DASAR PYTHON

BAB 5 : IF…ELSE & BOOLEAN

5.1 Tujuan

Mahasiswa mengenal materi If…else dan Boolean pada bahasa pemrograman python

5.2 Ulasan Materi

A. If…else

1. Kondisi Python dan Statements If

Python mendukung kondisi logika yang biasa dari matematika :

a. Sama dengan: a == b
b. Tidak Sama dengan: a != b
c. Kurang dari: a < b
d. Kurang dari atau sama dengan: a <= b
e. Lebih besar dari: a > b
f. Lebih besar dari atau sama dengan: a >= b

Kondisi ini dapat digunakan dalam beberapa cara, paling umum dalam "statements if" dan loop. Sebuah "statements if" ditulis dengan menggunakan kata kunci if. Contoh statement if :

```
a = 33
b = 200
if b > a:
  print("b is greater than a")
```

Output :

```
b is greater than a
```

Dalam contoh ini menggunakan dua variabel, a dan b, yang digunakan sebagai bagian dari pernyataan if untuk menguji apakah b lebih besar dari a. Karena a

adalah 33, dan b adalah 200, kita tahu bahwa 200 lebih besar dari 33, jadi mencetak ke layar bahwa "b lebih besar dari a".

2. **Indentasi**

Python bergantung pada indentasi (spasi putih di awal baris) untuk menentukan ruang lingkup dalam kode. Bahasa pemrograman lain sering menggunakan kurung kurawal untuk tujuan ini. Contoh statements if, tanpa indentasi (akan menimbulkan error) :

```
a = 33
b = 200
if b > a:
print("b is greater than a")
```

Output :

```
File "demo_if_error.py", line 4
    print("b is greater than a")
        ^
IndentationError: expected an indented block
```

3. **Elif**

Kata kunci elif adalah cara python untuk mengatakan "jika kondisi sebelumnya tidak benar, maka coba kondisi ini". Contoh :

```
a = 33
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
```

Output :

```
a and b are equal
```

```

```

Dalam contoh ini a sama dengan b, jadi kondisi pertama tidak benar, tetapi kondisi elif benar, jadi kami mencetak ke layar bahwa " a and b are equal ".

4. **Else**

Kata kunci else menangkap apa pun yang tidak ditangkap oleh kondisi sebelumnya.

Contoh :

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

Output :

```
a is greater than b
```

Dalam contoh ini a lebih besar dari b, jadi kondisi pertama tidak benar, juga kondisi elif tidak benar, jadi kita pergi ke kondisi lain dan mencetak ke layar bahwa " a is greater than b".

Juga dapat memiliki else tanpa elif :

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
else:
  print("b is not greater than a")
```

Output :

```

```

```
b is not greater than a
```

5. **Short Hand If**

   Jika hanya memiliki satu statement untuk dieksekusi, dapat meletakkannya di baris yang sama dengan statement if. Contoh satu line statement if :

```
if a > b: print("a is greater than b")
```

Output :

```
"a is greater than b"
```

6. **Short Hand If…Else**

   Jika Anda hanya memiliki satu pernyataan untuk dieksekusi, satu untuk if, dan satu untuk else, dapat meletakkan semuanya di baris yang sama. Contoh satu line statement if…else :

```
a = 2
b = 330
print("A") if a > b else print("B")
```

Output :

```
B
```

Juga dapat memiliki multiple statement else pada baris yang sama. Contoh satu line statement if else, dengan 3 kondisi :

```
a = 330
b = 330
print("A") if a > b else print("=") if a == b else print("B")
```

Output :

```
=
```

7. **And**

Kata kunci and adalah operator logika, dan digunakan untuk menggabungkan pernyataan statement, Contoh uji if a lebih besar dari b, AND if c lebih besar dari a :

```
a = 200
b = 33
c = 500
if a > b and c > a:
  print("Both conditions are True")
```

Output :

```
Both conditions are True
```

8. **Or**

Kata kunci or adalah operator logika, dan digunakan untuk menggabungkan pernyataan kondisional. Contoh uji jika a lebih besar dari b, OR if a lebih besar dari c :

```
a = 200
b = 33
c = 500
if a > b or a > c:
  print("At least one of the conditions is True")
```

Output :

```
At least one of the conditions is True
```

9. **Nested If**

Dapat memiliki statement if di dalam statement if, ini disebut statements nested if. Contoh :

```
```

```
x = 41

if x > 10:
  print("Above ten,")
  if x > 20:
    print("and also above 20!")
  else:
    print("but not above 20.")
```

Output :

```
Above ten,
and also above 20!
```

10. Statement Pass

Statement if tidak boleh kosong, tetapi if Anda untuk beberapa alasan memiliki statement if tanpa konten, masukkan statement pass untuk menghindari error. Contoh :

```
a = 33
b = 200

if b > a:
  pass

# having an empty if statement like this, would raise an error
without the pass statement
```

Output :

```

```

B. Boolean

1. Nilai Boolean

Dalam pemrograman, Anda sering perlu mengetahui apakah suatu ekspresi Benar atau Salah.

Anda dapat mengevaluasi ekspresi apa pun dengan Python, dan mendapatkan salah satu dari dua jawaban, Benar atau Salah.

Saat Anda membandingkan dua nilai, ekspresi dievaluasi dan Python mengembalikan jawaban Boolean. Contoh :

```
print(10 > 9)
print(10 == 9)
print(10 < 9)
```

Output :

```
True
False
False
```

Saat Anda menjalankan kondisi dalam pernyataan if, Python mengembalikan True atau False.

Print pesan berdasarkan kondisi True atau False. Contoh :

