

# WHILE LOOP

---

PERTEMUAN 11

# Apa Itu Perulangan While Pada Python?

Perulangan while pada python adalah proses pengulangan suatu blok kode program selama sebuah kondisi terpenuhi [\[1\]](#).

Singkatnya, perulangan while adalah perulangan yang bersifat *indefinite* alias tidak pasti, atau bahkan tidak terbatas [\[2\]](#).

Sebuah blok kode akan dilakukan terus-menerus selama suatu kondisi terpenuhi. Jika suatu kondisi **ternyata tidak terpenuhi** pada iterasi ke 10, maka **perulangan akan berhenti**. Jika kondisi yang sama pada saat yang berbeda ternyata berhenti pada iterasi ke 100, maka perulangan akan berhenti pada jumlah tersebut.

# Penulisan Sintaks While

Kita bisa menulis sintaks while dengan cara berikut:

```
while <kondisi>: # blok kode yang akan diulang-ulang
```

Terdapat 3 komponen utama:

- Yang pertama adalah keyword while, ini harus kita isi.
- Yang kedua adalah <kondisi>: ini bisa berupa variabel boolean atau ekspresi logika.
- Dan yang terakhir adalah blok (atau kumpulan baris) kode yang akan diulang-ulang **kondisi terpenuhi**.

# Perulangan Tanpa Batas

Perulangan while **sangat berkaitan** dengan variabel boolean, atau *logical statement*. Karena penentuan **kapan suatu blok kode akan diulang-ulang** ditinjau dari True or False dari suatu pernyataan logika.

Sehingga jika suatu kondisi itu selalu benar, maka perulangannya pun akan selalu di eksekusi.

Perhatikan contoh berikut:

```
while (1 + 2 == 3):  
    print('Halo Sahabat PTI!')
```

Jika dieksekusi, sistem akan mencetak tulisan “Halo Sahabat PTI!” berkali-kali tanpa henti.

Kita bisa **memaksanya berhenti dengan menekan tombol Ctr + C** jika menggunakan CLI, atau dengan cara menekan tombol stop jika menggunakan IDE atau sejenisnya.

# Kenapa perulangan di atas dieksekusi terus menerus?

Karena kita memerintahkan komputer untuk menulis “Halo Sahabat PTI!” selama satu ditambah dua sama dengan tiga.

Pertanyaannya: apakah satu ditambah dua sama dengan tiga terus-menerus atau tidak?

Jawabannya iya! Oleh karena itu sistem melakukan iterasi tak terbatas.

Nah, timbul pertanyaan.

---

Lalu bagaimana caranya agar kita **bisa memberhentikan** perulangan `while`?

---

Caranya gampang, kita buat kondisinya bersifat dinamis (alias bisa berubah-ubah).

Di dalam  
contoh  
berikut,

kita akan menampilkan  
angka 1 sampai dengan  
angka 5 menggunakan  
perulangan while.

```
i = 1  
while i <= 5:  
    print(i) i += 1
```

Kode program di atas akan  
menghasilkan output  
seperti berikut:

```
1  
2  
3  
4  
5
```



# Penjelasan

Pada kode program di atas, kita menginstruksikan sistem untuk:

- Melakukan perulangan selama variabel  $i$  kurang dari atau sama dengan 5.
- Setiap kali iterasi, sistem akan menampilkan nilai dari  $i$ .
- Dan yang terakhir, pada setiap iterasi, sistem akan menambahkan nilai  $i$  dengan angka 1.

Kalian bisa memodifikasi kode program di atas misalkan untuk:

menampilkan bilangan prima dari 1 sampai 100

menampilkan angka kelipatan 4 dari 1 sampai 100

menampilkan angka ganjil dari 1 sampai 27

dan sebagainya

## Contoh perulangan while untuk list

Untuk menampilkan semua item pada list, cara yang paling clean adalah dengan menggunakan metode for seperti yang telah kita bahas sebelumnya.

Meskipun begitu, kita tetap bisa menggunakan perulangan while untuk bermain-main dengan list.

Perhatikan  
contoh berikut:

```
listKota = [  
    'Jakarta', 'Surabaya', 'Depok', 'Bekasi',  
    'Solo', 'Jogjakarta', 'Semarang',  
    'Makassar'  
]  
  
# bermain index  
i = 0  
  
while i < len(listKota):  
    print(listKota[i])  
    i += 1
```

Jika  
dijalankan,  
hasilnya akan  
terlihat  
seperti ini:

Jakarta

Surabaya

Depok

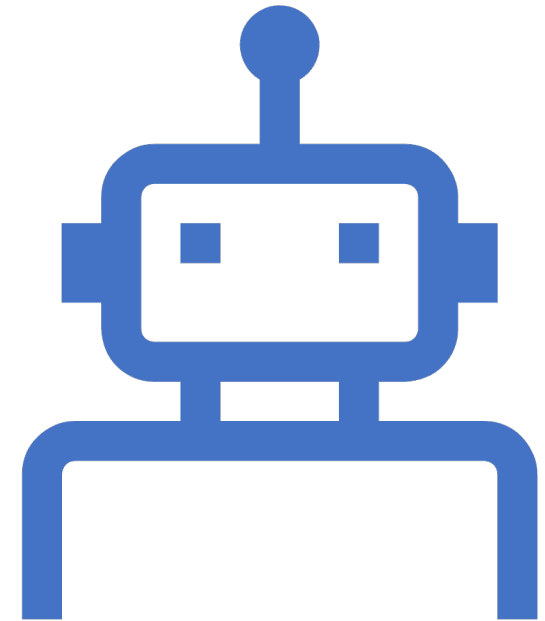
Bekasi

Solo

Jogjakarta

Semarang

Makassar



# Contoh perulangan while dengan inputan

- Kita juga bisa menggunakan while dengan inputan.
- Perhatikan contoh di bawah. Pada contoh ini kita akan meminta user untuk memasukkan angka ganjil lebih dari 50. Jika user justru memasukkan nilai genap atau nilai yang kurang dari 50, maka sistem akan meminta user untuk menginputkan kembali.

