KONSEP DATA WAREHOUSE

MIF POLIJE

(SEMESTER 4)

PENDAMPING PRAKTIKUM

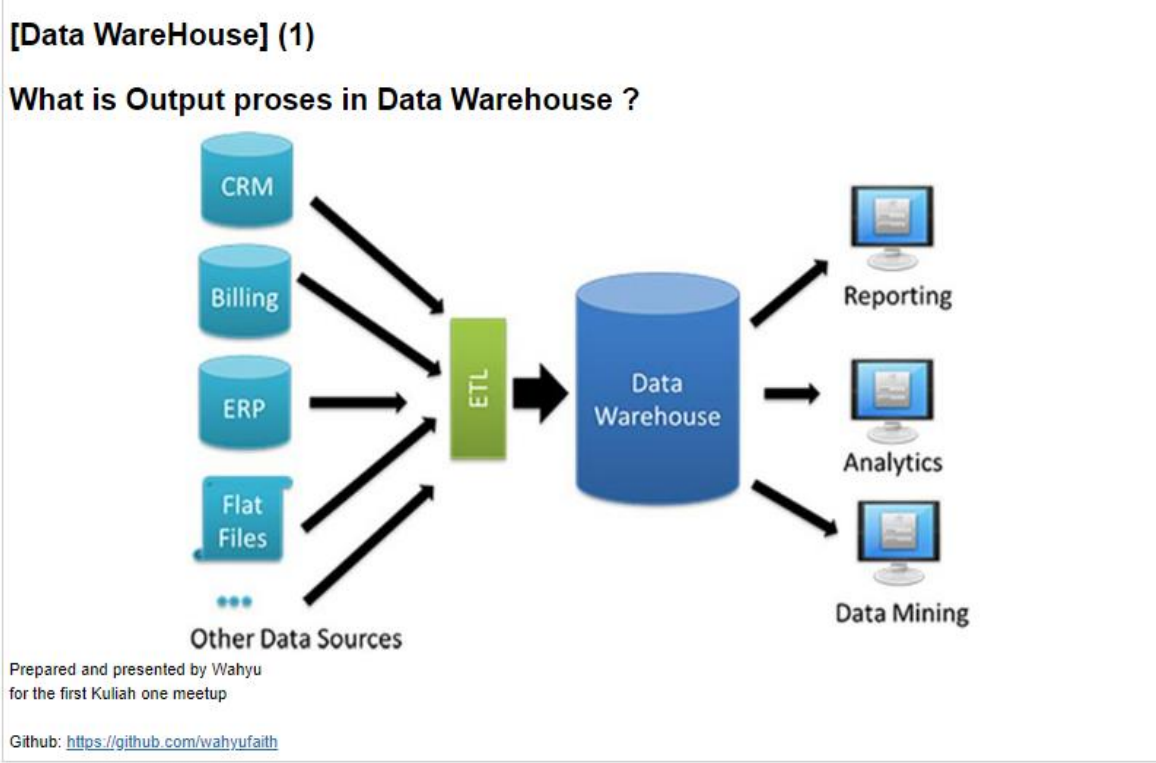
DATA WAREHOUSE PERTEMUAN 1

OLEH :

WAHYU PEBRIANTO

9 MARET 2020

1. Konsep Data Warehouse :



Gambar 1.1 Konsep Data Warehouse

1.1 Karakteristik *Data Warehouse*

Karakteristik *Data warehouse* salah satunya *Integraded* yaitu *data warehouse* di bangun dengan menggabungkan banyak data dari sumber yang berbeda, salah satu contoh *resource CRM(Customer Relationship Management), Resource (Billing), Resource ERP* dan Resource *Flat File.*

