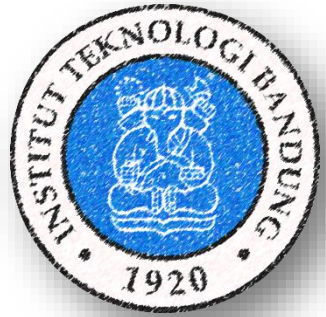


Pengulangan (Python)

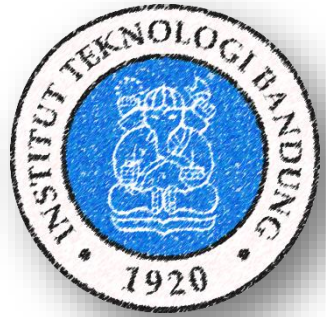
Tim Penyusun Materi Pengenalan Teknologi Informasi
Institut Teknologi Bandung © 2018





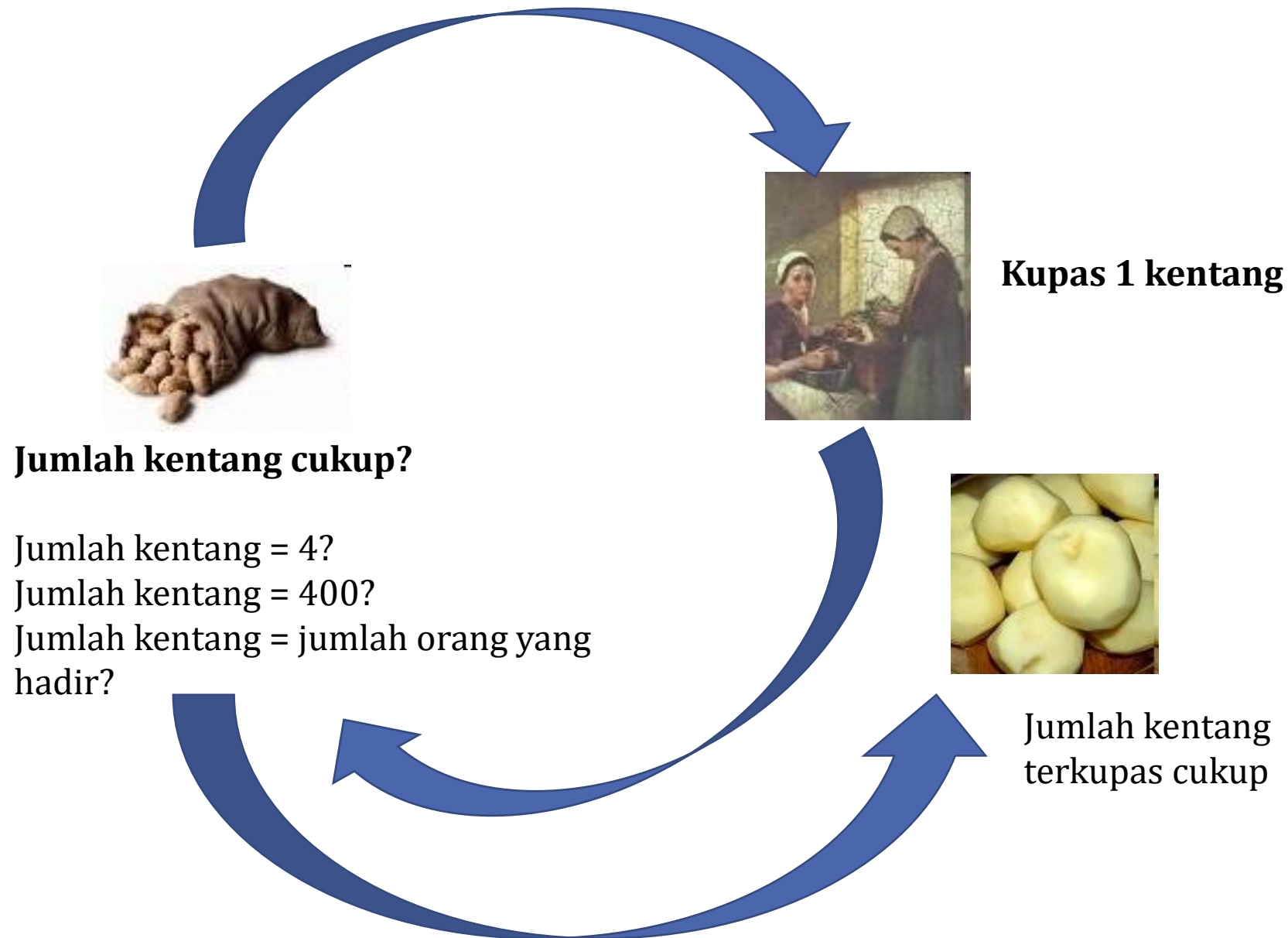
Tujuan

- Mahasiswa dapat menjelaskan jenis-jenis pengulangan dan penggunaannya serta elemen-elemen dalam pengulangan.
- Mahasiswa dapat menggunakan notasi pengulangan yang sesuai dengan benar.
- Mahasiswa dapat memanfaatkan jenis-jenis pengulangan dengan tepat dalam menyelesaikan persoalan sederhana yang diberikan.



