MODUL IV
STATEMENT
PERULANGAN

5.1 TUJUAN

1. Mengimplementaasikan struktur perulangan dalam bahasa Python
2. Menerapkan sintaks – sintaks pengulangan dalam menyelesaikan persoalan
3. Mampu menyelesaikan persoalan tentang perulangan

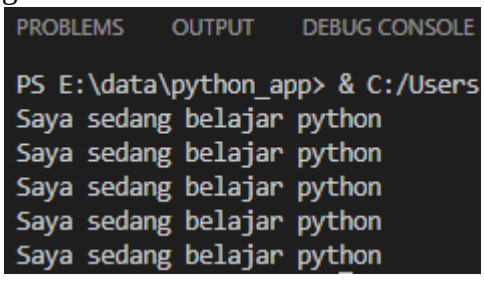
5.2 DASAR TEORI

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 :

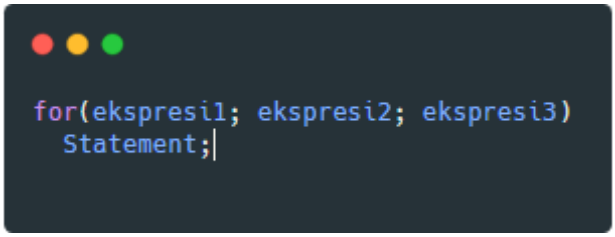
```
print('Saya sedang belajar python')
print('Saya sedang belajar python')
print('Saya sedang belajar python')
print('Saya sedang belajar python')
print('Saya sedang belajar python')
```

Output Program :



5.3 Struktur For

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 dalam bahasa java adalah sebagai berikut :



```
for(ekspresi1; ekspresi2; ekspresi3)
  Statement;
```

Sedangkan for untuk dua statement atau lebih dalam bahasa java adalah sebagai berikut :

```
for(ekspresi1; ekspresi2; ekspresi3) {
  Statement;
  Statement;
  ...
}
```

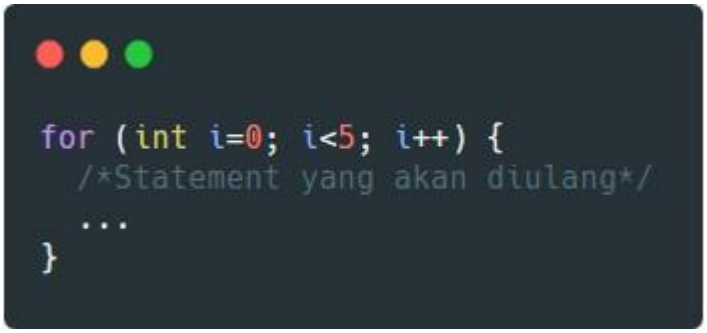
Keterangan :

Ekspresi 1 ⟶ digunakan sebagai proses inisialisasi variable yang akan dijadikansebagai pencacah(counter) dari proses pengulangan, dengan kata lain ekspresi ini akan dijadikan sebagai kondisi awal.

Ekspresi 2 ⟶ digunakan sebagai kondisi akhir, yaitu kondisi dimana proses pengulangan harus dihentikan. Perlu untuk diketahui bahwa pengulangan masih akan dilakukan selama kondisi akhir bernilai benar.

Ekspresi 3 ⟶ digunakan untuk menaikkan (increment) atau menurunkan(decremenet) nilai variable yang digunakan sebagai pencacah. Apabila pengulangan yang kita lakukan bersifat menaik, maka kita akan menggunakan statement increment, sedangkan apabila pengulangan yang akan kita gunakan bersifat menurun maka kita harus menggunakan statement decrement.

