

**LAPORAN PRAKTIKUM**

**PEMROGRAMAN 1**

**MODUL IV**



Oleh:

Muhamad Luthfi Hamdani

2211104020

SE 06-A

**PRODI S1 REKAYASA PERANGKAT LUNAK**

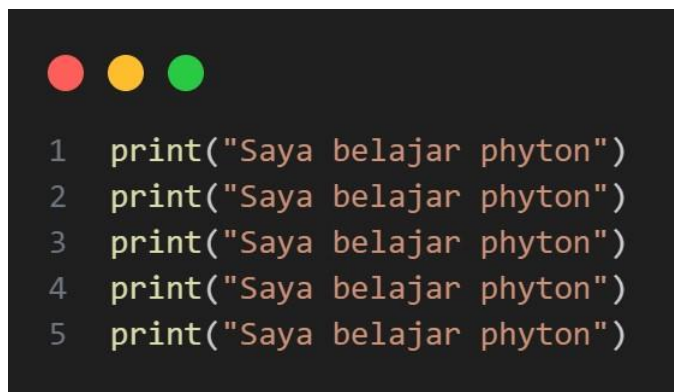
**FAKULTAS INFORMATIKA**

**INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

**2023**

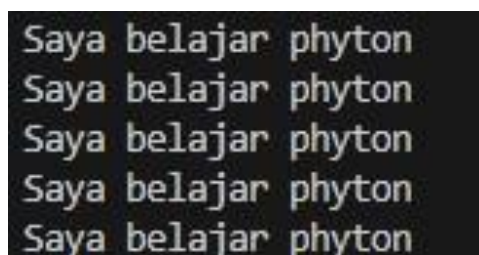
## I. DASAR TEORI

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



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

Output :



```
Saya belajar phyton
Saya belajar phyton
Saya belajar phyton
Saya belajar phyton
Saya belajar phyton
```

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

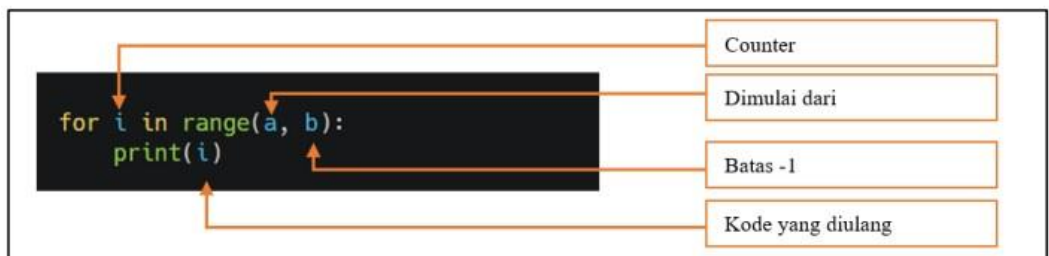
Perulangan dalam dunia pemrograman adalah baris kode atau instruksi yang dieksekusi oleh komputer secara berulang-ulang sampai suatu kondisi tertentu terpenuhi.

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.

### 3. 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 :



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

- a. Range (*max*)

```
for i in range(10):  
    print("Halo Dunia")
```

Output :

```
Halo Dunia  
Halo Dunia  
Halo Dunia  
Halo Dunia  
Halo Dunia  
Halo Dunia  
Halo Dunia  
Halo Dunia  
Halo Dunia  
Halo Dunia
```

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.

b. Range (*min,max*)

```
for i in range(1, 10):  
    print(f"perulangan ke- {i}")
```

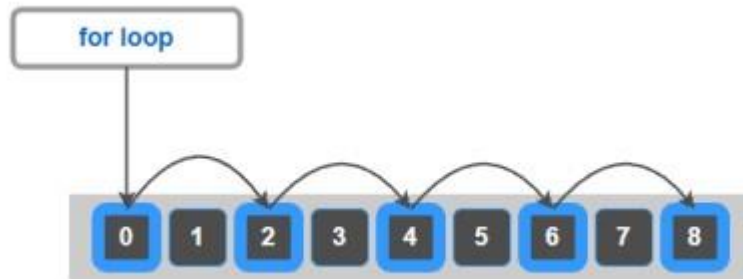
Output :

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

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, hasil perulangan yang dihasilkan adalah 1 sampai 9.