# Contoh

```
a = int(input('Masukkan bilangan ganjil lebih dari 50: `'))
```

```
while a % 2 != 1 or a <= 50:
```

```
    a = int(input('Salah, masukkan lagi: `'))  
    print('Benar')
```

## Output:

Masukkan bilangan ganjil lebih dari 50: 1

Salah, masukkan lagi: 2

Salah, masukkan lagi: 3

Salah, masukkan lagi: 10

Salah, masukkan lagi: 50

Salah, masukkan lagi: 52

Salah, masukkan lagi: 54

Salah, masukkan lagi: 55 Benar



# Contoh perulangan while dengan continue

Sama dengan perulangan for, kita juga bisa menggunakan perintah continue pada perulangan while.

Apa itu perintah continue?

Perintah continue berfungsi **untuk men-skip** iterasi sekarang ke iterasi selanjutnya.

## Contoh:

```
listKota = [  
    'Jakarta', 'Surabaya', 'Depok', 'Bekasi', 'Solo',  
    'Jogjakarta', 'Semarang', 'Makassar'  
]  
  
# skip jika i adalah bilangan genap  
# dan i lebih dari 0  
  
i = -1  
while i < len(listKota):  
    i += 1  
        if i % 2 == 0 and i > 0:  
            print('skip')  
            continue  
# tidak dieksekusi ketika continue dipanggil  
            print(listKota[i])
```

# Output:

Jakarta

Surabaya

skip <-- i sama dengan 2

Bekasi

skip <-- i sama dengan 4

Jogjakarta

skip <-- i sama dengan 6

Makassar

skip <-- i sama dengan 8

Pada output di atas, ketika i-nya adalah bilangan genap yang lebih dari satu, perintah `print(listKota[i])` tidak dieksekusi dan justru di-skip.

# Contoh perulangan while dengan break

Kita juga bisa menggunakan perintah break pada perulangan while.

Perintah break itu sebenarnya mirip dengan perintah continue.

Bedanya:

Ketika perintah break dipanggil, maka perulangan akan **dihentikan secara paksa**.

# Contoh.

---

```
listKota = [  
    'Jakarta', 'Surabaya', 'Depok', 'Bekasi', 'Solo',  
    'Jogjakarta', 'Semarang', 'Makassar'  
]  
  
kotaYangDicari = input('Masukkan nama kota yang dicari: ')  
  
i = 0  
while i < len(listKota):  
    if listKota[i].lower() == kotaYangDicari.lower():  
        print('Ketemu di index', i)  
        break  
  
    print('Bukan', listKota[i])  
    i += 1
```

# Outputnya.

```
Masukkan nama kota yang dicari: bekasi
```

```
Bukan Jakarta
```

```
Bukan Surabaya
```

```
Bukan Depok
```

```
Ketemu di index 3
```

**while ...  
else**

---

Sama seperti for, kita juga bisa menggunakan blok kode else pada perulangan while.

---

Tugasnya pun sama: yaitu untuk mendefinisikan suatu tugas yang akan dieksekusi ketika perulangan telah **selesai secara natural** tanpa dihentikan secara paksa.

Kita coba ubah program pencarian kota di atas dengan menambahkan blok kode else seperti berikut:

```
listKota = [  
    'Jakarta', 'Surabaya', 'Depok', 'Bekasi', 'Solo',  
    'Jogjakarta', 'Semarang', 'Makassar'  
]  
  
kotaYangDicari = input('Masukkan nama kota yang dicari: ')  
  
i = 0  
while i < len(listKota):  
    if listKota[i].lower() == kotaYangDicari.lower():  
        print('Ketemu di index', i)  
        break  
  
    print('Bukan ', listKota[i])  
    i += 1  
else:  
    print('Maaf, kota yang anda cari tidak ditemukan.')
```



Coba eksekusi lalu masukkan kota yang tidak ada pada variabel listKota.

```
Masukkan nama kota yang dicari: sidoarjo  
Bukan Jakarta  
Bukan Surabaya  
Bukan Depok  
Bukan Bekasi  
Bukan Solo  
Bukan Jogjakarta  
Bukan Semarang  
Bukan Makassar  
Maaf, kota yang anda cari tidak ditemukan.
```

Berbeda jika misal kota yang saya cari adalah kotak “Depok”:

Masukkan nama kota yang dicari: Depok

Bukan Jakarta

Bukan Surabaya

Ketemu di index 2

# Penjelasan

Di sini perintah yang ada di blok kode else tidak dieksekusi oleh sistem. Kenapa? Karena perulangannya diberhentikan secara paksa dengan perintah break, bukan karena berhenti secara natural.

# Next

---

For Loop