

**LAPORAN PRAKTIKUM
PEMROGRAMAN 1
MODUL IV
PERULANGAN**



RAFLI DHAFIN KAMIL

2211104018

S1SE06-A

**PRODI S1 REKAYASA PERANGKAT LUNAK
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2023**

IV. DASAR TEORI

Dalam pembuatan program, terkadang kita harus melakukan pengulangan suatu aksi misalnya untuk melakukan perhitungan berulang dengan menggunakan formula yang sama. Sebagai contoh, misalnya kita ingin membuat program yang dapat menampilkan sebuah teks *"Saya sedang belajar python"* sebanyak 5 kali, maka kita tidak perlu untuk menuliskan 5 buah statement melainkan kita hanya tinggal menempatkan satu buah statement ke dalam suatu struktur pengulangan. Dengan demikian program kita akan lebih efisien.

Sebagai gambaran bagi anda untuk dapat lebih memahami konsep perulangan, coba perhatikan kode program di bawah ini :

```
1  
2  
3 print("Saya Belajar Bahasa Pemrograman Python")  
4 print("Saya Belajar Bahasa Pemrograman Python")  
5 print("Saya Belajar Bahasa Pemrograman Python")  
6 print("Saya Belajar Bahasa Pemrograman Python")  
7 print("Saya Belajar Bahasa Pemrograman Python")
```

Output yang didapatkan adalah :

```
Saya Belajar Bahasa Pemrograman Python  
Saya Belajar Bahasa Pemrograman Python  
Saya Belajar Bahasa Pemrograman Python  
Saya Belajar Bahasa Pemrograman Python  
Saya Belajar Bahasa Pemrograman Python  
PS D:\Study & Work\Programming\Python>
```

Pada source code diatas sangat tidak efisien karena tidak menggunakan struktur perulangan di dalamnya.

Perulangan dalam dunia pemrograman adalah baris kode atau instruksi yang dieksekusi oleh komputer secara berulang-ulang sampai suatu kondisi tertentu terpenuhi.

Perbedaan percabangan dan perulangan:

- Jika percabangan, blok kode yang memenuhi kondisi tertentu hanya akan dieksekusi satu kali saja.
- Sedangkan perulangan, ia akan dilakukan seterusnya berulang-ulang dengan jumlah tertentu atau selama kondisi tertentu terpenuhi.

A. STRUKTUR FOR

For pada python lebih dikenal sebagai foreach. Struktur for ini digunakan untuk menuliskan jenis perulangan yang banyaknya sudah pasti atau telah diketahui sebelumnya. Oleh karena itu, disini kita harus melakukan inisialisasi nilai untuk kondisi awal pengulangan dan juga harus menuliskan kondisi untuk mengentikan proses pengulangan.

Pada praktikum ini saya mempelajari tiga jenis perulangan *for*, yaitu:

1) Range (max)

```
for i in range (7):  
    print("Hello Worudo ")
```

Maka output akan menampilkan perulangan sebanyak 7 kali :

```
Hello Worudo  
Hello Worudo  
Hello Worudo  
Hello Worudo  
Hello Worudo  
Hello Worudo  
Hello Worudo
```

Untuk `for` jenis pertama ini, kita masukan banyaknya perulangan yang ingin dilakukan ke dalam `range()`. Nilai variable `i` nantinya akan berubah, dimulai dari 0 hingga bilangan yang dimasukan ke `range()`, dan setiap perulangan bilangan tersebut akan dikurangi satu.

2) Range (min,max)

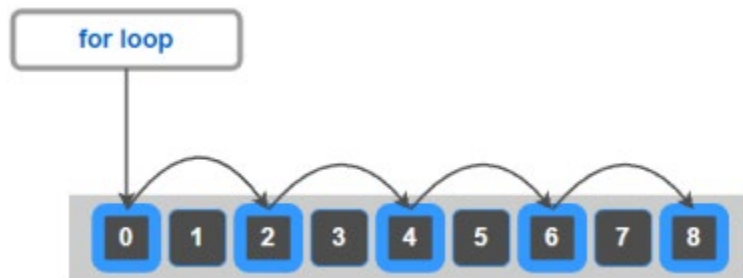
```
for i in range (1,8):  
    print("Hello Worudo ke - {i} ")
```

Hasil output akan menampilkan sebanyak yang diminta yaitu dari perulangan dengan sebanyak 8 kali karena kondisi awal adalah perulangan dari nilai 1

```
Hello Worudo ke - {i}  
Hello Worudo ke - {i}  
Hello Worudo ke - {i}  
Hello Worudo ke - {i}  
Hello Worudo ke - {i}  
Hello Worudo ke - {i}  
Hello Worudo ke - {i}
```

Pada perulangan ini counter `i` menyimpan nilai pada *range min* (nilai awal) adalah 1 dan *max* (batas akhir) adalah 10. Perlu diingat bahwa batas akhir selalu dikurang 1. Sehingga, perulangan yang dihasilkan adalah 1 sampai 7

Contoh diatas adalah perulangan *for* tanpa menggunakan **STEP**, berikut adalah ilustrasi jika perulangan *for* menggunakan **STEP**.



Dengan menggunakan **STEP** kita dapat mengatur berapa pertambahan di setiap perulangan yang dijalankan.

3) Range (min, max , step)

```
for i in range (1,8,2):  
    print(f"Hello Worudo ke - {i} ")
```

Output :

```
Hello Worudo ke - 1  
Hello Worudo ke - 3  
Hello Worudo ke - 5  
Hello Worudo ke - 7  
PS D:\Study & Work\Programming\Python> 
```

Pada struktur perulangan *for* ini counter *i* menyimpan nilai pada range *min* (nilai awal) adalah 0 dan *max* (nilai akhir) adalah 20, lalu pada code tersebut kita menambahkan *step* 2 untuk setiap perulangan.

Contoh lainnya kita akan membuat sebuah perulangan dengan decrement:

```
1 for i in range(10, 0, -1):
2     print(i)
```

Output:

```
PS C:\Users\Diva> &
10
9
8
7
6
5
4
3
2
1
```

B. STRUKTUR WHILE

Pada struktur pengulangan *while* kondisi akan diperiksa di bagian awal. Hal ini tentu menyebabkan kemungkinan bahwa apabila ternyata kondisi yang kita definisikan tidak terpenuhi (bernilai salah), maka proses pengulangan pun tidak akan pernah dilakukan. Bentuk umum dari struktur *while* :



Sebagai pembandingan dengan struktur pengulangan for, maka disini dituliskan kembali program yang akan menampilkan teks "Hallo World!" dengan menggunakan struktur while

```
1 i = 0
2 while i <= 7:
3     print("Hallo World!")
4     i += 1
```

Output :

```
PS C:\Users\Diva> &
Hallo World!
Hallo World!
Hallo World!
Hallo World!
Hallo World!
Hallo World!
Hallo World!
Hallo World!
Hallo World!
```

Perulangan decrement:

```
1 a = 10
2 b = 0
3 while a > b:
4     print("Kuota internet anda sisa", a, "GB")
5     a -= 1
```

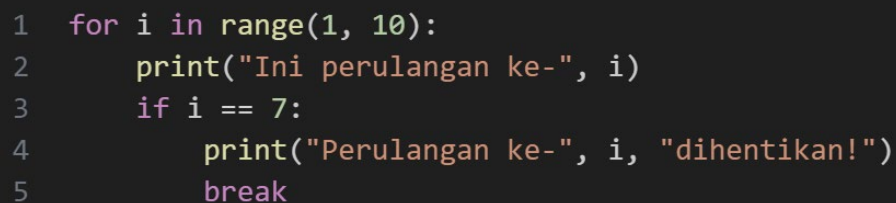
Output:

```
PS C:\Users\Diva> & C:/Users/Diva/AppData/
Kuota internet anda sisa 10 GB
Kuota internet anda sisa 9 GB
Kuota internet anda sisa 8 GB
Kuota internet anda sisa 7 GB
Kuota internet anda sisa 6 GB
Kuota internet anda sisa 5 GB
Kuota internet anda sisa 4 GB
Kuota internet anda sisa 3 GB
Kuota internet anda sisa 2 GB
Kuota internet anda sisa 1 GB
```

C. FUNGSI BREAK dan CONTINUE

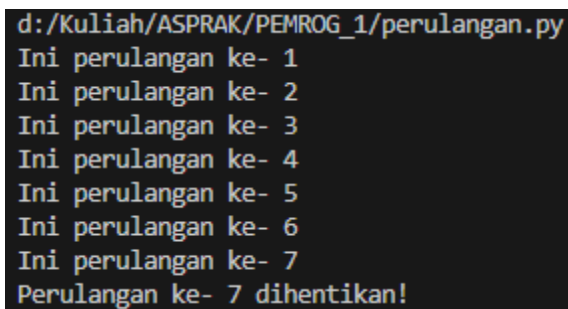
Pada python, kita bisa menginterupsi dan juga men-skip suatu iterasi pada perulangan. Terdapat 2 perintah yang bisa kita gunakan, yaitu: *break* untuk interupsi (memberhentikan paksa) sebuah perulangan dan *continue* untuk menskip ke iterasi selanjutnya

Contoh *break* pada perulangan *for*:

A screenshot of a code editor with a dark background and three colored window control buttons (red, yellow, green) at the top left. The code is a Python for loop that iterates from 1 to 10. It prints "Ini perulangan ke-" followed by the current value of i. When i reaches 7, it prints "Perulangan ke-" followed by i and "dihentikan!" and then breaks out of the loop.

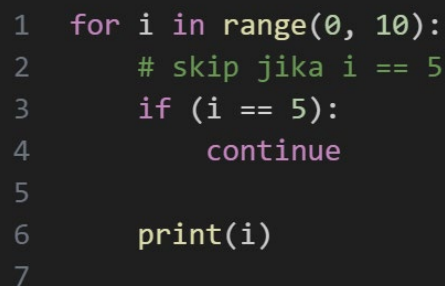
```
1 for i in range(1, 10):
2     print("Ini perulangan ke-", i)
3     if i == 7:
4         print("Perulangan ke-", i, "dihentikan!")
5         break
```

Output :

A screenshot of a terminal window showing the output of the Python code. The output consists of seven lines: "Ini perulangan ke- 1" through "Ini perulangan ke- 7", followed by "Perulangan ke- 7 dihentikan!".

```
d:/Kuliah/ASPRAK/PEMROG_1/perulangan.py
Ini perulangan ke- 1
Ini perulangan ke- 2
Ini perulangan ke- 3
Ini perulangan ke- 4
Ini perulangan ke- 5
Ini perulangan ke- 6
Ini perulangan ke- 7
Perulangan ke- 7 dihentikan!
```

Contoh *continue* pada perulangan *for*:

A screenshot of a code editor with a dark background and three colored window control buttons (red, yellow, green) at the top left. The code is a Python for loop that iterates from 0 to 10. It has a comment "# skip jika i == 5" and an if statement that checks if i is equal to 5. If it is, it uses the continue statement to skip the rest of the loop body and go to the next iteration. Otherwise, it prints the value of i.

```
1 for i in range(0, 10):
2     # skip jika i == 5
3     if (i == 5):
4         continue
5
6     print(i)
7
```


Output:

```
PS C:\Users\Diva> &
0
1
2
3
4
6
7
8
9
```

Contoh **break** pada perulangan *while*:

```
1 a = 0
2 while True:
3     print("Step ke-", a, "!")
4     a += 1
5     if a == 7:
6         print("Step ke-", a, "dihentikan!")
7         break
```

Output:

```
d:/Kuliah/ASPRAK/PEMROG_1/perulangan.py
Step ke- 0 !
Step ke- 1 !
Step ke- 2 !
Step ke- 3 !
Step ke- 4 !
Step ke- 5 !
Step ke- 6 !
Step ke- 7 dihentikan!
```

Contoh *continue* pada perulangan while:

```
1  angka = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
2
3  # skip jika i adalah bilangan genap
4  # dan i lebih dari 0
5  i = -1
6  while i < len(angka):
7      i += 1
8      if i % 2 == 0 and i > 0:
9          print('skip')
10         continue
11
12     # tidak dieksekusi ketika continue dipanggil
13     print(angka[i])
```

