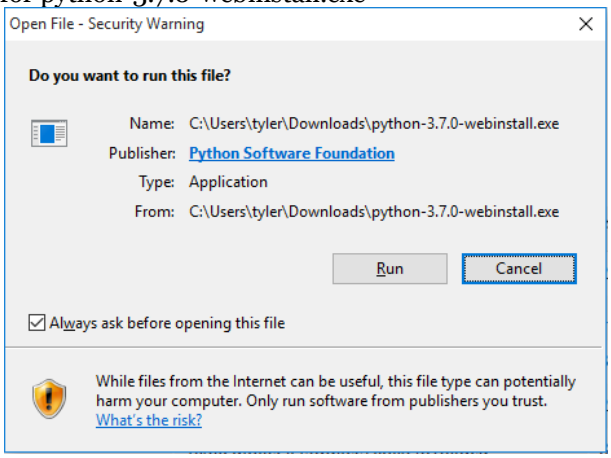
PETUNJUK PRAKTIKUM PYTHON 3.7.0

I. Instalasi

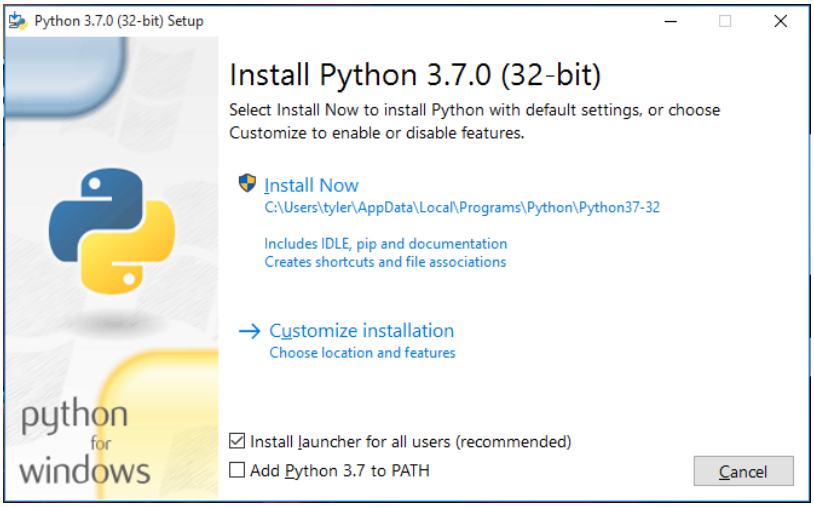
Installing python on windows PC!
If you're running Windows: the most stable Windows downloads are available from the Python for Windows page.

1. Visit the python for windows page.
2. Download python 3.7.0 by clicking the link in step 3.
3. https://www.python.org/ftp/python/3.7.0/python-3.7.0-webinstall.exe
4. Begin the download and follow screenshot instructions below.

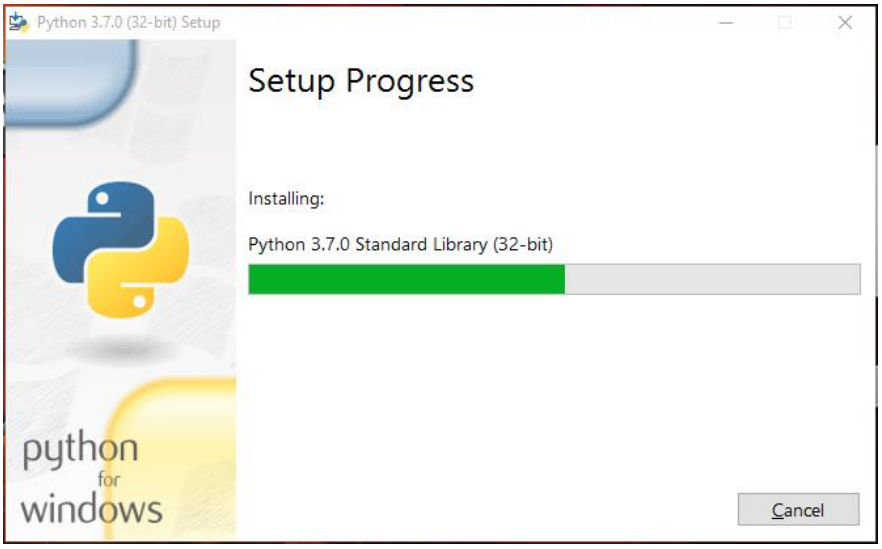
Screenshots from installing python on windows PC Python Release 3.7.0
Open your executable.
Or if you're not that techie, navigate to your download folder...
Or search your computer for python-3.7.0-webinstall.exe