1.2 Komponen 1 dalam Data Warehouse

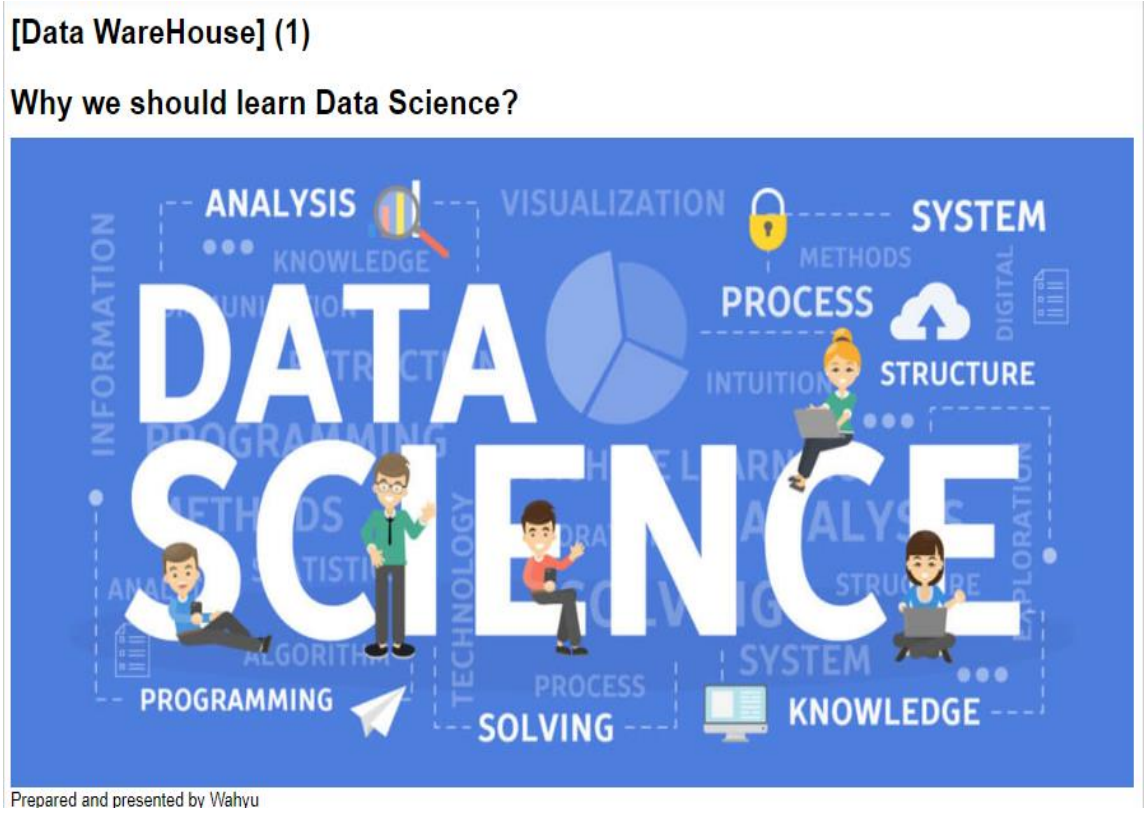
Data Staging Area salah satu komponen data warehouse dalam tahapan ini data di olah dari sumbernya/ sumber *resource*, proses ini terdiri dari *Extract, Transform, Load (ETL).*

1.3 Komponen 2 dalam Data Warehouse

Data Presentation Area merupakan tahapan setelah proses (ETL), dalam tahapan ini menghasilkan data yang diorganisasikan, disimpan dalam data warehouse, sehingga ketersediaan data dapat terjamin apabila dibutuhkan, salah satunya untuk kebutuhan laporan *(report)* dan kebutuhan analisis.

DATA SCIENCE

2. DATA SCIENCE



Gambar 2.1 Gambaran skill *data science*

Seorang yang dapat memanfaatkan penyimpanan data warehouse untuk keperluan analisa salah satunya adalah seorang *data science,*

2.1 Data Science

data science adalah proses penggalian data agar dapat difilter serta didapatkan data yang benar untuk menghasilkan produk data yang sebenar-benarnya, baik untuk memecahkan masalah yang komplek berdasarkan *analitic* yang bersumber dari data, Salah satunya data yang berasal dari penyimpanan *data warehouse.*

Untuk memahami dan mengetahui pekerjaan seorang *data science* kita harus mengetahui *Computer Science (ilmu komputer), Math and Statistics (Matematika dan Statistika),* untuk mendapatkan pengetahuan itu kita dapat memulainya dengan mempelajari *Computer Science,* yaitu mempelajari Bahasa pemograman python yang di mulai dari dasar *(basic).*

3. PYTHON

Gambar 3.1 Python

3.1 Keunggulan python

Salah satu keunggulan Bahasa pemograman python selain menyediakan banyak library yang seakan di kususkan untuk analisa data, Bahasa pemograman ini mudah

untuk pelajari, karena sintaksis program *python* lebih ringkas, sederhana dan mudah di baca. Berikut basic dari Bahasa pemograman python :

