

**LAPORAN PRAKTIKUM
PEMROGRAMAN 1
MODUL 3**



Oleh:

NAMA : GIDEON TORANAWA LADIYO

NIM : 2211104022

KELAS : SE-06A

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

I. 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.



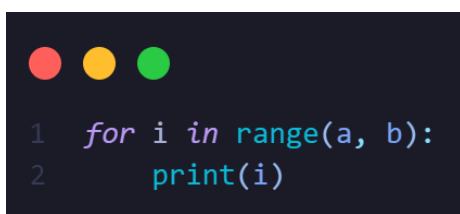
```
1 print("Saya belajar python")
2 print("Saya belajar python")
3 print("Saya belajar python")
4 print("Saya belajar python")
5 print("Saya belajar python")
```

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:

- Kalau 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.

STRUKTUR FOR

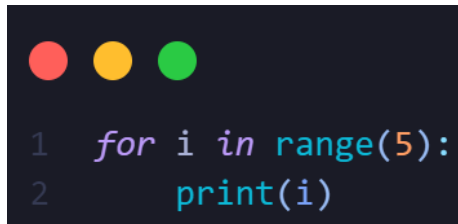
For pada python lebih dikenal sebagai foreach. Struktur for ini digunakan untuk menuliskan jenis perulangan yang banyaknya sudah pasti atau telah diketahui sebelumnya. Oleh karena itu, disini kita harus melakukan inisialisasi nilai untuk kondisi awal pengulangan dan juga harus menuliskan kondisi untuk mengentikan proses pengulangan. Adapun bentuk umum dari pendefinisian struktur for untuk satu statement adalah sebagai berikut :



```
1 for i in range(a, b):
2     print(i)
```

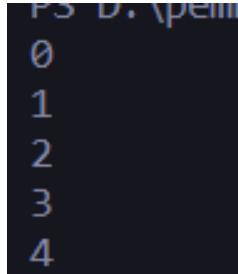
Pada praktikum ini kita akan belajar tiga jenis perulangan for, yaitu:

1. Range(max)

A screenshot of a Python code editor with a dark background. At the top, there are three colored circles: red, yellow, and green. Below them, the code is written in a syntax-highlighted font:

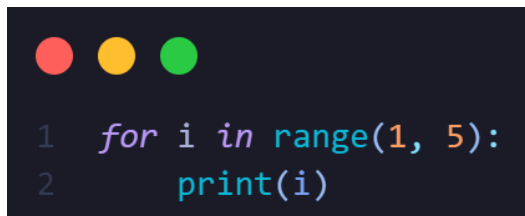
```
1 for i in range(5):  
2     print(i)
```

Output :

A screenshot of a terminal window showing the output of the first code snippet. The output consists of five lines, each containing a number from 0 to 4, printed vertically. The text is white on a dark background.

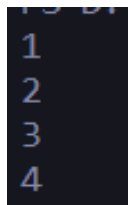
Untuk `for` jenis pertama ini, kita masukan banyaknya perulangan yang ingin dilakukan ke dalam `range()`. Nilai variable `i` nantinya akan berubah, dimulai dari 0 hingga bilangan yang dimasukan ke `range()`, dan setiap perulangan bilangan tersebut akan dikurangi satu.

2. Range(min,max)

A screenshot of a Python code editor with a dark background. At the top, there are three colored circles: red, yellow, and green. Below them, the code is written in a syntax-highlighted font:

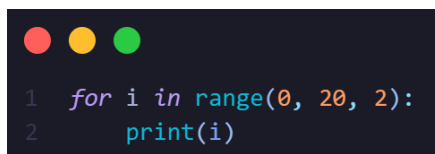
```
1 for i in range(1, 5):  
2     print(i)
```

Output :

A screenshot of a terminal window showing the output of the second code snippet. The output consists of four lines, each containing a number from 1 to 4, printed vertically. The text is white on a dark background.

Pada perulangan ini counter i menyimpan nilai pada range min (nilai awal) adalah 1 dan max (batas akhir) adalah 5. Perlu diingat bahwa batas akhir selalu dikurang Sehingga, hasil perulangan yang dihasilkan adalah 1 sampai 4.

3. Range(min,max,step)

A screenshot of a Python code editor with a dark background. At the top, there are three colored circles: red, yellow, and green. Below them, the code is written in a syntax-highlighted font:

```
1 for i in range(0, 20, 2):  
2     print(i)
```

Output :

```
0
2
4
6
8
10
12
14
16
18
```

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 menurun/decrement :

```
1 for i in range(20, 0, -2):
2     print(i)
```

Output :

```
20
18
16
14
12
10
8
6
4
2
```

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 :

```
1 a
2 b
3 while a < b:
4     print("Step ke-", a)
5     a += 1
```

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("Hello World!")
4     i += 1
```

Output :

```
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
```

Perulangan increment :

```
1 a = 1
2 b = 10
3 while a < b:
4     print("Step ke-", a)
5     a += 1
```

Output :

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

Perulangan decrement :

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

Output :

```
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
```

FUNGSI BREAK & 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* :