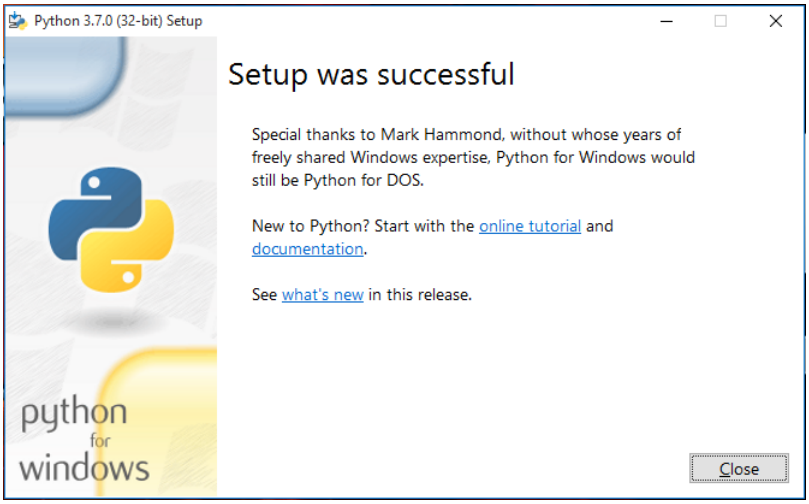
Screen shot 1 from python release 3.7.0



Install now option works for me! Check add python 3.7 to path to add it to your system variables, i guess…. we will look at that later. Worst case scenario, it's unchecked for a reason and reinstall would not hurt you at this point.



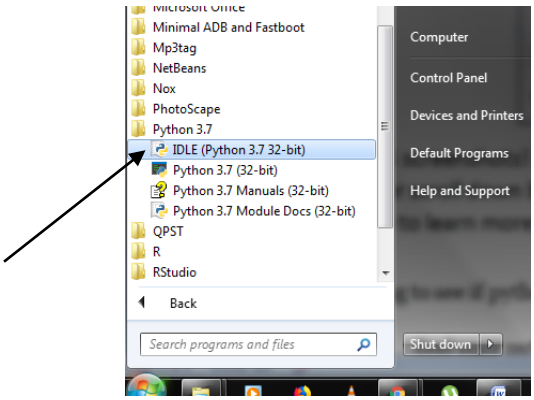
Setup progress — I'm literally always happy to get a screenshot during this progress bar load. I wonder how well this screenshot will rank next to other python 3.7.0 tutorials.

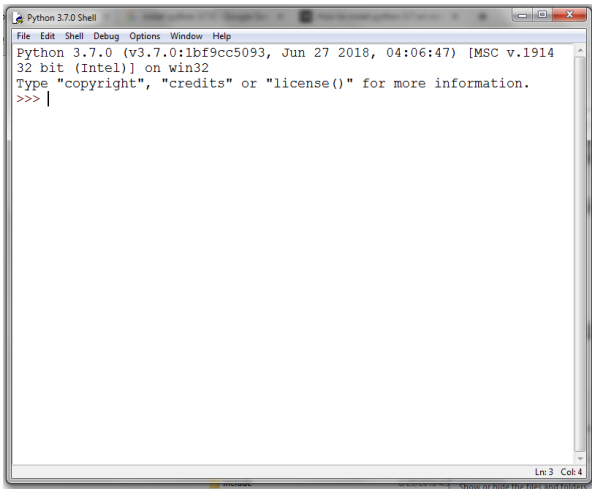


Only 4 screenshots! Not a bad little install. Open online tutorial and documentation to get your beak wet, or scroll down because like I said above, I don't want you to have to google a lot of stuff, scroll below to learn more about python or get a refresher. Also, links to tutorials built by Python!

