



EduAPM

PSE

LINGKUP  
PRIVAT

1/10

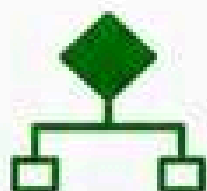
# Percabangan dan Perulangan

## Mengajarkan Logika kepada Komputer

Dalam Python, komputer bisa diajarkan untuk memilih tindakan dan mengulang pekerjaan. Inilah dasar logika program yang membuat kode menjadi cerdas dan terarah.



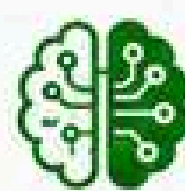
Kondisi



Keputusan



Pengulangan



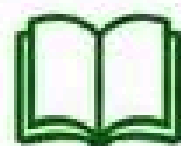
Logika



Aksi



Dua konsep inti yang perlu dipahami pemula: **percabangan** untuk memilih, **perulangan** untuk mengulang.



Pahami



Coba



Latih



Geser untuk mulai belajar  
percabangan dan perulangan.





# Apa Itu Percabangan?

## Program Memilih Berdasarkan Kondisi

Percabangan dipakai saat program harus menentukan aksi yang berbeda. Jika kondisi benar, jalankan satu perintah. Jika tidak, pilih aksi lain.



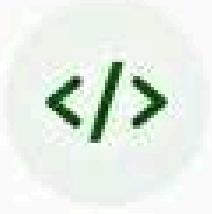
01



**Konsep**

Memeriksa  
kondisi

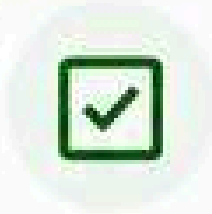
02



**Kata Kunci**

`'if'`

03



**Hasil**

Benar atau  
Salah

04



**Tujuan**

Memilih  
tindakan



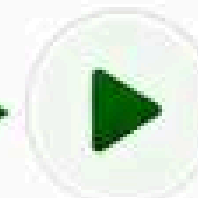
Kondisi



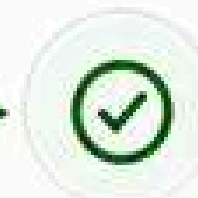
Cek



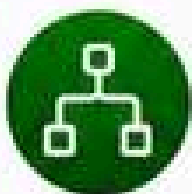
Putuskan



Jalankan



Selesai



Percabangan membantu  
program merespons data  
secara tepat.



Input



Evaluasi



Output



Lanjut untuk mengenal **if** dan **else**.





# Struktur if dan else

## Dua Arah Keputusan

Gunakan ``if`` saat kondisi benar.

Gunakan ``else`` untuk aksi cadangan saat kondisi tidak terpenuhi.



```
if nilai >= 75:  
    print("Lulus")  
else:  
    print("Belum Lulus")
```



Jika benar  
→ Lulus



Jika salah  
→ Belum Lulus



Input Nilai

01



Cek Kondisi

02



if

03



else

04



Tampilkan Hasil

05



Struktur sederhana ini sangat **sering dipakai** dalam program sehari-hari.



Jelas



Mudah



Praktis



Geser untuk melihat elif.





# Mengenal elif

## Saat Pilihan Lebih dari Dua

Jika program punya beberapa kemungkinan, gunakan `elif`. Program akan memeriksa kondisi dari atas ke bawah hingga menemukan yang sesuai.

`if` nilai  $\geq 85$  → Sangat Baik

`elif` nilai  $\geq 75$  → Baik

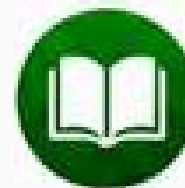
`else` → Perlu Belajar Lagi



**85 ke atas**  
Sangat Baik



**75–84**  
Baik



**di bawah 75**  
Perlu Belajar Lagi



Mulai  
01



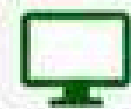
Cek 1  
02



Cek 2  
03



Pilih  
04



Tampilkan  
05



**Urutan kondisi penting.** Letakkan aturan yang lebih spesifik lebih dulu.



Urut



Teliti



Rapi



Selanjutnya: operator logika pada percabangan.





# Operator Logika and, or, dan not

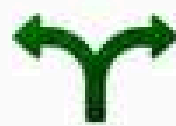
Percabangan sering memakai lebih dari satu syarat. Operator logika membantu program memeriksa hubungan antar kondisi.

and



Semua kondisi harus benar

or



Cukup salah satu benar

not



Membalik nilai kondisi

</>

```
lulus = nilai >= 75 and kehadiran >= 80
```



Syarat 1



Syarat 2



Gabung



Evaluasi



Hasil



Gunakan operator logika agar aturan program lebih nyata dan sesuai kebutuhan.



Gabungkan



Periksa



Putuskan



Kini saatnya belajar perulangan.