3.2 [Data Warehouse] Sintaks pada python (1/5)

statement (1)

Slide Type

Statement adalah sebuah intruksi perintah yang akan dieksekusi atau di tampilkan oleh program, Penulisan satu statement tidak diakhiri dengan tanda titik-koma

Slide Type

```
1 print("Hello World!")
2 print("Halo dan halo")
3 print("Halo Teman-teman semester 4")
```

```
Hello World!
Halo dan halo
Halo Teman-teman semester 4
```

string (2)

Slide Type

String adalah teks atau kumpulan dari karakter, dalam bahasa pemograman python kita dapat menulis string sesuai contoh di bawah ini.

Slide Type

```
1 judul = "Dengan petik dua"
2 penulis = 'dengan petik satu'
3 judul = """dengan petik tiga sebanyak tiga kali"""
4 penulis = '''dengan petik tiga '''
```

Case (3)

Sintak Python bersifat case sensitive, yang berarti antara huruf besar dan huruf kecil akan di bedakan

```
1 print("Halo dan halo")
2 PRINT("Halo Teman-teman semester 4")
```

```
Halo dan halo

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-29-e6e618dde9ab> in <module>
      1 print("Halo dan halo")
----> 2 PRINT("Halo Teman-teman semester 4")

NameError: name 'PRINT' is not defined
```

block of code (4)

Python tidak menggunakan tanda { } untuk menandai blok / grup kode. Blok kode di python menggunakan tanda indentasi (spasi). Jumlah spasi untuk setiap baris yang ada dalam satu blok kode harus sama. Contoh yang benar adalah sebagai berikut:

```
1  if 5 > 2:
2     print("Selamat belajar!")
3     print("Semangat Yah!")
```

```
Selamat belajar!
Semangat Yah!
```

contoh yang salah adalah sebagai berikut :

```
1 if 5 > 2:
2 print("Selamat belajar!")
3 print("Semangat Yah!")
```

```
  File "<ipython-input-9-573409088b48>", line 2
    print("Selamat belajar!")
        ^
IndentationError: expected an indented block
```

Coments (5)

Tanda pagar (#) digunakan untuk menandai komentar di python. Komentar tidak akan diproses oleh interpreter Python. Komentar hanya berguna untuk programmer untuk memudahkan memahami maksud dari kode.

```
#This is a comment
"""
This is a comment
written in
more than just one line
"""
print("Hello, World!") #This is a comment
```

```
Hello, World!
```

3.3 2. [Data Warehouse] Variables [1] and Data Type python

(1) VARIABEL

Variabel adalah lokasi di memori yang digunakan untuk menyimpan nilai. Pada saat kita membuat sebuah variabel, kita memesan tempat di dalam memori, di dalam Bahasa pemograman python kita langsung menginisialisasikan nama variable dan memberi nilai :

```
panjang = 100
lebar = 21.5
nama = "faith"
print(panjang)

#VARIABEL UNTUK MENDEFINISIKAN NILAI YANG SAMA
x = y = z = 3
print(x,y,z)
```

```
100
3 3 3
```

(1) BILANGAN (INTEGER DAN FLOAT) [Data Type]

Kita bisa menggunakan fungsi **type()** untuk mengetahui tipe data suatu objek di Bahasa pemograman python python :

```
x = 5
print(x, "tipenya adalah ", type(x))
x = 2.0
print(x, "tipenya adalah ", type(x))
```

```
5 tipenya adalah  <class 'int'>
2.0 tipenya adalah  <class 'float'>
```

(2) STRING [Data Type]

String adalah tipe data yang anggotanya berurut dan memiliki indeks. Indeks dimulai dari angka 0 bila dimulai dari depan dan -1 bila di indeks dari belakang, [4:7] artinya memesan data yang akan di tampilkan dimulai dari 4 dan pada batas 7 :

```
kalimat = "Saya adalah Programmer Python"
print(kalimat[0])
print(kalimat[-1])
print(kalimat[4:7])
print(kalimat[:4])
```

```
S
n
 ad
Saya
```

(3) LIST [Data Type]