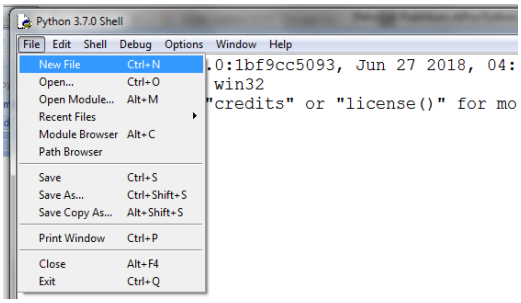
Python is located: C:\Users\tyler\AppData\Local\Programs\Python\Python37–32\

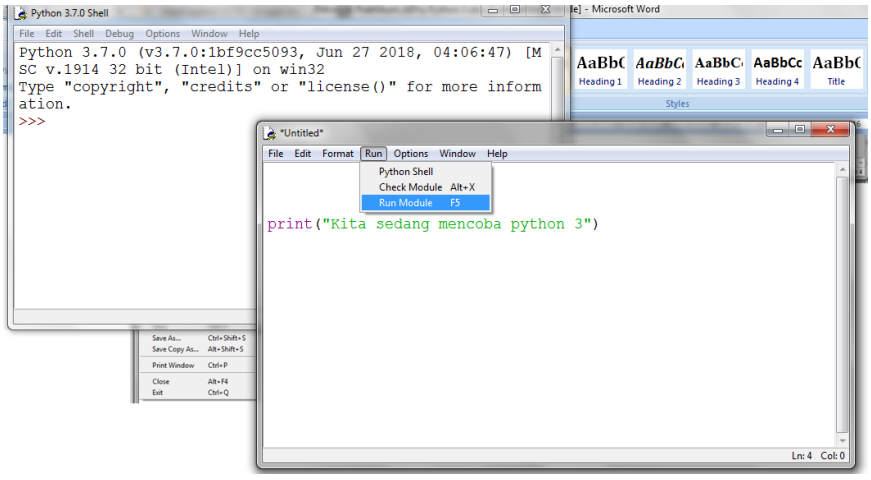
II. Operasi Awal





1. Klik ikon windows dan cari **IDLE (python 3.7 32-bit)** atau cari di **All Program** kemudian cari **Python 3.7** dan klik 2x selanjutnya klik **IDLE (python 3.7 32-bit)**
2. Muncul window **python 3.7.0 shell**
3. Klik **File** → klik **New File**
4. Muncul window baru sebut saja python Work di window ini kita bias ketik koding python yang akan di-run → hasilnya akan muncul di window **python 3.7.0 shell**





III. Input dan Output

print() function to output data to the standard output device (screen).
The actual syntax of the print() function is

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

Output formatting
Sometimes we would like to format our output to make it look attractive. This can be done by using the str.format() method. This method is visible to any string object.

```
x = 5; y = 10
print('The value of x is {} and y is {}'.format(x,y))
# Output : The value of x is 5 and y is 10
```