Output:

```
PS C:\Users\Diva> &
1
2
skip
4
skip
6
skip
8
skip
10
skip
```

D. PEMBAHASAN TUGAS

- 1) Buatlah sebuah program dengan statement perulangan dimana dapat menghitung total nilai dari suatu bilangan yang diinputkan. Dengan tampilan output sebagai berikut:

```
0/python.exe "d:/#1 ITTP/SEMESTER 2/Praktikum Pemrograman 1/Project
===== PROGRAM SEDERHANA MENGHITUNG JUMLAH TOTAL BILANGAN =====
Masukkan bilangan = 10
Total Nilai = 10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 = 55
```

**inputkan bilangan sesuai dengan dua angka terakhir nim kalian, jika yang depannya 0 cukup satu angka terakhir saja. (misal: 02, cukup input 2)*

Baris kode :

```
Python > Pertemuan 4 > Tugas1.py > ...
1  bill = int(input("Masukkan Angka Yang Ingin Di Looping : "))
2  jumlah = 0
3  samadengan = ""
4
5  while bill > 0:
6
7      jumlah += bill
8
9      samadengan += str(bill)
10     if bill != 1:
11         samadengan += " + "
12
13
14     bill -= 1
15
16  print("Hasil nya ",samadengan,"=",jumlah)
```

Output :

```
PS D:\Study & Work\Programming\Python> & C:/Users/dhafi/AppData/Local/Programs/Python/Python39/python.exe
Masukkan Angka Yang Ingin Di Looping : 18
Hasil nya  18 + 17 + 16 + 15 + 14 + 13 + 12 + 11 + 10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 = 171
PS D:\Study & Work\Programming\Python> |
```

- 2) Buatlah sebuah program dengan statement perulangan, dimana dapat menghitung hasil pangkat suatu bilangan. Dengan tampilan output sebagai berikut:

```
0/python.exe "d:/#1 ITTP/SEMESTER 2/Praktikum Pemrograman 1/
===== PROGRAM SEDERHANA MENGHITUNG PANGKAT =====
Masukkan bilangan = 36
Masukkan pencacah = 2
Hasil pangkat = 1296
```

**inputkan bilangan dengan 2 angka awal nim + 2 angka terakhir (misal: 21104015, maka $21+15 = 36$) dan pencacah inputkan nilai 2 saja.*

Baris Kode :

```
1  bilangan1 = int(input("masukkan 2 digit nim pertama : "))
2  bilangan2 = int(input("masukkan 2 digit nim belakang : "))
3
4  bilangan = bilangan1+bilangan2
5
6  print ("Jumlah : ",bilangan)
7
8
9  pencacah = 2
10
11  hasil = bilangan**pencacah
12
13  print("Pangkat : ",hasil)
```

Output :

```
masukkan 2 digit nim pertama : 22
masukkan 2 digit nim belakang : 18
Jumlah : 40
Pangkat : 1600
PS D:\Study & Work\Programming\Python> |
```

- 3) Buatlah sebuah program dengan statement perulangan untuk menentukan KPK dari dua buah bilangan bulat. Sebagai contoh KPK dari 8 dan 12 adalah 24.

```
0/python.exe "d:/#1 ITTP/SEMESTER 2/Praktikum Pemrograman 1/
===== PROGRAM SEDERHANA MENCARI KPK =====
Masukkan bilangan pertama = 8
Masukkan bilangan kedua = 12
KPK = 24
```

Baris Kode :

```
bil1 = int(input("Masukkan Bilangan pertama : "))
bil2 = int(input("Masukkan Bilangan kedua : "))

kpk = bil1

while kpk % bil2 != 0:
    kpk += bil1

print (f"KPK dari {bil1} dan {bil2} adalah {kpk}")
```

Output :

```
PS D:\Study & Work\Programming\Python> &
Masukkan Bilangan pertama : 10
Masukkan Bilangan kedua : 2
KPK dari 10 dan 2 adalah 10
PS D:\Study & Work\Programming\Python> [
```