Menyiapkan kentang untuk makan malam

- Asumsi: jumlah kentang tersedia tidak terbatas
- Pada suatu hari Ibu hanya mengupas kentang hanya 4 buah karena hanya anggota keluarga saja yang makan malam
- Pada hari yang lain, Ibu mengundang mahasiswa PTI-B sejumlah 400 orang untuk makan malam di rumahnya sehingga ibu mengupas 400 kentang untuk semua orang
- Hari yang lain, ibu tidak tahu berapa jumlah orang yang akan makan malam
 - Setiap selesai mengupas 1 kentang, dicek apakah jumlah cukup atau tidak

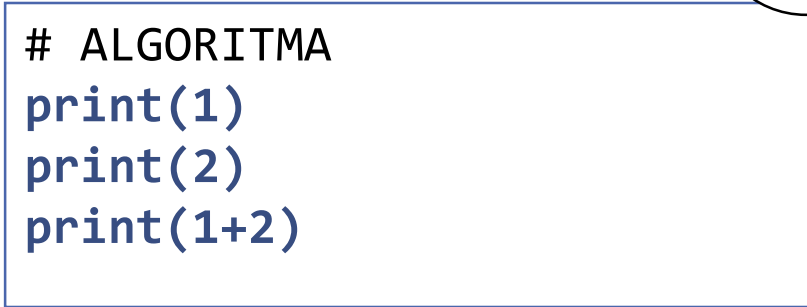


Menulis 1 dan 2

- Tuliskan program yang menuliskan angka 1 dan 2 dan selanjutnya 1+2 ke layar
- Contoh keluaran:



1
2
3




```
# ALGORITMA  
print(1)  
print(2)  
print(1+2)
```

Menulis 1 s.d. 3

- Tuliskan program yang menuliskan angka 1 s.d. 3 dan selanjutnya 1+2+3 ke layar
- Contoh keluaran:

1
2
3
6



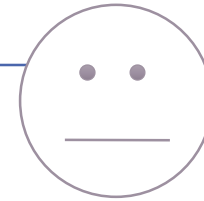
```
# ALGORITMA  
print(1)  
print(2)  
print(3)  
print(1+2+3)
```

Menulis 1 s.d. 10

- Tuliskan program yang menuliskan angka 1 s.d. 10 dan selanjutnya $1+2+3+\dots+10$ ke layar
- Contoh keluaran:

```
1
2
3
4
5
6
7
8
9
10
55
```


```
# ALGORITMA
print(1)
print(2)
print(3)
print(4)
print(5)
print(6)
... #lanjutkan sendiri!!
print(10)
print(1+2+3+4+5+6+7+8+9+10)
```



Menulis 1 s.d. 100

- Tuliskan program yang menuliskan angka 1 s.d. 100 dan selanjutnya $1+2+3+\dots+100$ ke layar
- Contoh keluaran:

```
1
2
3
4
5
6
7
8
9
10
... // lanjutkan sendiri!!
```



```
# ALGORITMA
print(1)
print(2)
print(3)
print(4)
print(5)
print(6)
... #lanjutkan sendiri!!
print(100)
print(1+2+3+4+5+6+7+8+9+10+ ... #lanjutkan sendiri!!)
```


Bagaimana kalau...

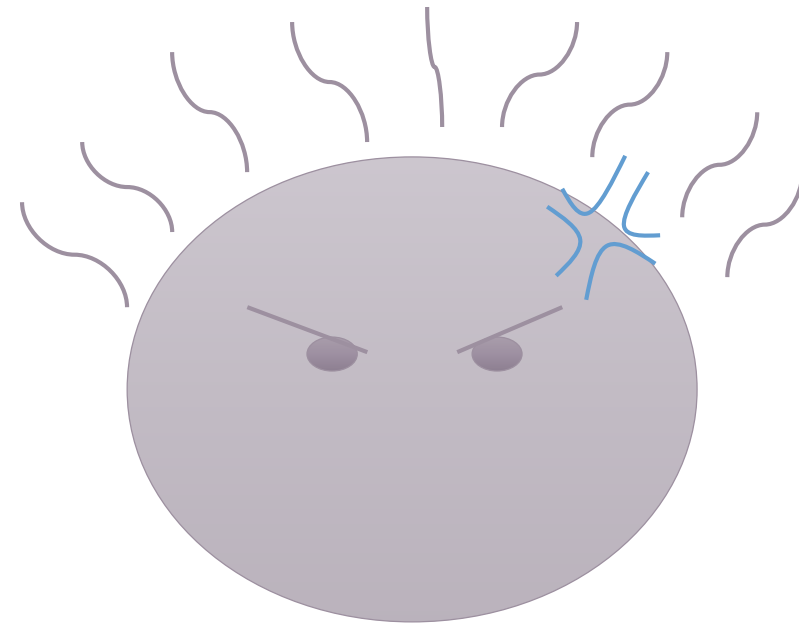
Anda diminta menulis dan menjumlahkan...

1 s.d. 1000 ???

1 s.d. 10000 ???

1 s.d. 1000000 ???

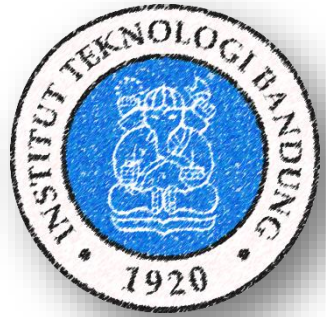
....