List adalah tipe data yang berisi satu atau beberapa nilai di dalamnya. Nilai – nilai ini sering juga disebut item, elemen, atau anggota list**,** Variable a adalah contoh variable yang di isi list, list di tulis dengan kurung siku [ ] :

```
#LIST AKAN DI ULAS PADA PERTEMUAN BERIKUTNYA
a = [5,10,15,20,25,30,35,40]
```

(4) TUPLE

Tuple mirip dengan list. Bedanya, tuple bersifat immutable, sehingga anggotanya tidak bisa diubah Variable a adalah contoh variable yang di isi tuple, tuple di tulis dengan tanda kurung ( ) :

```
#TUPLE AKAN DI ULAS PADA PERTEMUAN BERIKUTNYA
a = (255,255, 255)
print(a)
```

```
(255, 255, 255)
```

(5) SET

Set adalah salah satu tipe data di Python yang tidak berurut (*unordered*). Set memiliki anggota yang unik (tidak ada duplikasi). Jadi misalnya kalau kita meletakkan dua anggota yang sama di dalam set, maka otomatis set akan menghilangkan yang salah satunya

```
#SET AKAN DI ULAS PADA PERTEMUAN BERIKUTNYA
my_set = {1,2,3,3}
print(my_set)
```

```
{1, 2, 3}
```

3.4 [Data Warehouse] Operator python [1/7]

Operator adalah konstruksi yang dapat memanipulasi nilai dari operan, salah satunya operator aritmatika.

Contoh dalam program dengan menggunakan Bahasa pemograman python :

(1) PENJUMLAHAN

```
1 print(13 + 2)
2 apel = 7
3 jeruk = 9
4 buah = apel + jeruk
5 print(buah)
```

```
15
16
```

(2)PENGURANGAN

```
1 hutang = 10000
2 bayar = 5000
3 sisaHutang = hutang - bayar
4 print("Sisa hutang Anda adalah ", sisaHutang)
```

```
Sisa hutang Anda adalah  5000
```

(3) PERKALIAN

```
1 panjang = 15
2 lebar = 8
3 luas = panjang * lebar
4 print(luas)
```

```
120
```

(4) PEMBAGIAN

```
1 kue = 16
2 anak = 4
3 kuePerAnak = kue / anak
4 print("Setiap anak akan mendapatkan bagian kue sebanyak ", kuePerAnak)
```

```
Setiap anak akan mendapatkan bagian kue sebanyak  4.0
```

4.0 adalah nilai yang belum di bulatkan, apabila ingin dibulatkan kita dapat menggunakan operator '//' pada pembagian. Contoh gambar di bawah ini :

```
1 kue = 16
2 anak = 4
3 kuePerAnak = kue // anak
4 print("Setiap anak akan mendapatkan bagian kue sebanyak ", kuePerAnak)
```

```
Setiap anak akan mendapatkan bagian kue sebanyak  4
```

(5) SISA BAGI / MODULUS

```
1 bilangan1 = 14
2 bilangan2 = 5
3 hasil = bilangan1 % bilangan2
4 print("Sisa bagi dari bilangan ", bilangan1, " dan ", bilangan2, " adalah ", hasil)
```

```
Sisa bagi dari bilangan  14  dan  5  adalah  4
```

(6) PANGKAT

```
1 bilangan3 = 8
2 bilangan4 = 2
3 hasilPangkat = bilangan3 ** bilangan4
4 print(hasilPangkat)
```

```
64
```

(7) PEMBAGIAN

```
1 print(10/3)
```

```
3.3333333333333335
```

Bila di bulatkan :

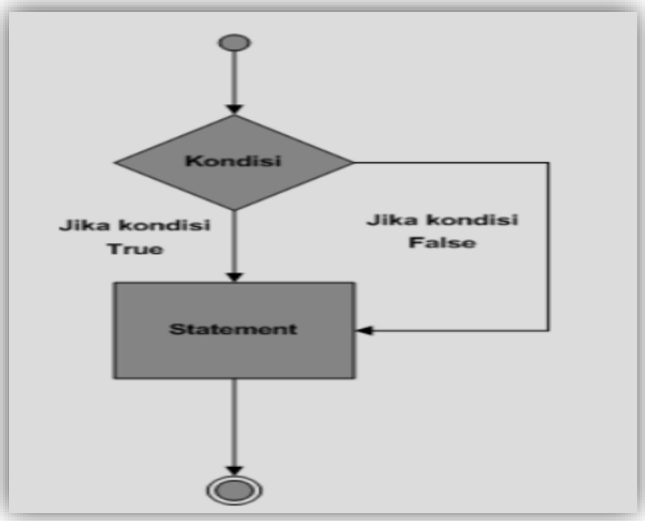
```
1 print(10//3)
```

```
3
```

3.5 [Data Warehouse] Percabangan python [1/3]