Here the curly braces {} are used as placeholders. We can specify the order in which it is printed by using numbers (tuple index).

```
print('I love {0} and {1}'.format('bread','butter'))
# Output : I love bread and butter   # menggunakan "..." atau '...' sama saja
print('I love {1} and {0}'.format('bread','butter'))
# Output : I love butter and bread

print('Hello {name}, {greeting}'.format(greeting = 'Goodmorning', name = 'John'))
# Output : Hello John, Goodmorning
```

use the % operator to accomplish this.

```
x = 12.3456789
print('The value of x is %3.2f' %x)
# Output : The value of x is 12.35
print('The value of x is %3.4f' %x)
# Output : The value of x is 12.3457
```

```
# tanda # digunakan untuk komentar atau statement yang
#       didahului oleh tanda ini tidak akan di-eksekusi

"""
    atau untuk komentar yang lebih dari satu baris maka
    tanda " sebanyak 3 kali dan diakhiri dengan Tanda "
```

```
    sebanyak 3 kali
    seperti untuk komentar ini
"""

# ============================================================
# pemberian nilai secara langsung
# ============================================================

a=3
b=7
c=a+b
print("display nilai a, b dan c")
print(a)  # tampil dan ganti baris
print(b)  # tampil dan ganti baris
print(c)  # tampil dan ganti baris

     # hasil run :
     """
     >>>
      RESTART:
     C:/Users/FAMILY/AppData/Local/Programs/Python/Python37-
     32/coba awal.py
     3
     7
     10
     """

print("Ada tambahan end= ")
print(a,end="")  # tampil di baris yang sama tanpa spasi
print(b,end=" ") # tampil di baris yang sama dengan spasi --> angka akan terpisah
print(c)

     # hasil run :
     """
     Ada tambahan end=
     37 10
     """

print("Tampilan ringkas ")
print("Nilai a :",a,", nilai b :",b," dan nilai c :",c)

     # hasil run :
     """
     Tampilan ringkas
     Nilai a : 3 , nilai b : 7  dan nilai c : 10
     """
```

```
# ============================================================
# pemberian nilai lewat keybord
# input lewat keybord
# ============================================================
a=input("Masukkan nilai a :")
print("Nilai a :",a)
# b=a+2   # tidak bisa dioperasikan karena a adalah string dan 2 adalah int
b=int(a)+2
print("Nilai b :",b)

print("-- input -- langsung dikonversi ke int")
a=int(input("Masukkan nilai a :"))
print("Nilai a :",a)
b=a+2
print("Nilai b :",b)

     # hasil run :
     """
     Masukkan nilai a :3  --> entry via keyboard
     Nilai a : 3
     Traceback (most recent call last):
       File
     "C:/Users/FAMILY/AppData/Local/Programs/Python/Python37-
     32/coba awal.py", line 56, in <module>
         b=a+2
     TypeError: can only concatenate str (not "int") to str
     """
     """
     Masukkan nilai a :5
     Nilai a : 5
     Nilai b : 7
     """
     """
     -- input -- langsung dikonversi ke int
     Masukkan nilai a :8
     Nilai a : 8
     Nilai b : 10
     """

# ============================================
# Char -- dianggap String
#  dan
# String
# ============================================

char1=input("Masukkan satu karakter : ")
print("satu karakter :",char1)
```

```
string1=input("Masukkan string : ")
print("string :",string1)


     # hasil run :
     """
     Masukkan satu karakter : f
     satu karakter : f
     Masukkan string : Saya sedang menggunakan command input
     untuk mamasukkan string lewat keyboard
     string : Saya sedang menggunakan command input untuk
     mamasukkan string lewat keyboard

     Masukkan satu karakter : saya seorang programmer
     satu karakter : saya seorang programmer
     Masukkan string : saya bangga menjadi seorang programmer
     string : saya bangga menjadi seorang programmer

     """

# ================================================
#  Conditional  -- Boolean -- True atau False
# ================================================

c1=2>3
print("nilai kondisi 2>3 :",c1)

print("nilai kondisi:",(7==6+1))


     # hasil run :
     """
     nilai kondisi 2>3 : False
     nilai kondisi: True
     """
```

IV. If-else

Syntax

The syntax of the *if...else* statement is

```
if expression:
  statement(s)
else:
  statement(s)
```

```
# ========================================================
# if - else
# ========================================================

a = "python"
if type(a) is str: print(a,"adalah String")

     # hasil run :
     """
     python  adalah String
     """

if 2>3: print("True")
else: print("False")

     # hasil run :
     """
     False
     """

#kondisi=True
kondisi=False
if kondisi: print("nilai kondisi adalah True")
else: print("nilai kondisi adalah False")

     # hasil run :
     """
     nilai kondisi adalah False
     """
```

V. If-elif-else

The elif *Statement*

The **elif** statement allows you to check multiple expressions for TRUE and execute a block of code as soon as one of the conditions evaluates to TRUE.

