conditionLooping

August 26, 2022

Nama : Muhammad Reesa Rosyid

Program : Python for Data Science

1 Sesi 2 - Conditions, Control Flow & Looping

1.1 Conditional

Dalam program Python, if statement adalah bagaimana kalian melakukan desicion making. If statement memungkinkan eksekusi kondisional dari pernyataan atau kelompok pernyataan berdasaekan nilai ekspresi.

1.1.1 Pengenalan pada statement if

Bentuk dasar pada if:

if (expr):

```
<statement>
```

Berikut contoh code menggunakan if statement:

1.1.2 Indentasi pada if

Indentasi digunakan untuk mendefinisikan pernyataan atau block. Dalam Program Python, pernyataan yang menjorok ke tingkat sama dianggap sebagai bagian dari block yang sama.

Pernyataan dasar majemuk if akan seperti ini:

if (expr):

```
<statement>

<statement>

...

<statement>
```

Berikut ini merupakan contoh penerapanya:

[2]:
```
x = 0
y = 5

if x < y:
    print('yes')  #Truthy

if y < x:
    print('yes')  #Falsy

if x:
    print('yes')  #Falsy
    #Dianggap false karena x = 0. 0 berarti false

if y:
    print('yes')  #Truthy
    #Dianggap truth karena selain bilangan 0 maka dianggap truth

if 'aul' in 'rgrault':
    print('yes')  #Truthy

if'quux' in ['foo', 'bar', 'baz']:
    print('tyes')  #Falsy
```

```
yes
yes
yes
```

[3]:
```
#Apakah line tersebut dieksekusi?

if 'foo' in ['foo', 'bar', 'baz']: #yes
    print('Outer condition is true') #yes

    if 10>20: #yes
        print('inner condition 1') #no

    print('Between inner condition') #yes

    if 10<20: #yes
        print('inner condition 2') #yes

    print('End of outer condition') #yes
print('After outer condition') #yes
```

```
Outer condition is true
Between inner condition
inner condition 2
End of outer condition
After outer condition
```

1.1.3 Ketentuan penggunaan else dan elif

Else dan elif(else if) digunakan untuk mengevaluasi kondisi dan mengambil satu jalur jika benar, tetapi menentukan jalur alternatif lain bila tidak benar.

Jika benar, rangkaian pertama dijalankan, dan rangkaian kedua dilewati. Jika salah maka rangkaian kedua akan dijalankan. Eksekusi ditentukan oleh indentasi seperti contoh di atas.

Berikut ini bentuk dasar dari if else:

if (expr):

    <statement(x)>

else:

    <statement(y)>

Berikut ini contoh penggunaan pada if else:

[8]:
```
hargaBuku = 20000
hargaMajalah = 5000
uang = 2000

if uang > hargaBuku:
    print("beli buku")
else:
    print("uang tidak cukup")
```

```
uang tidak cukup
```

Terdapat sintaks untuk percabangan berdasarkan beberapa alternatif. Dalam kasus ini perlu menggunakan sintaks elif. Rangkaian akan bergantian dievaluasi dan apabila tidak ada yang benar maka else yang akan dieksekusi.

Contoh dasar menggunakan if, elif, dan else:

if (expr):

    <statement(w)>

elif (expr):

    <statement(x)>

elif (expr):

    <statement(y)>

else:

    <statement(z)>

Pada penerapan contoh kasus:

[9]:
```
hargaBuku = 20000
hargaMajalah = 5000
```

```
uang = 15000

if uang > hargaBuku:
    print("beli buku")
elif uang > hargaMajalah:
    print("beli majalah")
else:
    print("uang tidak cukup")
```

```
beli majalah
```

1.2 Looping

Iteration artinya mengeksekusi block kode yang sama berulang kali, berpotensi bisa berkali-kali. Struktur pemrograman yang mengimplementasikan iterasi disebut loop.

Dalam pemrograman, ada dua jenis iterasi, indefinite dan definite: * Dengan indefinite iteration, beberapa kali loop dijalankan tidak ditentukan secara eksplisit sebelumnya. Sebaliknya, blok yang ditunjuk dijalankan berulang kali selam beberapa kondisi terpenuhi. * Dengan definite iteration, berapa kali blok ditunjuk akan dieksekusi ditentukan secara eksplisit pada saat perulangan dimulai.

1.2.1 While loop

Format loop ditampilkan di bawah ini:

while (expr):

    <statement(s)>

Contoh penerapan:

[1]:
```
i = 0
while i < 10:
    i += 1
    print(f"i ke {i}", "halo dunia")
```

```
i ke 1 halo dunia
i ke 2 halo dunia
i ke 3 halo dunia
i ke 4 halo dunia
i ke 5 halo dunia
i ke 6 halo dunia
i ke 7 halo dunia
i ke 8 halo dunia
i ke 9 halo dunia
i ke 10 halo dunia
```

Inilah yang terjadi dalam contoh diatas: * n adalah 10. Ekspresi dalam while pada baris 2 adalah n>0. Yang mana benar sehingga dieksekusi. Pasa baris 3, n+1 menjadi 1 lalu di print ke layar * Ketika badan loop selesai, eksekusi program kembali ke atas pada baris 2, dan ekspresi dievaluasi lagi, ekspresinya masih benarm jadi badan dieksekusi lagi dan di cetak 2. * Looping berlanjut

sampai n menjadi 10, ketika ekspresi diuji menghasilkan salah maka loop berhenti. Eksekusi dilanjutkan setelah badan loop.