Pengulangan: Latar Belakang

- Melakukan suatu instruksi, bahkan aksi, secara berulang-ulang
 - Komputer: memiliki performansi yang sama
 - Manusia: punya kecenderungan untuk melakukan kesalahan (karena letih atau bosan)





Pengulangan (*Looping*)

- Elemen:
 - **Kondisi pengulangan:** ekspresi logik
 - **Badan pengulangan:** aksi yang diulang
- Jenis-jenis notasi pengulangan di Python:
 - Berdasarkan kondisi mengulang di awal: **while**
 - Berdasarkan pencacah: **for**

Contoh-1

- Tuliskan program yang menerima masukan sebuah integer misalnya N dan menuliskan angka 1, 2, 3, ... N dan menuliskan $1+2+3+\dots+N$ ke layar.
- Asumsikan $N > 0$.
- Contoh:

N = 1

Tampilan di layar:

1

1

N = 5

Tampilan di layar:

1

2

3

4

5

15

N = 10

Tampilan di layar:

1

2

3

4

5

6

7

8

9

10

55

Kondisi Mengulang di Awal (while)

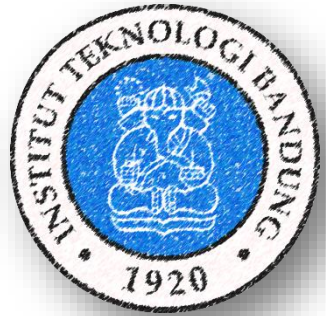
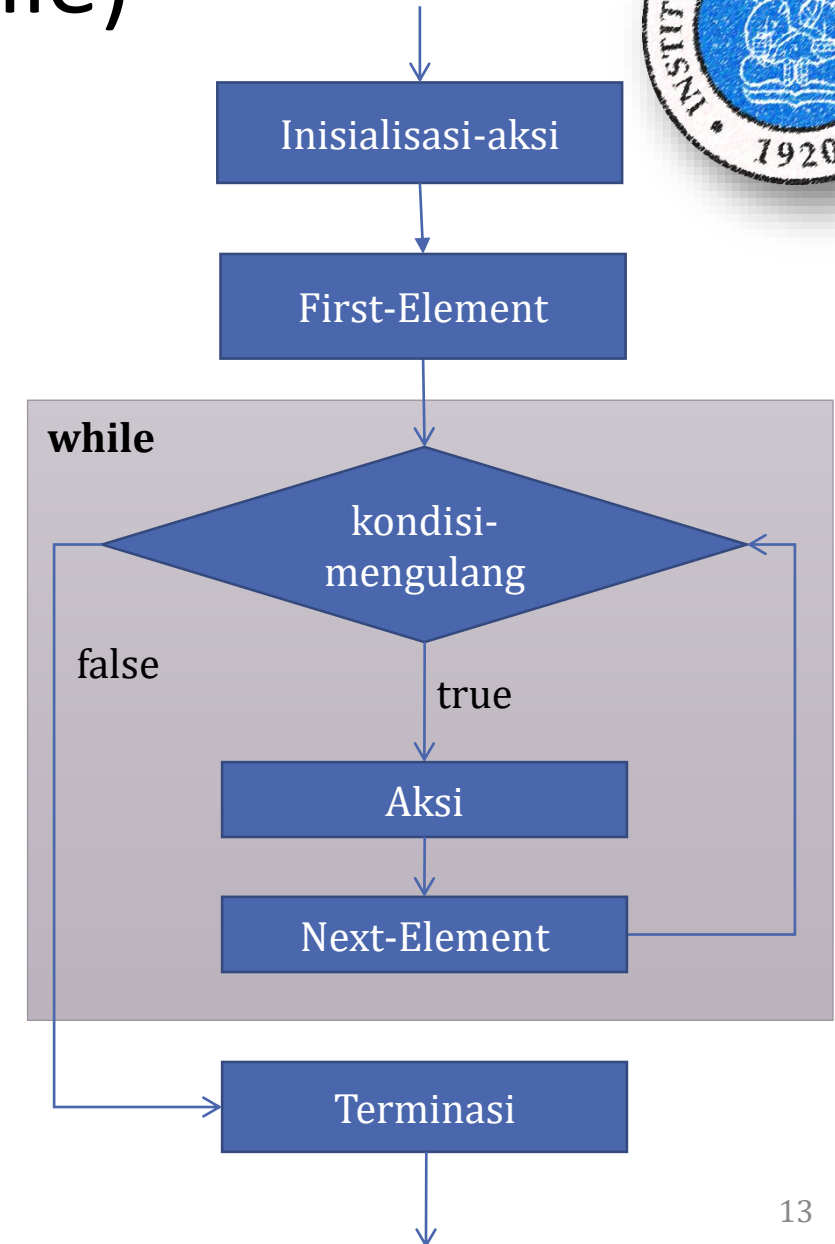
Python

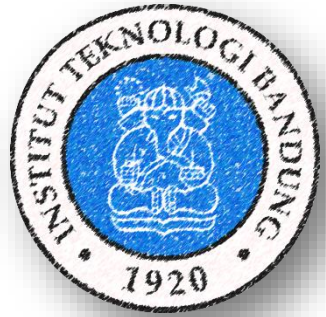
```
Inisialisasi-aksi  
First-Element  
while (kondisi-mengulang):  
    Aksi  
    Next-Element  
# kondisi-mengulang=false  
Terminasi
```

pseudocode

```
Inisialisasi-Aksi  
First-Element  
while (kondisi-mengulang) do  
    Aksi  
    Next-Element  
{ kondisi-mengulang = false }  
Terminasi
```

flowchart






Pengulangan Berdasarkan Kondisi Mengulang di Awal (**while**)

- **Aksi** akan dilakukan selama **kondisi-mengulang** masih dipenuhi (berharga **true**)
- Pengulangan ini berpotensi untuk menimbulkan **Aksi** “kosong” (Aksi tidak pernah dilakukan sama sekali)
 - Karena pada *test* yang pertama, **kondisi-mengulang** langsung tidak dipenuhi (berharga **false**) sehingga langsung ke luar *loop*

Contoh-1: while



```
# Program JumlahAngka
# Menghitung 1+2+3+...+N Asumsi N > 0

# KAMUS
# N : int
# i, sum : int

# ALGORITMA
N = int(input())           # Inisialisasi
sum = 0                    # Inisialisasi
i = 1                      # First-Element
while (i <= N):             # Kondisi-mengulang
    print(i)                # Aksi
    sum = sum + i           # Aksi
    i = i + 1               # Next-Element
# i > N
print(sum)                 # Terminasi
```

Berdasarkan Pencacah (for)

Python

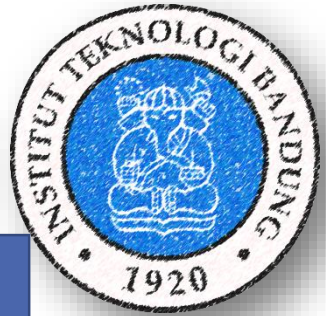
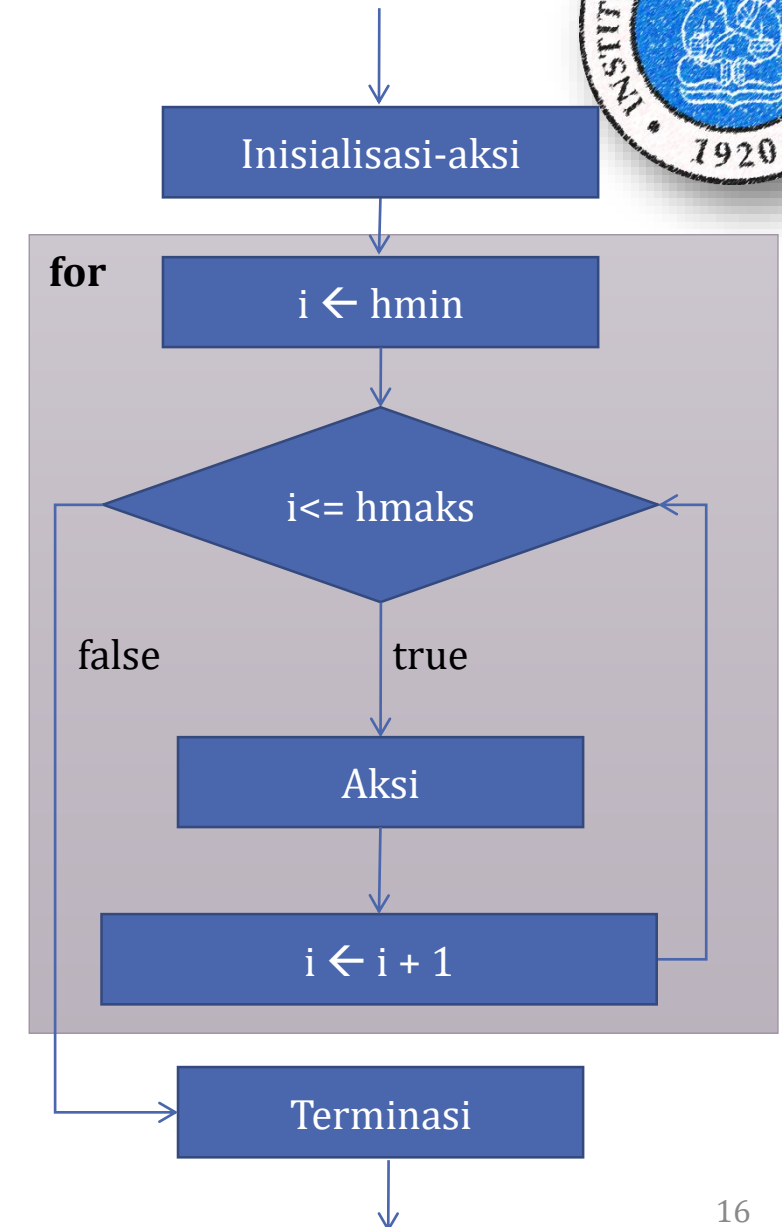
```
Inisialisasi-aksi  
for i in range(hmin, hmaks+1):  
    Aksi  
Terminasi
```

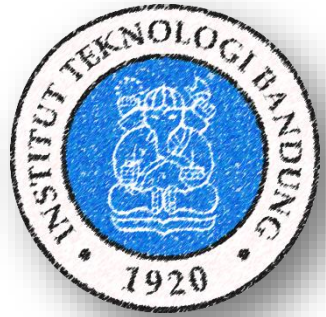
i adalah variabel pencacah (bisa diganti variabel lain)
hmin = nilai i di awal *loop*; *hmaks* = nilai i terakhir yang diproses;
nilai i ketika keluar *loop* adalah *hmaks*+1
Setiap berulang, i di-*increment* (ditambah 1)

pseudocode

```
Inisialisasi-aksi  
i traversal [hmin..hmaks]  
    Aksi  
Terminasi
```

flowchart





Berdasarkan Pencacah (for)