Secara umum contoh untuk mengilustrasikan struktur pengulangan for adalah sebagai berikut:



```
for (int i=0; i<5; i++) {
  /*Statement yang akan diulang*/
  ...
}
```

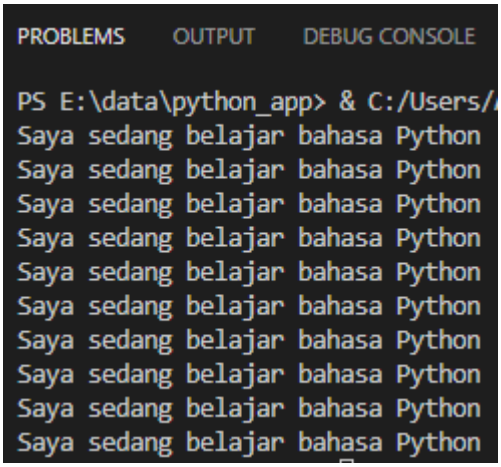
Pada sintaks di atas, mula – mula kita menginisialisasi variable i dengan nilai 0, kemudian karena ekspresi (0<5) bernilai benar maka program akan melakukan statement untuk pertama kalinya. Setelah itu variable i akan dinaikkan nilainya sebesar 1 melalui statement increment i++ sehingga nilai i sekarang menjadi 1. Sampai disni program akan mengecek ekspres (i<5). Oleh karena ekspresi (2<5) bernilai benar, maka program akan melakukan statement yang kedua kalinya. Begitu seterusnya sampai nilai i bernilai 4. Namun pada saat variable i telah bernilai 5 maka program akan keluar dari proses pengulangan. Hal ini disebabkan karena ekspresi (5<5) bernilai salah.

Untuk membuktikan hal tersebut, perhatikan contoh program di bawah ini dimana kita akan menampilkan teks “Saya sedang belajar bahasa Python” sebanyak 10 kali.

Kode Program :

```
for i in range(0,10):
    print('Saya sedang belajar bahasa Python')
```

Output Program :



```
PROBLEMS    OUTPUT    DEBUG CONSOLE

PS E:\data\python_app> & C:/Users/
Saya sedang belajar bahasa Python
Saya sedang belajar bahasa Python
Saya sedang belajar bahasa Python
Saya sedang belajar bahasa Python
Saya sedang belajar bahasa Python
Saya sedang belajar bahasa Python
Saya sedang belajar bahasa Python
Saya sedang belajar bahasa Python
Saya sedang belajar bahasa Python
Saya sedang belajar bahasa Python
```

5.4 Struktur While

Pada struktur pengulangan jenis ini 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 :

```
while (ekspresi) {
  Statement_yang_akan_diulang1;
  Statement_yang_akan_diulang2;
  ...
}
```

Sebagai pembanding dengan struktur pengulangan for diatas, maka disini dituliskan kembali program yang akan menampilkan teks “Saya sedang belajar Python” dengan menggunakan struktur while. Adapun sintaksnya adalah sebagai berikut.

Kode program :

```
i = 0
while i <= 5:
    print('Saya sedang belajar Python')
    i+=1
```

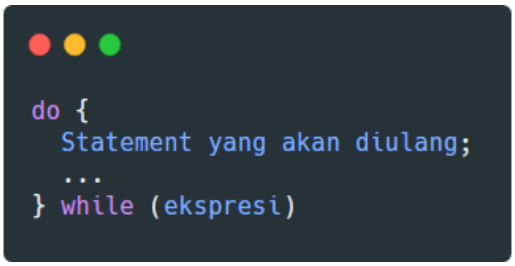
Output Program :

```
PROBLEMS    OUTPUT    DEBUG CONS

PS E:\data\python_app> & C:/Us
Saya sedang belajar Python
Saya sedang belajar Python
Saya sedang belajar Python
Saya sedang belajar Python
Saya sedang belajar Python
Saya sedang belajar Python
```