Percabangan digunakan untuk mengambil keputusan apabila di dalam program dihadapkan pada kondisi tertentu. Percabangan melakukan evaluasi berdasarkan kondisi true dan false,

(1) PERNYATAAN IF



Pada gambar di atas adalah percabangan if, percabangan if dieksekusi berdasarkan kondisi true dan false, apabila kondisi true program akan menjalankan statement, apabila kondisi false maka program tidak akan menjalankan statement.

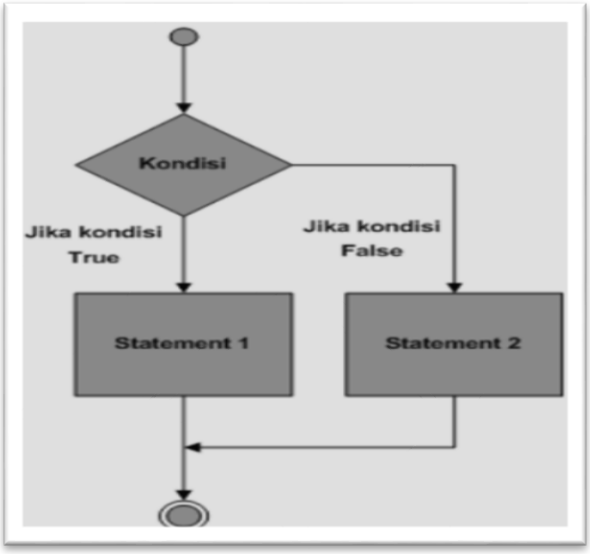
Contoh percabangan if dan perbedaan kondisi true dan false:

```
1 nilai = 9
2
3 if(nilai > 7):
4     print("Selamat Anda Jadi programmer")
5
6 if(nilai > 10):
7     print("Selamat Anda Jadi programmer handal")
```

```
Selamat Anda Jadi programmer
```

Pada gambar di atas menjelaskan blok program if pertama memiliki kondisi true dan blok program kedua dengan kondisi false.

(2) PERNYATAAN IF ELSE



Pada gambar di atas adalah percabangan if else, percabangan if else dieksekusi berdasarkan kondisi true dan false, namun apabila kondisi true program akan

menjalankan statement satu dan apabila kondisi false maka program akan menjalankan statement dua.

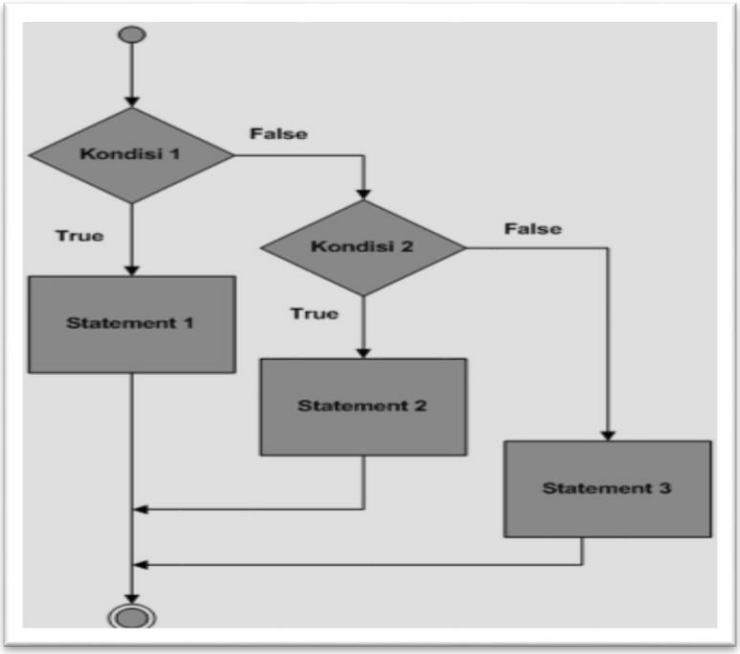
Contoh percabangan if else:

```
nilai = 3
if(nilai > 7):
    print("Selamat Anda Jadi programmer")
else:
    print("Maaf Anda Gagal jadi programmer")
```

```
Maaf Anda Gagal jadi programmer
```

Pada gambar di atas blok program dengan kondisi false, karena yang digunakan adalah percabangan if else maka yang dijalankan adalah statement dua.