Similar to the **else**, the **elif** statement is optional. However, unlike **else**, for which there can be at most one statement, there can be an arbitrary number of **elif** statements following an **if**.

syntax

```
if expression1:
  statement(s)
elif expression2:
  statement(s)
elif expression3:
  statement(s)
else:
  statement(s)
```

```
# ========================================================
# if - elif - else
#   seperti perintah
#     switch – case di JAVA
# ========================================================

"""
angka=1 --> cetak angka sama dengan satu
angka=2 --> cetak angka sama dengan dua
angka=3 --> cetak angka sama dengan tiga
angka=4 --> cetak angka sama dengan empat
angka=5 --> cetak angka sama dengan lima
"""

angka=2
#angka=7
if angka==1: print("angka sama dengan satu")
elif angka==2: print("angka sama dengan dua")
elif angka==3: print("angka sama dengan tiga")
elif angka==4: print("angka sama dengan empat")
elif angka==5: print("angka sama dengan lima")
else : print("angka yang anda masukkan SALAH ! ")
```

```
# hasil run :
"""
angka sama dengan dua
# nilai kondisiA adalah True tetapi kondisiB False
"""
```

VI. If-else tersarang (if-else di dalam if-else)

Nested if statements

if...elif...else statement inside another if...elif...else statement. This is called nesting in computer programming.

```
# ========================================================
#  If-else tersarang (if-else di dalam if-else)
# ========================================================

kondisiA=True
kondisiB=False
if kondisiA:
    if kondisiB : print("nilai kondisiA dan kondisiB adalah True")
    else : print("nilai kondisiA adalah True tetapi kondisiB False")
else:
    if kondisiB : print("nilai kondisiA adalah False tetapi kondisiB
True")
    else : print("nilai kondisiA dan kondisiB adalah False")

     # hasil run :
     """
     nilai kondisiA adalah True tetapi kondisiB False
     """
```

VII. Looping For

Syntax of **for** Loop

for val in sequence:

Body of for

Here, val is the variable that takes the value of the item inside the sequence on each iteration. Loop continues until we reach the last item in the sequence. The body of for loop is separated from the rest of the code using indentation.

The **range()** function
We can generate a sequence of numbers using range() function. range(10) will generate numbers from 0 to 9 (10 numbers).
We can also define the start, stop and step size as range(start,stop,step size). step size defaults to 1 if not provided.

```
# ========================================================
#  Looping For
# ========================================================

n=4
for i in range(n) : print("nilai i :",i)

    # hasil run :
    """
    nilai i : 0
    nilai i : 1
    nilai i : 2
    nilai i : 3
    """
n=4
for i in range(1,n) : print("nilai i :",i)

    # hasil run :
    """
    nilai i : 1
    nilai i : 2
    nilai i : 3
    """

n=10
for i in range(1,n,3) : print("nilai i :",i)

    # hasil run :
    """
    nilai i : 1
    nilai i : 4
    nilai i : 7
```

```
    """

# ========================================
#    float range
# ========================================
"""
```

Yield

yield is a keyword that is used like return, except the function will return a generator.

```
>>> def createGenerator():
...    mylist = range(3)
...    for i in mylist:
...        yield i*i
...
>>> mygenerator = createGenerator() # create a generator
>>> print(mygenerator) # mygenerator is an object!
<generator object createGenerator at 0xb7555c34>
>>> for i in mygenerator:
...     print(i)
0
1
4
```

Here it's a useless example, but it's handy when you know your function will return a huge set of values that you will only need to read once.