```
a = 200
b = 33

if b > a:
  print("b is greater than a")
else:
  print("b is not greater than a")
```

Output :

```
b is not greater than a
```

2. **Evaluasi Nilai dan Variabel**

   Fungsi bool() memungkinkan untuk mengevaluasi nilai apa pun, dan memberi Anda True atau False sebagai imbalannya,

   Mengevaluasi string dan angka. Contoh :

```
print(bool("Hello"))
print(bool(15))
```

   Output :

```
True
True
```

   Evaluasi dua variabel. Contoh :

```
x = "Hello"
y = 15

print(bool(x))
print(bool(y))
```

   Output :

```
True
True
```

3. **Kebanyakan Nilai True**

   Hampir semua nilai dievaluasi ke True jika memiliki semacam konten. Setiap string Benar, kecuali string kosong. Setiap nomor Benar, kecuali 0. Setiap daftar, tuple, set, dan kamus adalah True, kecuali yang kosong.

   Berikut ini akan mengembalikan True. Contoh :

```
print(bool("abc"))
print(bool(123))
```

```
print(bool(["apple", "cherry", "banana"]))
```

Output :

```
True
True
True
```

4. **Beberapa Nilai False**

Faktanya, tidak banyak nilai yang bernilai False, kecuali nilai kosong, seperti (), [], {}, "", angka 0, dan nilai None. Dan tentu saja nilai False dievaluasi menjadi False. Berikut ini akan mengembalikan False. Contoh :

```
print(bool(False))
print(bool(None))
print(bool(0))
print(bool(""))
print(bool(()))
print(bool([]))
print(bool({}))
```

Output :

```
False
False
False
False
False
False
False
```

Satu nilai lagi, atau objek dalam kasus ini, dievaluasi menjadi False, dan itu jika Anda memiliki objek yang dibuat dari kelas dengan fungsi __len__ yang mengembalikan 0 atau False. Contoh :

```
class myclass():
  def __len__(self):
    return 0

myobj = myclass()
print(bool(myobj))
```

Output :

```
False
```

5. **Fungsi dapat Mengembalikan Boolean**

   Anda dapat membuat fungsi yang mengembalikan Nilai Boolean.

   Cetak jawaban dari suatu fungsi. Contoh :

```
def myFunction() :
  return True

print(myFunction())
```

Output :

```
True
```

Anda dapat mengeksekusi kode berdasarkan jawaban Boolean dari suatu fungsi.

Cetak "YES!" jika fungsi mengembalikan True, jika tidak, cetak "NO!". Contoh :

```
def myFunction() :
  return True

if myFunction():
  print("YES!")
else:
  print("NO!")
```

```

```

Output :

```
YES!
```

Python juga memiliki banyak fungsi bawaan yang mengembalikan nilai boolean, seperti fungsi isinstance(), yang dapat digunakan untuk menentukan apakah suatu objek memiliki tipe data tertentu:

Periksa apakah suatu objek adalah bilangan bulat atau tidak. Contoh :

```
x = 200
print(isinstance(x, int))
```

Output :

```
True
```

5.3 Praktikum

Contoh : Print pesan berdasarkan kondisi True atau False

Pada contoh ini kita akan belajar menerapkan pesan kondisi true atau false sederhana yaitu menggunakan sintaks “if”. Berikut adalah tahapan yang akan di uraikan :

1. Buat variabel a dengan memiliki nilai 200 seperti berikut

   ```
   a = 200
   ```

2. Tambahkan variabel integer b dengan memiliki nilai 33

   ```
   b = 33
   ```

3. Buat fungsi dengan nama ***pengecekan*** yang memiliki parameter variabel angka1 dan angka2

   ```
   def pengecekan(a, b):
   ```

4. Dengan mengimplementasikan sebuah kondisi, tuliskan sintaks “if”, dan kombinasikan dengan operator perbandingan lebih besar dari seperti source code dibawah ini. Memberikan kondisi b > a akan menghasilkan tampilan Boolean True atau False.

```
    if b > a:
```

5. Isi dari kondisi “if” tambahkan perintah return seperti dibawah ini

```
        return "b is greater than a"
```

6. Dibawah hasil dari if tambahkan else :

```
    else:
```

7. Maka return “b is not greater than a” pada hasil else.

```
        return "b is not greater than a"
```

8. Tampilkan hasil dari kondisi if else menggunakan fungsi print()

```
print(pengecekan(a, b))
```

9. Berikut adalah full source code untuk kondisi if else seperti sintaks dibawah ini

```
a = 200
b = 33

def pengecekan(a, b):
    if b > a:
        return "b is greater than a"
    else:
        return "b is not greater than a"

print(pengecekan(a, b))
```

Percobaan 1 : Print pesan berdasarkan kondisi True atau False

Sekarang kerjakan percobaan 1 dengan mengikuti langkah-langkah dibawah ini !

1. Buatlah variabel ***angka1*** yang memiliki nilai 150
2. Buatlah variabel ***angka2*** yang memiliki nilai 40
3. Buat fungsi dengan nama ***pengecekan*** yang memiliki parameter variabel angka1 dan angka2
4. Berikan kondisi if…else untuk pengecekan kondisi menggunakan kondisi if dan else, apabila ,angka2 > angka1 maka menampilkan “angka 2 lebih besar dari angka 1”.Jika tidak maka menampilkan “angka 2 tidak lebih besar dari angka 1”
5. Tampilkan hasil dari kondisi if else menggunakan fungsi print()
6. Jalankan kode program percobaan 1 diatas dengan menekan tombol Check Code Validity