(3) PERNYATAAN IF…ELIF…ELSE…



Pada gambar di atas adalah percabangan if ..elif..else.., percabangan if ..elif..else.. dieksekusi berdasarkan kondisi true dan false, namun di percangan ini apabila kondisi satu true program akan menjalankan statement satu dan apabila kondisi false program masih akan mempertimbangkan kondisi dua, apabila kondisi dua true

program akan menjalankan statement 2, apabila false program akan menjalankan statement tiga, percabangan ini digunakan apabila program di tugaskan untuk menyelesaikan masalah berdasarkan berbagai kondisi yang berantai.

Contoh percabangan if.. elif.. else.. :

```
hari_ini = "Minggu"

if(hari_ini == "Senin"):
    print("Saya akan kuliah")
elif(hari_ini == "Selasa"):
    print("Saya akan kuliah")
elif(hari_ini == "Rabu"):
    print("Saya akan kuliah")
elif(hari_ini == "Kamis"):
    print("Saya akan kuliah")
elif(hari_ini == "Jumat"):
    print("Saya akan kuliah")
elif(hari_ini == "Sabtu"):
    print("Saya akan kuliah")
elif(hari_ini == "Minggu"):
    print("Saya akan libur")
```

```
Saya akan libur
```

3.6 [Data Warehouse] Looping python

Secara umum, pernyataan pada bahasa pemrograman akan dieksekusi secara berurutan. Pernyataan pertama dalam sebuah fungsi dijalankan pertama, diikuti oleh yang kedua, dan seterusnya. Tetapi akan ada situasi dimana Anda harus menulis banyak kode, dimana kode tersebut sangat banyak. Jika dilakukan secara manual maka Anda hanya akan membuang-buang tenaga dengan menulis beratus-ratus bahkan beribu-ribu kode. Untuk itu Anda perlu menggunakan perulangan di dalam bahasa pemrograman Python, terdapat dua perulangan yaitu perulangan for dan perulangan while.

(1) FOR LOOP

Perulangan for dengan bentuk umum memiliki sintaks sebagai berikut:

```
angka = [1,2,3,4,5]
for x in angka:
    print(x)
```

```
1
2
3
4
5
```

Perulangan for dengan *range(),* fungsi *range()* dapat digunakan untuk menghasilkan deret bilangan. Contohnya *range(10)* akan menghasilkan bilangan dari 0 sampai dengan 9 atau (10 bilangan), berikut contoh perulangan dengan fungsi *range()* :

```
ulang = 10

for i in range(ulang):
    print ("Perulangan ke-"+str(i))
```

```
Perulangan ke-0
Perulangan ke-1
Perulangan ke-2
Perulangan ke-3
Perulangan ke-4
Perulangan ke-5
Perulangan ke-6
Perulangan ke-7
Perulangan ke-8
Perulangan ke-9
```

Kita juga bisa menentukan batas bawah, batas atas, dan interval dengan format *range(batas bawah, batas atas, interval)*. Bila *interval* dikosongkan, maka nilai default 1 yang akan digunakan :

Berikut for dengan range(batas bawah,batas atas) :

```
1  for x in range(1, 11):
2    print(x)
```

```
1
2
3
4
5
6
7
8
9
10
```

Berikut for dengan *range(batas bawah, batas atas, interval)* :

```
1  for x in range(2, 30, 3):
2    print(x)
```

```
2
5
8
11
14
17
20
23
26
29
```

(2) WHILE LOOP

Perulangan menggunakan *while* akan menjalankan blok program terus menerus selama kondisi bernilai true.

```
1  hitung = 0
2  while (hitung < 9):
3      print ("berhitung: ", hitung)
4      hitung = hitung + 1
5
6  print ("selesai !")
```

```
berhitung:  0
berhitung:  1
berhitung:  2
berhitung:  3
berhitung:  4
berhitung:  5
berhitung:  6
berhitung:  7
berhitung:  8
selesai !
```