To master yield, you must understand that when you call the function, the code you have written in the function body does not run. The function only returns the generator object, this is a bit tricky :-)

```
"""

def frange(start, stop, step):
     i = start
     while i < stop:
         yield i
         i += step

# ================================================
#   round(n,m) -- pembulatan m angka dibelakang koma
# ================================================
n=2
for i in frange(0,n,0.4) : print("nilai i :",round(i,2))

    # hasil run :
    """
```

```
    nilai i : 0
    nilai i : 0.4
    nilai i : 0.8
    nilai i : 1.2
    nilai i : 1.6
    """

n=4
for i in range(n,0,-1) : print("nilai i :",i)

    # hasil run :
    """
    nilai i : 4
    nilai i : 3
    nilai i : 2
    nilai i : 1
    """

buah = ["apel", "jeruk", "pisang"]
for x in buah: print("nama buah :",x)

    # hasil run :
    """
    nama buah : apel
    nama buah : jeruk
    nama buah : pisang
    """

for x in "apel dan jeruk": print(x,end="")

    # hasil run :
    """
    apel dan jeruk
    """
```

VIII. Looping while

Syntax of **while** Loop in Python

```
while test_expression:
    Body of while
```

In while loop, test expression is checked first. The body of the loop is entered only if the `test_expression` evaluates to `True`. After one iteration, the test expression is checked again. This process continues until the `test_expression` evaluates to `False`.

while loop with else

Same as that of for loop, we can have an optional else block with while loop as well.
The else part is executed if the condition in the while loop evaluates to False.
The while loop can be terminated with a break statement. In such case, the else part is ignored.
Hence, a while loop's else part runs if no break occurs and the condition is false.
Here is an example to illustrate this.

```
# Example to illustrate
# the use of else statement
# with the while loop

counter = 0
while counter < 3:
   print("Inside loop")
   counter = counter + 1
else:
   print("Inside else")
```

```
# Output
Inside loop
Inside loop
Inside loop
Inside else
```

```
# ========================================================
#  Looping For
# ========================================================

i = 1
while i < 6:
  print("i :",i)
  i += 1
```

```
# hasil run :
"""
i : 1
i : 2
i : 3
i : 4
i : 5
"""
```

```
# ========================================
#   break --- statement
# ========================================

i = 1
while i < 6:
  print("i :",i)
  if i == 3:    # tidak mencetak angka > 3
    break
  i += 1


        # hasil run :
        """
        i : 1
        i : 2
        i : 3
        """

# ========================================
#   continue --- statement
# ========================================

i = 0
while i < 6:
  i += 1
  if i == 3:   # nilai i = 3 TIDAK tercetak
    continue
  print("i :",i)


        # hasil run :
        """
        i : 1
        i : 2
        i : 4
        i : 5
        i : 6
        """
```

IX. Array

Array Methods

Python has a set of built-in methods that you can use on lists/arrays.

Method	Description
append()	Adds an element at the end of the list
clear()	Removes all the elements from the list
copy()	Returns a copy of the list
count()	Returns the number of elements with the specified value
extend()	Add the elements of a list (or any iterable), to the end of the current list
index()	Returns the index of the first element with the specified value
insert()	Adds an element at the specified position
pop()	Removes the element at the specified position
remove()	Removes the first item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list

Array 1 Dimensi

Nilai array sudah diberikan

```
x=[1,2,3,4,5,6]
print("Nilai-nilai x : ", end="")
for i in range(0,len(x)):
    print(x[i],end=" ")

        # hasil run :
        """
        Nilai-nilai x : 1 2 3 4 5 6
        """
```