1.2.2 Break & Continue

Break Python break statement segera menghentikan perulangan seluruhnya. eksekusi program dilanjutkan ke pertanyaan pertama setelah badan loop.

Contoh:

[2]:
```
i = 0
while i < 5:
    print("masih looping")
    i += 1
    if i == 3:
        break
print("Looping Selesai")
```

```
masih looping
masih looping
masih looping
Looping Selesai
```

Continue Python continue statement segera menghentikan iterasi loop saat dijalankan. Eksekusi melompat ke atas loop, dan controlling expression dievaluasi ulang untuk menentukan apakah loop akan dieksukusi lagi atau tidak.

Contoh:

[3]:
```
n = 5
while n>0:
    n -= 1
    if n == 2:
        continue
    print(n)
print("Loop selesai")
```

```
4
3
1
0
Loop selesai
```

1.2.3 For Loop

For loop berguna untuk mengulang blok kode beberapa kali secara berurutan semisal tipe data list, tupel, dict, set, dan string. Menggunakan for loop untuk mengeksekusi satu set kode pernyataan. Menggunakan for loop untuk mengulang string.

Format for loop:

for (var) in (iterable):

```
    <statement(s)>
```

Contoh penrapan for loop:

[4]:
```
buahbuahan = ["Pepaya", "Mangga", "Pisang", "Jambu"]
for buah in buahbuahan:
    print(f"Saya ingin makan buah {buah}")
```

```
Saya ingin makan buah Pepaya
Saya ingin makan buah Mangga
Saya ingin makan buah Pisang
Saya ingin makan buah Jambu
```

1.2.4 Nested Looping

Menurut artian dalam bahasa Inggris, Nested loop dapat diartikan sebagai looping atau perulangan yang bersarang. Maksudnya adalah dalam looping tersebut masih terdapat looping lagi bahkan dalam looping tersebut, masih memungkinkan untuk membuat perulangan lagi sehingga looping tersebut bersarang kedalam.

Contoh penerapan pada while:

[5]:
```
i = 1
j = 1
while j <= 10:
    while i <= 10:
        print(f"{j} x {i} = {j*i}")
        i += 1
    i = 1
    j += 1
    print("="*50)
```

```
1 x 1 = 1
1 x 2 = 2
1 x 3 = 3
1 x 4 = 4
1 x 5 = 5
1 x 6 = 6
1 x 7 = 7
1 x 8 = 8
1 x 9 = 9
1 x 10 = 10
==================================================
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
```

```
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
================================================
3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
3 x 9 = 27
3 x 10 = 30
================================================
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
================================================
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
================================================
6 x 1 = 6
6 x 2 = 12
6 x 3 = 18
6 x 4 = 24
6 x 5 = 30
6 x 6 = 36
6 x 7 = 42
6 x 8 = 48
6 x 9 = 54
6 x 10 = 60
```

```
================================================
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
================================================
8 x 1 = 8
8 x 2 = 16
8 x 3 = 24
8 x 4 = 32
8 x 5 = 40
8 x 6 = 48
8 x 7 = 56
8 x 8 = 64
8 x 9 = 72
8 x 10 = 80
================================================
9 x 1 = 9
9 x 2 = 18
9 x 3 = 27
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
9 x 10 = 90
================================================
10 x 1 = 10
10 x 2 = 20
10 x 3 = 30
10 x 4 = 40
10 x 5 = 50
10 x 6 = 60
10 x 7 = 70
10 x 8 = 80
10 x 9 = 90
10 x 10 = 100
================================================
```

Contoh penerapan pada for:

[1]:
```
for j in range(1, 11):
    for i in range(1, 11):
        print(f"{j} x {i} = {j*i}")
    print("="*50)
```

```
1 x 1 = 1
1 x 2 = 2
1 x 3 = 3
1 x 4 = 4
1 x 5 = 5
1 x 6 = 6
1 x 7 = 7
1 x 8 = 8
1 x 9 = 9
1 x 10 = 10
==================================================
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
==================================================
3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
3 x 9 = 27
3 x 10 = 30
==================================================
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
```

```
================================================
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
================================================
6 x 1 = 6
6 x 2 = 12
6 x 3 = 18
6 x 4 = 24
6 x 5 = 30
6 x 6 = 36
6 x 7 = 42
6 x 8 = 48
6 x 9 = 54
6 x 10 = 60
================================================
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
================================================
8 x 1 = 8
8 x 2 = 16
8 x 3 = 24
8 x 4 = 32
8 x 5 = 40
8 x 6 = 48
8 x 7 = 56
8 x 8 = 64
8 x 9 = 72
8 x 10 = 80
================================================
9 x 1 = 9
9 x 2 = 18
9 x 3 = 27
```

```
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
9 x 10 = 90
==================================================
10 x 1 = 10
10 x 2 = 20
10 x 3 = 30
10 x 4 = 40
10 x 5 = 50
10 x 6 = 60
10 x 7 = 70
10 x 8 = 80
10 x 9 = 90
10 x 10 = 100
==================================================
```

[ ]: