain Analisis Algoritma UPJ
09/17/23 UPJ
Universitas Pembangunan Jaya UPJ
10:58:26 UPJ

---------------------------------------------------------------------------
-----
TypeError                                 Traceback (most recent call
last)
c:\Users\revor\OneDrive\Documents\Code_Kuliah\DAA_Pertemuan_3.ipynb
Cell 12 line 2
      <a
href='vscode-notebook-cell:/c%3A/Users/revor/OneDrive/Documents/Code_K
uliah/DAA_Pertemuan_3.ipynb#X14sZmlsZQ%3D%3D?line=0'>1</a> for aColor
in Mahasiswa:
----> <a
href='vscode-notebook-cell:/c%3A/Users/revor/OneDrive/Documents/Code_K
uliah/DAA_Pertemuan_3.ipynb#X14sZmlsZQ%3D%3D?line=1'>2</a>
print(aColor + " UPJ")
```

```
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Struktur dalam python tuple

```
bin_colors = tuple(bin_colors)
print(bin_colors)
bin_colors[0]
```

```
('Red', 'Green', 'Blue', 'Yellow')
```

```
'Red'
```

Susun tuple bernama "UPJ", isi dengan Universitas, Pembangunan, Jaya

```
UPJ = ('Universitas', 'Pembangunan', 'Jaya')
```

Nested Tuple

```
hari_awal = ("Senin", "Selasa", "Rabu")
hari_akhir = ("Kamis", "Jumat", "Sabtu")

hari = (hari_awal, hari_akhir)
print(hari)
```

```
(('Senin', 'Selasa', 'Rabu'), ('Kamis', 'Jumat', 'Sabtu'))
```

Latihan

```
pertama = (100,)
kedua = (200, 400, 600,)
ketiga = (300,)
keempat = (400, 800,)
nested_tuple = (pertama, kedua, ketiga, keempat)
print(nested_tuple)
```

```
((100,), (200, 400, 600), (300,), (400, 800))
```

Dictionary

```
bin_colors = {
    "manual-color" : "Yellow",
    "approved-color" : "Green",
    "refused-color" : "Red",
}
print(bin_colors)
```

```
{'manual-color': 'Yellow', 'approved-color': 'Green', 'refused-color': 'Red'}
```

```
bin_colors.get('approved-color')
```

```
'Green'
```

```
bin_colors['approved-color'] = "magenta"
bin_colors.get('approved-color')
```

```
'magenta'
```

Latihan

```
mahasiswa = {}

x = str(input("Masukkan nama : "))
mahasiswa['Nama'] = x
x = str(input("Masukkan nim : "))
mahasiswa['Nim'] = x
x = str(input("Masukkan prodi : "))
mahasiswa['prodi'] = x
x = str(input("Masukkan universitas : "))
mahasiswa['Universitas'] = x

print(mahasiswa)
```

```
{'Nama': 'Revo Rahmat', 'Nim': '2022071047', 'prodi': 'Informatika', 'Universitas': 'Universitas Pembangunan Jaya'}
```

Sets

```
green = {'Grass', 'leaves'}
print(green)
```

```
{'leaves', 'Grass'}
```

```
green = {'grass', 'leaves', 'leaves'}
print(green)
```

```
{'leaves', 'grass'}
```

```
set_01 =  {4,5,6,2}
print(set_01)
set_02 = set()
set_03 = set([2,1,4,3])
print(set_03)
```

```
{2, 4, 5, 6}
{1, 2, 3, 4}
```

```
set_01 = {4,5,6,2}
print(set_01[1])
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
c:\Users\revor\OneDrive\Documents\Code_Kuliah\DAA_Pertemuan_3.ipynb Cell 31 line 2
      <a href='vscode-notebook-cell:/c%3A/Users/revor/OneDrive/Documents/Code_Kuliah/DAA_Pertemuan_3.ipynb#X42sZmlsZQ%3D%3D?line=0'>1</a> set_01 = {4,5,6,2}
----> <a href='vscode-notebook-cell:/c%3A/Users/revor/OneDrive/Documents/Code_Kuliah/DAA_Pertemuan_3.ipynb#X42sZmlsZQ%3D%3D?line=1'>2</a> print(set_01[1])

TypeError: 'set' object is not subscriptable
```

```
set_01.add(1)
print(set_01)
```

```
{1, 2, 4, 5, 6}
```

```
set_01.discard(6)
print(set_01)
```

```
{1, 2, 4, 5}
```

```
set_01.add(13)
print(set_01)
```

```
{1, 2, 4, 5, 13}
```