Nilai array di inputkan lewat keyboard

```
n=5
y=[]
for i in range(n):
    nil=input("masukkan nilai y : ")
    y.append(nil)

print("Nilai-nilai y : ", end="")
for i in range(0,len(y)):
    print(y[i],end=" ")


        # hasil run :
        """
        masukkan nilai y : 2
        masukkan nilai y : 3
        masukkan nilai y : 4
        masukkan nilai y : 5
        masukkan nilai y : 6
        Nilai-nilai y : 2 3 4 5 6
        """
```

Array 2 Dimensi

Nilai array sudah diberikan

```
xx=[[12,2],[13,4],[15,6]]
print("Nilai-nilai xx : ")
for i in range(len(xx)):
    for j in range(len(xx[0])):
        print(xx[i][j],end=" ")
    print()

        # hasil run :
        """
        Nilai-nilai xx :
        12 2
        13 4
        15 6
        """
```

Nilai array di inputkan lewat keyboard

```
nBar=3
nCol=2
yy=[]
for i in range(nBar):
    bar=[]
    for j in range(nCol):
        print("masukkan nilai yy[",i,",",j,"]: ",end="")
        nil=input()
        bar.append(nil)
    yy.append(bar)

print("Nilai-nilai yy : ")
for i in range(len(yy)):
    for j in range(len(yy[0])):
        print(yy[i][j],end=" ")
    print()


        # hasil run :
        """
        masukkan nilai yy[ 0 , 0 ]: 2
        masukkan nilai yy[ 0 , 1 ]: 4
        masukkan nilai yy[ 1 , 0 ]: 3
        masukkan nilai yy[ 1 , 1 ]: 5
        masukkan nilai yy[ 2 , 0 ]: 6
        masukkan nilai yy[ 2 , 1 ]: 8
        Nilai-nilai yy :
        2 4
        3 5
        6 8
        """
```

X. Function atau method

```
# ========================================
#   function
# ========================================
def loop(n):
    for i in range(n):
        print("i :",i)


loop(5)

        # hasil run :
        """
        i : 0
        i : 1
        i : 2
        i : 3
        i : 4
        """
```

XI. class

```
# ========================================
#   class tanpa constructor
# ========================================

class SimpleNo:
     def tampilN(self):
        print("mencoba fungsi tampilN")

     def kaliN(self,angka):
          return angka



osNo = SimpleNo()   # create object class Simple
print("Mencoba clas SimpleNo")
osNo.tampilN()  # eksekusi function tampilN()
hasNo=osNo.kaliN(7)   # eksekusi function kaliN()
print("SimpleNo --- hasil perkalian :",hasNo)


          # hasil run :
          """
```

```
          Mencoba clas SimpleNo
          mencoba fungsi tampilN
          SimpleNo --- hasil perkalian : 7
          """


# ========================================
#   class dengan constructor
# ========================================

class Simple:
     def __init__(self):   # nilai awal by default = 5
         self.n=5

     def tampilN(self):
        print("mencoba fungsi tampilN")

     def kaliN(self,angka):
          return self.n*angka



os1 = Simple()   # create object class Simple
print("Mencoba clas Simple")
os1.tampilN()  # eksekusi function tampilN()
has1=os1.kaliN(7)   # eksekusi function kaliN()
print("Simple --- hasil perkalian :",has1)


          # hasil run :
          """
          Mencoba clas Simple
          mencoba fungsi tampilN
          Simple --- hasil perkalian : 35
          """



class Simple2:
     def __init__(self, angka):  # nilai awal bisa diinput
         self.n=angka

     def tampilN(self):
        print("mencoba fungsi tampilN")

     def kaliN(self,angka):
          return self.n*angka

print("Mencoba clas Simple2")
os2 = Simple2(10)   # create object class Simple
os2.tampilN()      # eksekusi function tampilN()
has2=os2.kaliN(5)   # eksekusi function kaliN()
```

```
print("Simple2 --- hasil perkalian :",has2)

          # hasil run :
          """
          Mencoba clas Simple2
          mencoba fungsi tampilN
          Simple2 --- hasil perkalian : 50
          """
```