- Pengulangan dilakukan berdasarkan *range* harga suatu variabel *pencacah* (dalam contoh sebelumnya **i**)
 - *Range* harga pencacah yang diproses adalah dari **hmin** ke **hmaks**
- Pencacah harus suatu variabel dengan type yang terdefinisi suksesor dan predesesornya, misalnya integer
- **Aksi** akan dilakukan selama nilai pencacah masih berada dalam *range* yang ditentukan
- Harga pencacah di-*increment*, setiap kali **Aksi** selesai dilakukan
 - Karena itulah, nilai akhir *range* harus ditulis **hmaks+1** (agar **hmaks** tetap diproses)

Contoh-1: for



```
# Program JumlahAngka
# Menghitung 1+2+3+...+N. Asumsi N > 0

# KAMUS
# N : int
# i, sum : int

# ALGORITMA
N = int(input())      # Inisialisasi
sum = 0               # Inisialisasi

for i in range(1,N+1):
    print(i)           # Aksi
    sum = sum + i      # Aksi

print(sum)            # Terminasi
```

Mencacah Mundur

Python

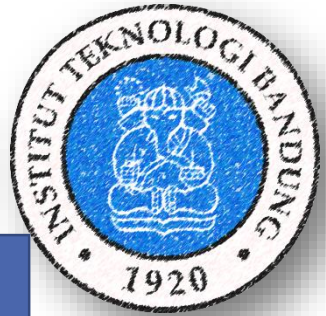
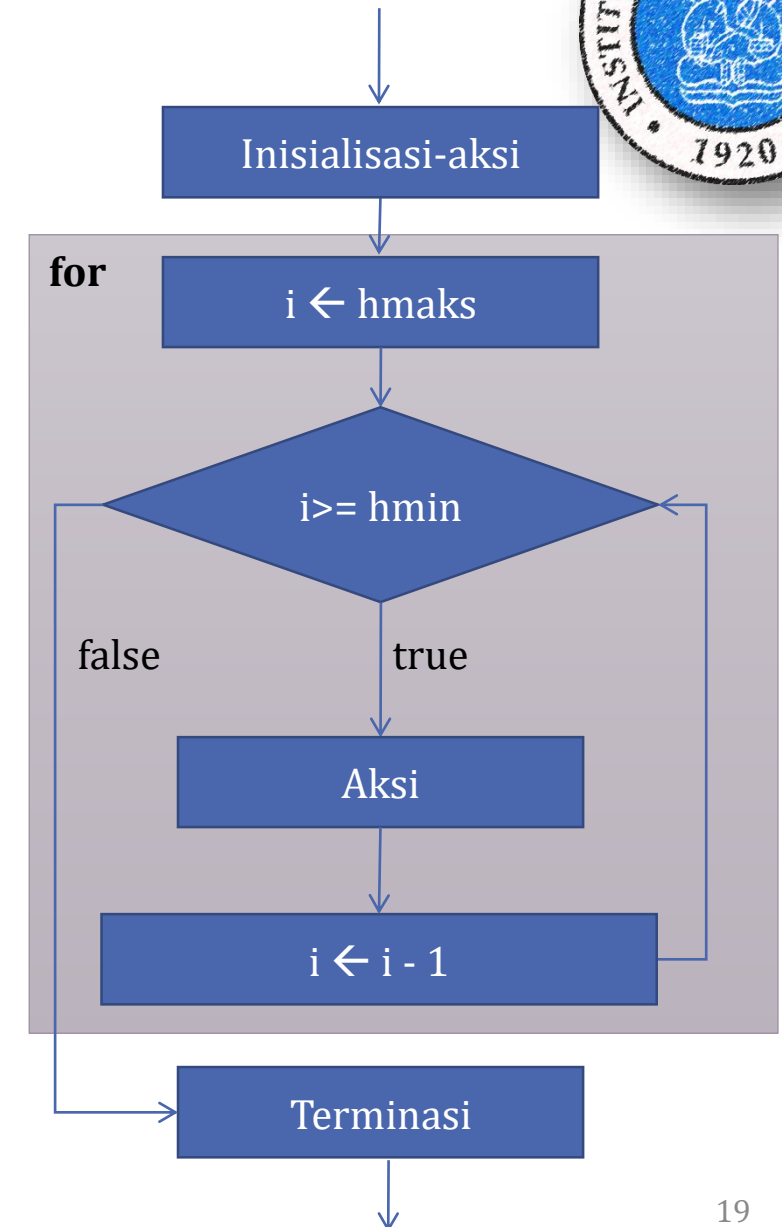
```
Inisialisasi-aksi  
for i in range(hmaks, hmin-1, -1):  
    Aksi  
Terminasi
```

i adalah variabel pencacah (bisa diganti variabel lain)
hmaks = nilai *i* di awal *loop*; *hmin* = nilai *i* terakhir yang diproses;
nilai *i* ketika keluar *loop* adalah *hmin*-1
-1: Setiap berulang, *i* di-*decrement* (dikurangi 1)

pseudocode

```
Inisialisasi-aksi  
i traversal [hmaks..hmin]  
    Aksi  
Terminasi
```

flowchart



Contoh-2

- Buatlah program yang menerima masukan 10 buah bilangan integer (dari keyboard) dan menuliskan ke layar jumlah total ke-10 integer tersebut.
- Contoh:

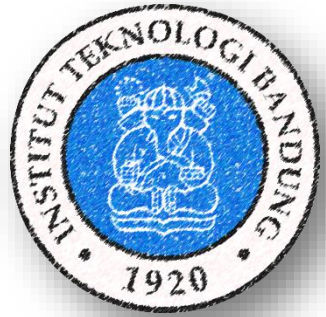
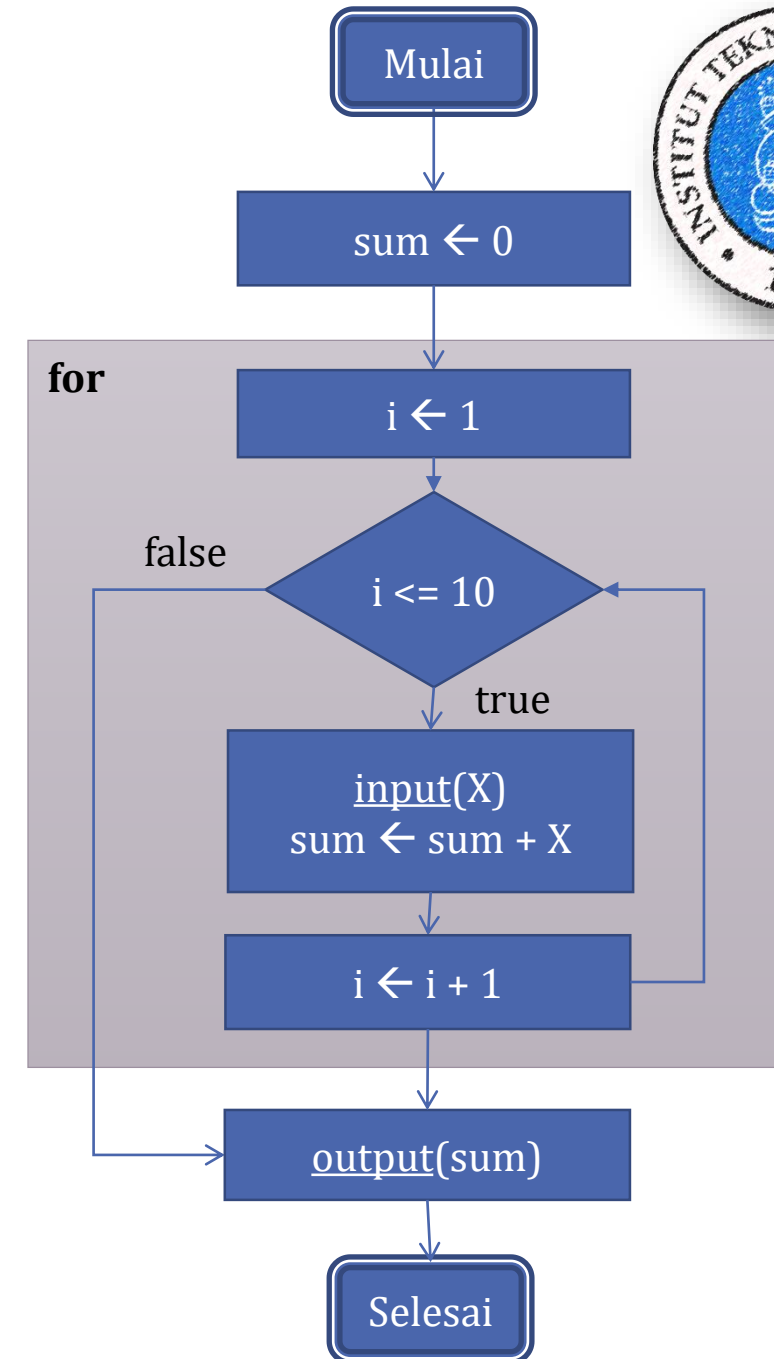
Masukan	Tampilan di Layar
2	18
1	
0	
-9	
7	
13	
2	
2	
1	
-1	

Contoh-2: for Pseudocode+Flowchart

pseudocode

```
sum ← 0 { inisialisasi }  
i traversal [1..10]  
    input(X)      { aksi }  
    sum ← sum + X { aksi }  
output(sum) { terminasi }
```

flowchart



Contoh-2: for Python

```
# Program Jumlah10Angka
# Menerima masukan 10 buah integer dan
# menjumlahkan totalnya

# KAMUS
# N, i, sum : int

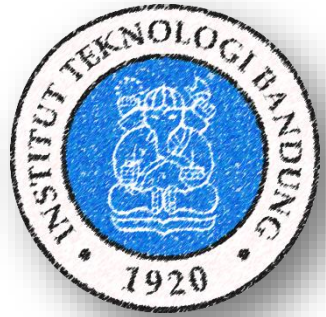
# ALGORITMA
sum = 0                                # Inisialisasi

for i in range(1, 11):
    N = int(input())                  # Aksi
    sum = sum + N                     # Aksi

print(sum)                            # Terminasi
```