Pada gambar di atas apabila kita tidak menulis hitung = hitung + 1 maka akan jadi *infinite loop* atau akan dilakukan perulangan secara terus-menerus. Hasilnya akan jadi seperti berikut:

```
In [*]: 1 hitung = 0
        2 while (hitung < 9):
        3     print ("berhitung: ", hitung)
        4     #hitung = hitung + 1
        5
        6 print ("selesai !")
```

```
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
berhitung:  0
```

3.7 [Data Warehouse] Functions python [1/4]

Sebelumnya anda menggunakan fungsi *print()*, untuk menampilkan statement, fungsi *print()* adalah fungsi bawaan dari Bahasa pemograman python, kali ini kita akan membuat fungsi berdasarkan kebutuhan kita sendiri, berikut conothnya :

(1) MEMBUAT FUNCTION

```
def nama_fungsi():
    print ("Hello ini Fungsi")
```

(2) CALLING A FUNCTIONS

```
1 def my_function():
2   print("hallo saya fungsi anda")
3
4 my_function()
```

```
hallo saya fungsi anda
```

(3) FUNCTIONS WITH PARAMETER

```
def my_function(parameter):
    print(parameter)

my_function(1)
```

```
1
```

```
def luas_segitiga(alas, tinggi):
    luas = (alas * tinggi) / 2
    print ("Luas segitiga:",luas)

# Pemanggilan fungsi dan mengisi parameter/varibel
luas_segitiga(4, 6)
```

```
Luas segitiga: 12.0
```

(4) FUNCTION RETURN VALUE

Fungsi dengan return berguna untuk melakukan pengembalian, contoh kasusnya apabila kita membuat sebuah fungsi perhitungan, namun kita ingin memanggil nilai perhitungan dari fungsi tersebut, maka fungsi tersebut harus di beri return agar dapat di akses di fungsi lain.

Contoh fungsi tanpa return :

```
def my_function(x):
  5 * x

print(my_function(3))
print(my_function(5))
print(my_function(9))
```

```
None
None
None
```

Contoh fungsi dengan return :

```python
def my_function(x):
  return 5 * x

print(my_function(3))
print(my_function(5))
print(my_function(9))
```

```
15
25
45
```

```python
# rumus: sisi x sisi
def luas_persegi(sisi):
    luas = sisi * sisi
    return luas

# rumus: sisi x sisi x sisi
def volume_persegi(sisi):
    volume = luas_persegi(sisi) * sisi

luas_persegi(5)
```

```
25
```

3.8 [Data Warehouse] I/O Python

Kita sering menggunkana fungsi bawaan python yaitu fungsi *print()* untuk menampilkan output, ada sebuah kasus bagaimana bila kita ingin menampilkan input menggunakan bahasa pemograman python agar program yang kita buat menjadi lebih interaktif, caranya yaitu dengan menggunakan fungsi *input()*, fungsi input juga merupakan fungsi bawaan dari Bahasa pemograman python yang membuat program yang kita buat menjadi lebih interaktif.

Contoh program dengan fungsi *input()* yang dikombinasikan dengan fungsi *output()*/*print()*:

```python
username = input("Isi nama anda:")
print("nama anda adalah: " + username)
```

```
Isi nama anda:
```

Apabila kita mengisi input dengan ‘wahyu’ maka output yang di hasilkan sebagai berikut :

```
1 username = input("Isi nama anda:")
2 print("nama anda adalah: " + username)
```

```
Isi nama anda:wahyu
nama anda adalah: wahyu
```

LATIHAN

LATIHAN 1: Membuat Program dengan List

Untuk memantapkan pemahaman, silahkan coba latihan berikut.

#Buat sebuah list untuk menyimpan kenalanmu

#Isi list sebanyak 5

#Tampilkan isi list indeks nomer 3

#Tampilkan panjang list

#Tampilkan semua teman dengan perulangan

LATIHAN 2:

Untuk memantapkan pemahaman, silahkan coba latihan berikut.

#Buat python untuk menjumlahkan dua bilangan dengan ekspresi input dan output pada format int.

LATIHAN 3:

Untuk memantapkan pemahaman, silahkan coba latihan berikut.

#Membuat Kalkulator Sederhana Menggunakan Python dengan (penjumlahan,pengurangan,perkalian,pembagian)