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.

c. Range (*min*, *max*, *step*)

```
for i in range(0, 20, 2):  
    print(f"step ke- {i}")
```

Output :

```
step ke- 0  
step ke- 2  
step ke- 4  
step ke- 6  
step ke- 8  
step ke- 10  
step ke- 12  
step ke- 14  
step ke- 16  
step ke- 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.

Source code:

```
for i in range(20, 0, -2):  
    print(f"step ke- {i}")
```

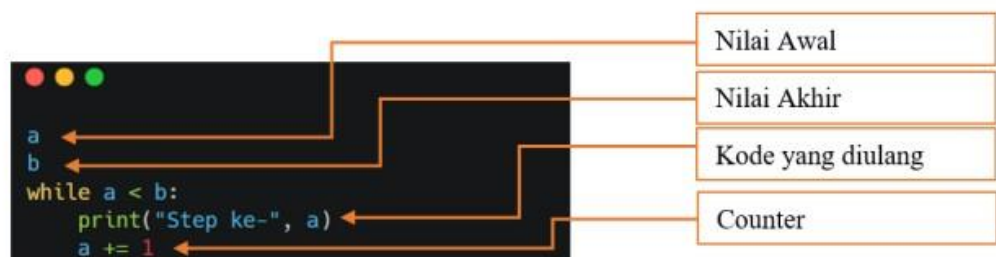
Output :

```
step ke- 20  
step ke- 18  
step ke- 16  
step ke- 14  
step ke- 12  
step ke- 10  
step ke- 8  
step ke- 6  
step ke- 4  
step ke- 2
```

#### 4. 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 pembanding dengan struktur pengulangan for, maka disini dituliskan kembali program yang akan menampilkan teks “Hallo World!” dengan menggunakan struktur while.

```
i = 0
while i <= 7:
    print("hello world!")
    i += 1
```

Output:

```
hello world!
hello world!
hello world!
hello world!
hello world!
hello world!
hello world!
hello world!
```

Perulangan increment:

```
a = 1
b = 10
while a < b:
    print("step ke-", a)
    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:

```
a = 10  
b = 0  
while a > b:  
    print("Kuota Internet tersisa", a, "GB")  
    a -= 1
```

Output :

```
Kuota Internet tersisa 10 GB  
Kuota Internet tersisa 9 GB  
Kuota Internet tersisa 8 GB  
Kuota Internet tersisa 7 GB  
Kuota Internet tersisa 6 GB  
Kuota Internet tersisa 5 GB  
Kuota Internet tersisa 4 GB  
Kuota Internet tersisa 3 GB  
Kuota Internet tersisa 2 GB  
Kuota Internet tersisa 1 GB
```

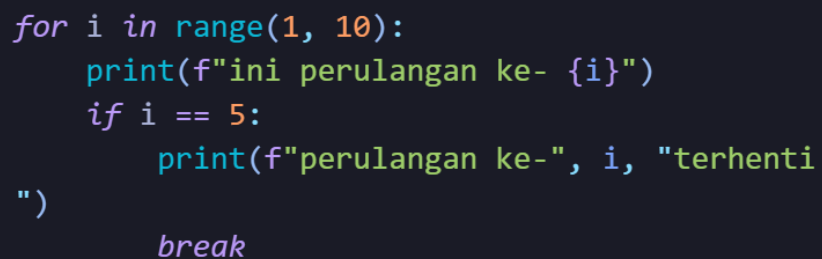
## 5. FUNGSI BREAK & CONTINUE

Pada python, kita bisa menginterupsi dan juga men-skip suatu iterasi pada perulangan.



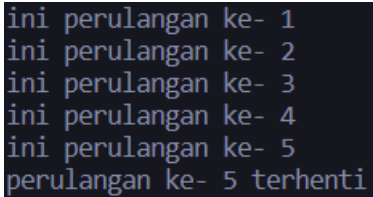
Terdapat 2 perintah yang bisa kita gunakan, yaitu: **break** untuk interupsi (memberhentikan paksa) sebuah perulangan dan **continue** untuk menskip ke iterasi selanjutnya.

Contoh **break** pada perulangan *for*:

A code editor window with a dark background and three colored window control buttons (red, yellow, green) at the top left. It contains Python code for a for loop that prints iteration numbers and breaks at iteration 5.


```
for i in range(1, 10):  
    print(f"ini perulangan ke- {i}")  
    if i == 5:  
        print(f"perulangan ke-", i, "terhenti")  
        break
```

Output:

A terminal window showing the output of the Python code. It displays five lines of iteration messages, with the fifth line including the word 'terhenti' (stopped).

```
ini perulangan ke- 1  
ini perulangan ke- 2  
ini perulangan ke- 3  
ini perulangan ke- 4  
ini perulangan ke- 5  
perulangan ke- 5 terhenti
```

Contoh **continue** pada perulangan *for*:




```
for i in range(0, 10):  
    #skip jika i == 5  
    if(i == 5):  
        continue  
    print(i)
```

Output :

```
0  
1  
2  
3  
4  
6  
7  
8  
9
```

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



```
a = 0  
while True:  
    print("step ke-", a, "!")  
    a += 1  
    if a == 7:  
        print("step ke-", a, "dihentikan!")  
        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 perulangan while:

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

Output :

```
1
2
skip
4
skip
6
skip
8
skip
10
skip
```

## II. GUIDED

### A. sistem login sederhana