```
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 :

```
PS D:\pemrograman\prakti
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
```

Contoh *continue* pada perulangan *for*:

```
1 print("perulangan ke -", i, "dihentikan!")
2     break
```

Output :

```
PS D:\pe
0
1
2
3
4
5
6
8
9
```

Contoh *break* pada *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 :

```
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 *while* :

```
1 angka = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
2 i = -1
3 while i < len(angka):
4     i += 1
5     if i % 2 == 0 and i > 0:
6         print("skip")
7         continue
8     print(angka[i])
```

Output :

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

II. GUIDED

1. sistem login sederhana, dengan ketentuan jika user memasukan password yang salah dalam waktu 3x maka tidak dapat melakukan login kembali

```
1 username = "dee"
2 password = "awokawok"
3
4 login_att = 0
5
6 while login_att < 4:
7     user1 = input("masukkan username : ")
8     pass1 = input("masukkan password : ")
9
10    if login_att == 3:
11        print("Login gagal! silahkan coba lagi nanti!")
12        break
13
14    elif user1 == username and pass1 == password:
15        print(f"Selamat datang {user1}!")
16        break
17
18    else:
19        print("coba cek kembali, username dan password anda mungkin salah")
20        login_att += 1
```

Output :

```
masukkan username : as
masukkan password : as
coba cek kembali, username dan password anda mungkin salah
masukkan username : as
masukkan password : as
coba cek kembali, username dan password anda mungkin salah
masukkan username : as
masukkan password : as
coba cek kembali, username dan password anda mungkin salah
masukkan username : as
masukkan password : as
Login gagal! silahkan coba lagi nanti!
```

2. program mencari bilangan genap, dimana inputan berasal dari user adalah berupa range maksimal.

```
1 bil = int(input("masukkan bilangan maksimal : "))
2 i = 0
3 while i < bil:
4     print(i)
5     i += 2
6     if i == (bil):
7         break
```


Output :

```
masukkan bilangan maksimal : 30
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
```

3. program dengan menggunakan for untuk menentukan nilai factor dari persekutuan terbesar dari dua buah bilangan bulat. Sebagai contoh kita memasukkan dua buah bilangan bulat yaitu 12 dan 72, maka FPB dari kedua bilangan tersebut adalah 12.

```
1 def fpb(x, y):
2     if x > y:
3         terkecil = y
4     else:
5         terkecil = x
6     for i in range(1, terkecil + 1):
7         if ((x % i == 0) and (y % i == 0)):
8             fpb = i
9     return fpb
10
11 nilai1 = int(input("Masukkan bilangan pertama : "))
12 nilai2 = int(input("Masukkan bilangan kedua : "))
13 print("FPB =", fpb(nilai1, nilai2))
14 # def fpb(x, y):
```

Output :

```
Masukkan bilangan pertama : 12
Masukkan bilangan kedua : 72
FPB = 12
```

III. UNGUIDED

1. program dengan statement perulangan dimana dapat menghitung total nilai dari suatu bilangan yang diinputkan.

```

1 total = 0
2 bil = int (input("Masukan bilangan = "))
3 print("Total nilai =", end = ' ')
4
5 while bil >= 1:
6     print(bil, end = ' ')
7     if bil == 1:
8         print('=', end = ' ')
9     else:
10        print('+', end = ' ')
11    total += bil
12    bil -= 1
13 print(total)

```

Output :

```

total = 0
bil = int (input("Masukan bilangan = "))
print("Total nilai =", end = ' ')

while bil >= 1:
    print(bil, end = ' ')
    if bil == 1:
        print('=', end = ' ')
    else:
        print('+', end = ' ')
    total += bil
    bil -= 1
print(total)

```

2. program dengan statement perulangan, dimana dapat menghitung hasil pangkat suatu bilangan.

Bilangan = 2 angka awal NIM + 2 angka akhir NIM = 22 + 22 = 44

```

1 bilangan = int(input("Masukkan bilangan : "))
2 pangkat = int(input("Masukkan pencacah : "))
3 hasil = 1
4
5 for i in range(pangkat):
6     hasil *= bilangan
7
8 print("Hasil pangkat :", hasil)

```

Output :

```

Masukkan bilangan : 44
Masukkan pencacah : 2
Hasil pangkat : 1936

```

3. program dengan statement perulangan untuk menentukan KPK dari dua buah bilangan bulat.

```

1  def fpb(a, b):
2      if a > b:
3          terkecil = b
4      else:
5          terkecil = a
6      for i in range(1, terkecil + 1):
7          if ((a % i == 0) and (b % i == 0)):
8              fpb = i
9      return fpb
10
11 def kpk(a, b):
12     kpk = int(a * b / fpb(a, b))
13     return kpk
14
15 angka1 = int(input("Masukan bilangan pertama = "))
16 angka2 = int(input("Masukan bilangan kedua = "))
17 print("kpk= ", kpk(angka1, angka2))

```

Output :

```

Masukan bilangan pertama = 8
Masukan bilangan kedua = 12
kpk = 24

```