Contoh-2: Diskusi

- Paling tepat menggunakan **for**:
 - Karena berapa kali Aksi harus diulang diketahui secara pasti, yaitu 10x → berarti *range* harga pencacah untuk pengulangan diketahui secara pasti, yaitu dari 1..10 (nilai terakhir pencacah ketika keluar *loop* = 11)
- Kurang tepat menggunakan **while** karena tidak ada kemungkinan kasus “kosong”
 - **while** lebih tepat digunakan jika ada kemungkinan Aksi tidak pernah dilakukan sama sekali (kasus kosong) → dalam hal ini, Aksi pasti dilakukan, minimum 1 kali



Contoh-3

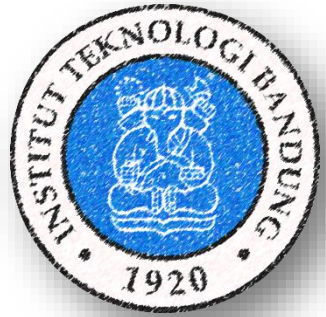
- Buatlah program yang membaca sejumlah bilangan integer dari keyboard sampai pengguna memasukkan angka -999 (angka -999 tidak termasuk bilangan yang diolah).
- Tuliskan berapa banyak bilangan yang dimasukkan, nilai total, dan rata-rata semua bilangan
- Jika dari masukan pertama sudah menuliskan -999, maka tuliskan pesan “Tidak ada data yang diolah”
- Petunjuk: Gunakan pengulangan **while**

No	Input	Output
1	<u>-1</u> <u>12</u> <u>-6</u> <u>10</u> <u>2</u> <u>-999</u>	Banyak bilangan = <u>5</u> Jumlah total = <u>17</u> Rata-rata = <u>3.40</u>
2	<u>-999</u>	<u>Tidak ada data yang diolah</u>

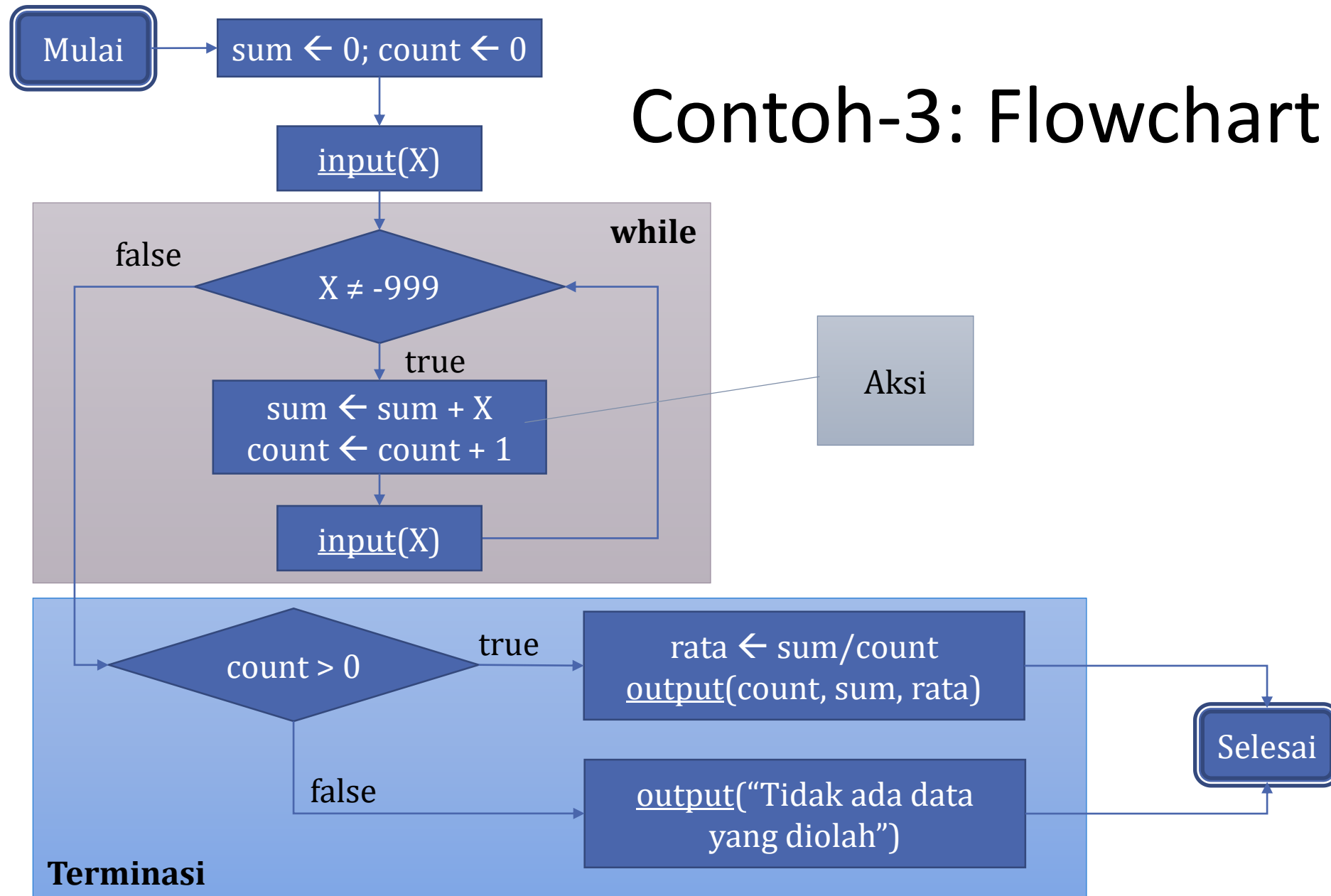
Contoh-3: Pseudocode

```
sum ← 0; count ← 0 { Inisialisasi }

output(count,sum,rata)
else { count = 0 }
    output("Tidak ada data yang diolah")
```