Union Set

```
set_A =  {1,2,3,4,}
set_B = {3,4,5,6,}
```

```
print(set_A | set_B)
print(set_A.union(set_B))
```

```
{1, 2, 3, 4, 5, 6}
{1, 2, 3, 4, 5, 6}
```

Intersection / Irisan Set

```
print(set_A & set_B)
print(set_A.intersection(set_B))
```

```
{3, 4}
{3, 4}
```

Difference Set

```
print(set_A - set_B)
print(set_A.difference(set_B))
```

```
{1, 2}
{1, 2}
```

```
print(set_B - set_A)
print(set_B.difference(set_A))
```

```
{5, 6}
{5, 6}
```

Symmetric Difference

```
print(set_A ^ set_B)
print(set_A.symmetric_difference(set_B))
```

```
{1, 2, 5, 6}
{1, 2, 5, 6}
```

Latihan

```
yellow = {'dandelions', 'fire hydrant', 'leaves'}
red = {'rose', 'blood', 'leaves', 'fire hydrant'}
print(yellow.union(red))
print(yellow.intersection(red))
```

```
{'dandelions', 'leaves', 'rose', 'fire hydrant', 'blood'}
{'fire hydrant', 'leaves'}
```

Data Frames

```
import pandas as pd

df = pd.DataFrame([
    ['1', 'Fares', 32, True],
    ['2', 'Elena', 23, False],
    ['3', 'steven', 40, True]
])
df
```

```
   0       1   2      3
0  1   Fares  32   True
1  2   Elena  23  False
2  3  steven  40   True
```

```
df.columns= ['id', 'nama', 'age', 'decision']
df
```

```
  id    nama  age  decision
0  1   Fares   32      True
1  2   Elena   23     False
2  3  steven   40      True
```

```
# Seleksi
df[['nama', 'age']]
```

```
     nama  age
0   Fares   32
1   Elena   23
2  steven   40
```

```
df.iloc[1:3,:]
```

```
  id    nama  age  decision
1  2   Elena   23     False
2  3  steven   40      True
```

```
df[df.age>30]
```

```
  id    nama  age  decision
0  1   Fares   32      True
2  3  steven   40      True
```

```
df[df.age<30]
```

```
  id   nama  age  decision
1  2  Elena   23     False
```

```
df[(df.age<35) & (df.decision == True)]
```

```
  id   nama  age  decision
0  1  Fares   32      True
```

```
Data = {
    'Satu' : [1,1,1,1,1,],
    'Dua' : [2,2,2,2,2],
    'Tiga': [3,3,3,3,3,],
}

df = pd.DataFrame(Data,index=['a','b','c','d','e'])
df.head()
```

```
   Satu  Dua  Tiga
a     1    2     3
b     1    2     3
c     1    2     3
d     1    2     3
e     1    2     3
```

Latihan

```
mhs = {}
prodi, jmlh_mahasiswa, laki_laki, perempuan = [], [], [], []
x = int(input("Berapa banyak data yang kamu masukkan : "))

for i in range(0, x):
    prodi.append(input("Masukkan Prodi : "))
    jmlh_mahasiswa.append(input("Masukkan jumlah mahasiswa : "))
    laki_laki.append(input("Masukkan jumlah mahasiswa laki - laki :"))
    perempuan.append(input("Masukkan jumlah mahasiswa perempuan : "))

mhs['prodi'] = prodi
mhs['Mahasiswa'] = jmlh_mahasiswa
mhs['Laki_Laki'] = laki_laki
mhs['Perempuan'] = perempuan

df = pd.DataFrame(mhs)
df.index = df.index + 1
df.head()
```

```
              prodi Mahasiswa Laki_Laki Perempuan
1       Informatika        50        30        20
2  Sistem Informasi        55        30        25
3      Teknik Sipil        40        30        10
```