5.5 Struktur Do-While

Berbeda dengan struktur while dimana kondisinya terletak di awal blok pengulangan, pada struktur do-while kondisi diletakkan di akhir blok pengulangan. Hal ini menyebabkan bahwa statement yang terdapat di dalam blok pengulangan ini pasti akan dieksekusi minimal satu kali, walaupun kondisinya bernilai salah sekalipun. Maka dari itu struktur do-while ini banyak digunakan untuk kasus – kasus pengulangan yang tidak mempedulikan benar atau salahnya kondisi pada saat memulai proses pengulangan. Adapaun bentuk umum dari struktur pengulangan do- while adalah seperti yang tertulis di bawah ini :



```
do {
  Statement yang akan diulang;
  ...
} while (ekspresi)
```

Mungkin bagi anda yang baru mengenal bahasa pemrograman Python akan merasa bingung untuk membedakan struktur pengulangan while dan do-while. Dimana cara penggunaan do-while pada python sedikit berbeda dengan bahasa pemrograman lain. Untuk memahami perbedaanya perhatikan dua buah contoh program berikut ini.

Kode program pada javascript:

```
var quota = 9;

// perulangan while-do
while(quota > 0){
    console.log("Masih ada quota, bisa");
    quota--;
}

// perulangan do-while
do{
    console.log("Quota habis!");
    quota--;
} while(quota > 0)
```

Output program javascript:

```
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Quota habis!
```

Kode program python:

```python
quota = 8

# Perulangan while-do
while(quota > 0):
    print("Masih ada quota, bisa internetan!")
    quota = quota - 1

#perulangan do-while
while(True):
    print("Quota habis!")
    quota = quota - 1
    # periksa bensin
    if(quota < 0):
        break
```

Output program python:

```
PROBLEMS 10   OUTPUT   DEBUG CONSO

PS E:\data\python_app> & C:/Users/A
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Masih ada quota, bisa internetan!
Quota habis!
```

Kapankah Waktu yang Tepat Menggunakan while dan do/while?

Tergantung dari kasusnya. Bila syarat perulangannya tidak berkaitan dengan hasil hitung pada blok kode yang diulang, maka pakailah while. Tetapi, bila syarat perulangannya berkaitan dengan hasil perhitungan di blok kode yang diulang, maka pakailah do/while.

5.6 LATIHAN

1. Kasus

Membuat program dengan menggunakan statement for untuk membedakan perulangan yang menaik dan perulangan interval. Program akan memiliki output sebagai berikut :

```
PERULANGAN MENURUN
1
2
3
4
5

PERULANGAN DENGAN INTERVAL
2
4
6
8
```

Solusi :

```
print('PERULANGAN MENURUN')
for i in range(1,6):
    print(i)

print('')
print('PERULANGAN DENGAN INTERVAL')
for i in range(2,10,2):
    print(i)
```

2. Kasus :

Membuat 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 8 dan 12, maka FPB dari kedua bilangan tersebut adalah 4. Untuk lebih jelasnya perhatikan table di bawah ini :

Bilangan	Faktor
8	1,2,**4**,8
12	1,2,3,**4**,6,12

Program tersebut mempunyai output sebagai berikut :

```
Masukan bilangan pertama: 8
Masukan bilangan kedua: 12
Faktor Persekutuan Terbesar = 4
```

Solusi :

```
# 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

nilai1 = int(input("Masukan bilangan pertama: "))
nilai2 = int(input("Masukan bilangan kedua: "))

print("Faktor Persekutuan Terbesar =", hitung_FPB(nilai1, nilai2))
```

5.6 TUGAS

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

```
Masukkan bilangan : 5
Total Nilai = 5 + 4 + 3 + 2 + 1 = 15
```

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

```
Masukkan bilangan   : 2
Masukkan pencacah   : 3
Hasil pangkat       : 8
```

C. Buatlah sebuah program dengan statement perulangan untuk menentukan KPK dari dua buah bilangan bulat. Sebagai contoh KPK dari 8 dan 12 adalah 24. Untuk lebih jelasnya perhatikan table KPK di bawah ini :

8	16	**24**	32	40	48	…
12	**24**	36	48	60	72	…

Program tersebut mempunyai output sebagai berikut :

```
Masukkan bilangan pertama   : 12
Masukkan bilangan kedua     : 8
KPK : 24
```