```
print("=====LOGIN=====")

count = 0

while count < 3:
    username = input('Enter your username: ')
    password = input('Enter your password: ')

    if username == 'luthfi' and password == '123123':
        print('Login Berhasil.')
        break
    else:
        print('Password salah, silahkan coba lagi.')
        count += 1

else:
    print("Akun anda terblokir")

print("=====")
```

Output :

-jika username dan password benar

```
=====LOGIN=====
Enter your username: luthfi
Enter your password: 123123
Login Berhasil.
=====
```

-jika username dan password salah lebih dari 3x

```
=====LOGIN=====
Enter your username: asd
Enter your password: asd
Password salah, silahkan coba lagi.
Enter your username: asd
Enter your password: asd
Password salah, silahkan coba lagi.
Enter your username: asd
Enter your password: asd
Password salah, silahkan coba lagi.
Akun anda terblokir
=====
```

**B.** program mencari bilangan genap

```
bil = int(input("masukkan bilangan maksimal : "))
i = 0
while i < bil:
    print(i)
    i += 2
    if i == bil:
        break
```

Output :

```
masukkan bilangan maksimal : 20
0
2
4
6
8
10
12
14
16
18
```

C. program dengan menggunakan for untuk menentukan nilai factor dari persekutuan terbesar dari dua buah bilangan bulat.

```
print("====PROGRAM MENCARI FPB====")

#mendefinisikan fungsi
def hitung_fpb(x,y):

    #memilih bilangan yang paling kecil
    if x > y:
        terkecil = y
    else:
        terkecil = x
    for i in range(1, terkecil+1):
        if((x % i == 0) and (y % i == 0)):
            fpb = i
    return fpb

nilai = int(input("masukkan bilangan pertama : "))
nilai2 = int(input("masukkan bilangan kedua : "))
print("FPB = ",hitung_fpb(nilai, nilai2))
```

Output :

```
=====PROGRAM Mencari FPB=====
masukkan bilangan pertama : 12
masukkan bilangan kedua : 102
FPB = 6
```

### III. UNGUIDED

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

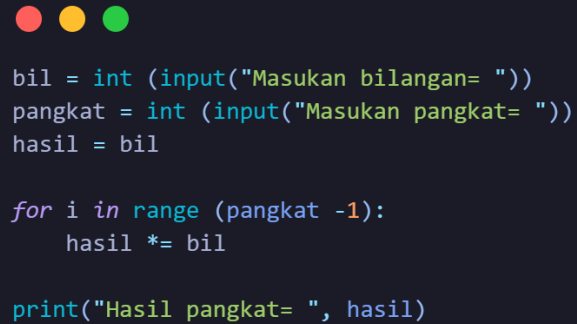
```
total = 0
bil = int (input("Masukan bilangan= "))
print("Total nilai= ", end = ' ')

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

Output :

```
Masukan bilangan= 20
Total nilai= 20 + 19 + 18 + 17 + 16 + 15 +
14 + 13 + 12 + 11 + 10 + 9 + 8 + 7 + 6 + 5
+ 4 + 3 + 2 + 1 = 210
```

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



```

bil = int (input("Masukan bilangan= "))
pangkat = int (input("Masukan pangkat= "))
hasil = bil

for i in range (pangkat -1):
    hasil *= bil

print("Hasil pangkat= ", hasil)

```

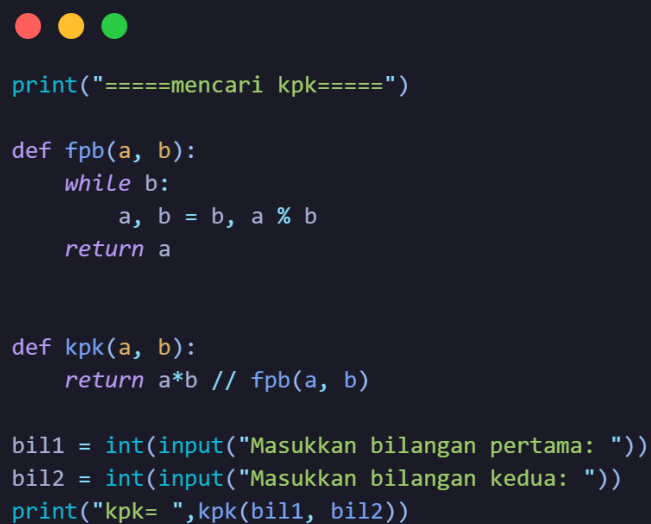
Output :

```

Masukan bilangan= 22
Masukan pangkat= 20
Hasil pangkat= 705429498686404044207947776

```

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



```

print("====mencari kpk====")

def fpb(a, b):
    while b:
        a, b = b, a % b
    return a

def kpk(a, b):
    return a*b // fpb(a, b)

bil1 = int(input("Masukkan bilangan pertama: "))
bil2 = int(input("Masukkan bilangan kedua: "))
print("kpk= ",kpk(bil1, bil2))

```



Output :

```
====mencari kpk====  
Masukkan bilangan pertama: 22  
Masukkan bilangan kedua: 20  
kpk= 220
```