# Apa Itu Perulangan?

## Mengulang Tugas Secara Otomatis

Perulangan membuat komputer menjalankan instruksi yang sama berkali-kali tanpa harus menulis kode berulang.



**Tujuan:**  
Menghemat kode



**Manfaat:**  
Lebih cepat



**Dipakai untuk:**  
Data berulang



**Contoh:**  
Cetak angka



```
for angka in range(1, 6):  
    print(angka)
```



Mulai



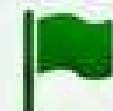
Ulangi



Proses



Lanjut



Selesai



Dua bentuk utama perulangan di Python adalah **for** dan **while**.



for



while



latihan



Geser untuk mengenal perulangan for.





# Perulangan *for*

Cocok untuk Jumlah Ulangan yang Jelas

'for' digunakan saat kita ingin memproses setiap item dalam daftar atau rentang angka yang sudah diketahui.

```
for angka in range(1, 6):  
    print(angka)
```

1 2 3 4 5

[...] range(1, 6)

menentukan rentang angka.

▶ mulai dari 1

angka pertama yang diambil.

■ berhenti sebelum 6

angka 6 tidak dieksekusi.

≡ cetak setiap putaran

kode di dalam blok dijalankan berulang.



Tentukan Rentang

01



Ambil Nilai

02



Jalankan Kode

03



Ulangi

04



Berhenti

05



for juga sering dipakai untuk membaca isi list, seperti daftar nilai atau nama.



Data



List



Urutan



Lanjut ke perulangan *while*.





# Perulangan while

## Berjalan Selama Kondisi Masih Benar

'while' cocok dipakai saat jumlah pengulangan belum pasti. Program akan terus berjalan selama syaratnya bernilai benar.

```
password = ""  
while password != "python123":  
    password = input("Masukkan password: ")
```



### Ubah nilai kondisi

Pastikan ada perubahan nilai di dalam perulangan agar kondisi bisa menjadi salah.



### Hindari infinite loop

Pastikan ada perubahan nilai di dalam perulangan agar kondisi bisa menjadi salah.



Jika kondisi tidak pernah berubah, perulangan bisa berjalan tanpa akhir.



### Cek Kondisi

Periksa apakah kondisi bernilai benar.



### Jalankan

Jalankan blok kode selama kondisi benar.



### Perbarui

Ubah nilai variabel agar kondisi bisa berubah.



### Cek Lagi

Kembali memeriksa kondisi setelah diperbarui.



### Berhenti

Berhenti jika kondisi menjadi salah (False).



**while** sangat berguna untuk validasi input dan proses yang menunggu kondisi tertentu.



Validasi



Kontrol



Aman



Berikutnya: **break** dan **continue**.





# break dan continue

## Mengontrol Jalannya Perulangan

`break` menghentikan loop lebih awal.

`continue` melewati satu putaran lalu lanjut ke putaran berikutnya.



### break

Berhenti saat data yang dicari sudah ditemukan



### continue

Lewati data tertentu tanpa menghentikan seluruh proses



```
for nilai in daftar:
```

```
    if nilai == 0: continue
```

```
    if nilai > 90: break
```



Loop

01



Cek

02



continue?

03



break?

04



Lanjut/Selesai

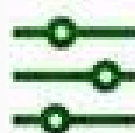
05



Percabangan sering dipakai di dalam perulangan untuk membuat alur program lebih pintar.



Saring



Atur



Kontrol



Geser untuk rangkuman akhir.





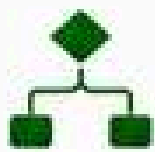
EduAPM

PSE LINGKUP PRIVAT

10/10

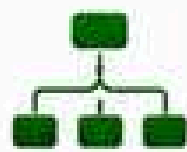
# Rangkuman Percabangan dan Perulangan

Percabangan membantu komputer memilih tindakan. Perulangan membantu komputer mengulang pekerjaan. Keduanya adalah fondasi penting untuk membangun logika program.



**if/else**

Memilih salah satu kondisi untuk dijalankan.



**elif**

Menambahkan kondisi tambahan secara berurutan.



**for**

Mengulang kode berdasarkan jumlah atau urutan.



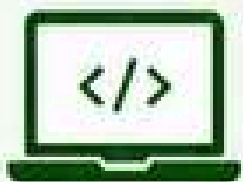
**while**

Mengulang kode selama kondisi masih terpenuhi.



**break & continue**

Mengontrol alur perulangan sesuai kebutuhan.



**Latihan kecil:**

buat program yang mengecek nilai dan menampilkan angka 1–5.



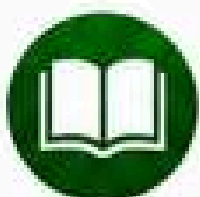
Pahami



Praktikkan



Kembangkan



Setelah memahami dasar ini, kamu siap belajar **fungsi**, **list**, dan **proyek Python** sederhana.



Siap belajar Python lebih lanjut bersama EduAPM?