Matrix

```
import numpy as np
my_Matrix = np.array([
    [11,12,13,],
    [21,22,23,],
    [31,32,33]
    ])
print(my_Matrix)
print(type(my_Matrix))
```

```
[[11 12 13]
 [21 22 23]
 [31 32 33]]
<class 'numpy.ndarray'>
```

```
matrixA = np.array([[1,2,3],[4,5,6]])

print('Matrix A : ')
print(matrixA)


matrixA = np.transpose(matrixA)
print()

print("Transpose Matrix A : ")
print(matrixA)
```

```
Matrix A :
[[1 2 3]
 [4 5 6]]

Transpose Matrix A :
[[1 4]
 [2 5]
 [3 6]]
```

Latihan

```
matriks = []
baris = int(input("Masukkan banyak baris : "))
kolom = int(input("Masukkan banyak kolom : "))

for i in range(baris):
    row = []
    for j in range(kolom):
        row.append(input(f"Masukkan nilai untuk baris ke {i} : "))
    matriks.append(row)
```

```
matriks = np.array(matriks)
print(matriks)
print()

matriks = np.transpose(matriks)
print(matriks)
```

```
[['100' '200' '300']
 ['700' '600' '500']
 ['900' '1000' '800']]

[['100' '700' '900']
 ['200' '600' '1000']
 ['300' '500' '800']]
```

Data abstrak

```
my_vector = [22,23,44,55]
print(my_vector)
print(type(my_vector))


my_vector = np.array([22,33,44,55])
print(my_vector)
print(type(my_vector))
```

```
[22, 23, 44, 55]
<class 'list'>
[22 33 44 55]
<class 'numpy.ndarray'>
```

Stack

```
class Stack:
    def __init__(self):
        self.items = []
    def isEmpty (self):
        return self.items == []
    def push(self, item):
        self.items.append(item)
    def pop(self):
        return self.items.pop()
    def peek(self):
        return self.items[len(self.items) - 1]
    def size(self):
        return len(self.items)
```

```
stack = Stack()
stack.push('Red')
stack.push('Green')
stack.push('Blue')
stack.push('Yellow')

print(stack)


stack.pop()
stack.isEmpty()
```

```
<__main__.Stack object at 0x000001ACCABDF250>

False
```

Queue

```
class Queue(object):
    def __init__(self):
        self.items = []
    def isEmpty(self):
        return self.items == []
    def enqueue(self, item):
        self.items.insert(0,item)
    def dequeue (self):
        return self.items.pop()
    def size(self):
        return len(self.items)

queue = Queue()
queue.enqueue('red')
queue.enqueue('Green')
queue.enqueue('Blue')
queue.enqueue('Yellow')
print(queue.size())
print(queue.dequeue())
print(queue.dequeue())
```

```
4
red
Green
```

Pohon Faktor

```
# Array isinya primer
count = 0
```

```
primer = []

for i in range(2, 100):
    count = 0
    for j in range(1, i + 1):
        if i % j == 0:
            count = count + 1
    if count == 2:
        primer.append(i)

x = int(input("Masukkan angka yang akan di carikan pohon faktornya :
"))
faktor = {}
count = 0

for i in primer:
    while x % i == 0:
        x = x / i
        count = count + 1
    if count != 0:
        faktor[i] = count
    count = 0

print(faktor)


{2: 3, 3: 1, 5: 1}
```

Binary Tree

```
class Node:
    def __init__(self, data):
        self.left = None
        self.right = None
        self.data = data
    def PrintTree(self):
        if self.left:
            self.left.PrintTree()
        print(self.data)
        if self.right:
            self.right.PrintTree()

root = Node(2)
root.left = Node(3)
root.right = Node(5)
root.left.left = Node(7)
root.left.right = Node(9)
root.right.left = Node(11)
```

```
root.right.right = Node(13)
root.PrintTree()
```

```
7
3
9
2
11
5
13
```