Contoh-3: Flowchart



Contoh-3: Python

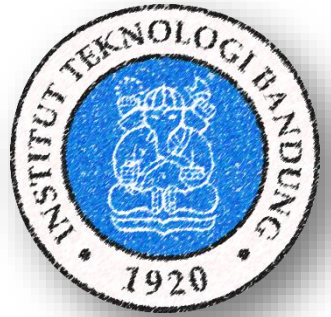
```
# Program RataBilangan
# Menerima masukan sejumlah bilangan integer sampai pengguna
# memasukkan -999 dan dan menampilkan banyak bilangan, total, dan
# rata-ratanya

# KAMUS
# X, count, sum : int
# rata : float

# ALGORITMA
sum = 0; count = 0      # Inisialisasi

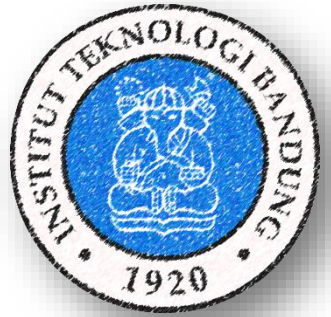
X = int(input())        # First-Elmt
while (X != -999):
    count = count + 1    # Aksi
    sum = sum + X
    X = int(input())     # Next-Elmt
# X = -999

# Terminasi
if (count > 0):
    print("Banyaknya bilangan = " + str(count))
    print("Jumlah total = " + str(sum))
    rata = sum/count
    print("Rata-rata = " + str(rata))
else:
    print ("Tidak ada data yang diolah")
```



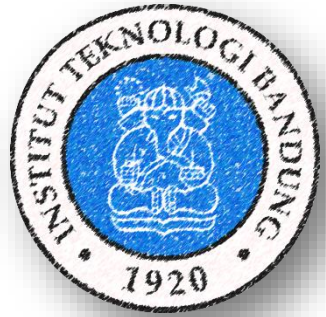
Contoh-3: Diskusi

- Pengulangan menggunakan **while** paling tepat karena:
 - Ada kemungkinan Aksi tidak pernah dilakukan sama sekali (kasus kosong), yaitu jika nilai X yang pertama kali dimasukkan user adalah -999 (lihat contoh ke-2)
- **For** tidak tepat digunakan karena tidak terdefinisi *range* nilainya



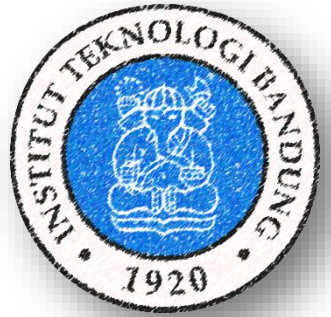
Latihan

- Untuk semua soal berikut, berlatihlah untuk membuat program Python dengan notasi pengulangan yang terbaik atau yang diminta
- Opsional: buatlah juga flowchart/pseudocode (tergantung yang diminta oleh dosen kelas)



Latihan-1

- Buatlah program yang membaca ada berapa banyak mahasiswa di kelas, misalnya N (Asumsi: $N > 0$, tidak perlu diperiksa)
- Selanjutnya, bacalah N buah character yang merepresentasikan nilai tugas KU1072. Nilai tugas yang mungkin adalah: 'A', 'B', 'C', 'D', 'E', 'F'. Asumsikan masukan nilai selalu benar.
- Jika mahasiswa mendapatkan nilai: 'A', 'B', 'C', atau 'D', maka mahasiswa dinyatakan lulus; sedangkan jika mendapat 'E' atau 'F' maka mahasiswa dinyatakan tidak lulus.
- Tuliskan ke layar berapa banyak mahasiswa yang lulus dan berapa yang tidak lulus.



Latihan-2

- Buatlah program untuk membaca sekumpulan bilangan bulat (integer) positif. Pembacaan data diakhiri jika pengguna memasukkan nilai negatif.
- Selanjutnya, cetaklah berapa banyak bilangan genap dan ganjil.
- 0 adalah bilangan genap.

Latihan-3: Lagu Anak Ayam

- Masih ingatkah dengan lagu Anak Ayam??

Anak ayam turunlah 5
Mati satu tinggalah 4
Mati satu tinggalah 3
Mati satu tinggalah 2
Mati satu tinggalah 1
Mati satu tinggal induknya

Anak ayam turunlah 1
Mati satu tinggal induknya

generalisasi

Anak ayam turunlah N
Mati satu tinggalah $N-1$
Mati satu tinggalah $N-2$
....
Mati satu tinggalah 1
Mati satu tinggal induknya

- Buatlah 2 versi program yang menerima masukan sebuah integer positif, misalnya N (asumsi $N > 0$), dan menuliskan lirik lagu Anak Ayam di atas dengan menggunakan perulangan **for** dan **while**.
 - Berikan komentar, apakah masing-masing jenis pengulangan tepat untuk persoalan ini.