



# Laporan Praktikum Algoritma dan Pemrograman

<b>NIM</b>	<b>71220933</b>
<b>Nama Lengkap</b>	<b>Yosep Yoga Jalu Pamungkas</b>
<b>Minggu ke / Materi</b>	<b>03 / Struktur Kontrol Percabangan</b>

**SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.**

**SAYA SANGGUP MENERIMA SANKSI JIKA MELAKUKAN KEGIATAN PLAGIASI, TERMASUK SANKSI TIDAK LULUS MATA KULIAH INI.**

**PROGRAM STUDI INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS KRISTEN DUTA WACANA**

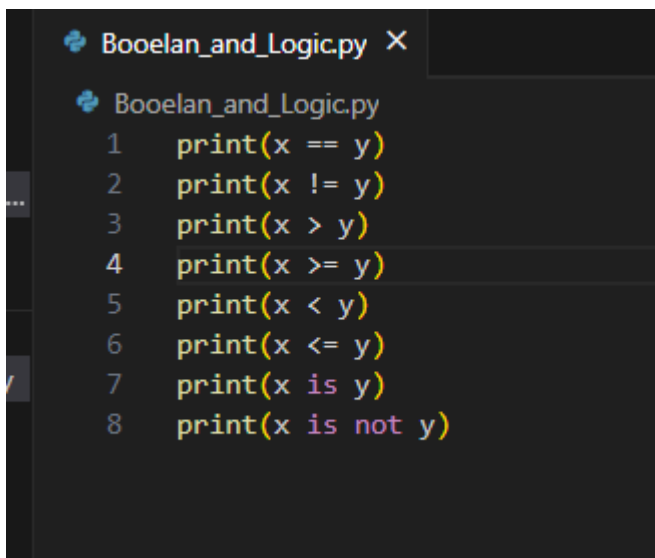
**YOGYAKARTA  
2024**

## BAGIAN I

### MATERI 1

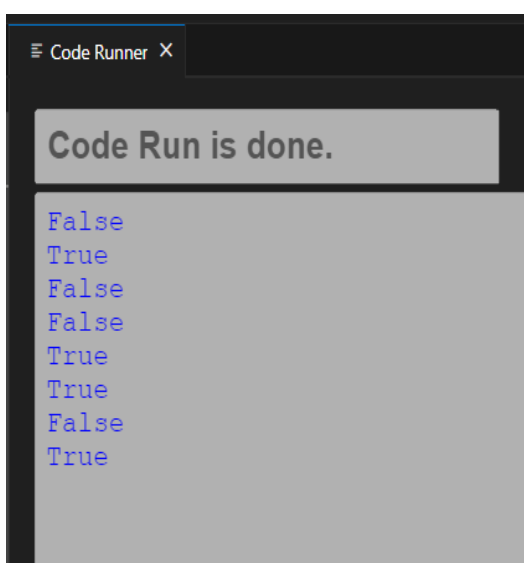
#### Boolean and logic

Boolean AND logic adalah operasi logika di mana hasilnya adalah True hanya jika kedua operand memiliki nilai True. Jika salah satu atau kedua operand memiliki nilai False, maka hasilnya juga akan menjadi False. operator AND direpresentasikan dengan simbol &&.



```
Booelan_and_Logic.py X
Booelan_and_Logic.py
1 print(x == y)
2 print(x != y)
3 print(x > y)
4 print(x >= y)
5 print(x < y)
6 print(x <= y)
7 print(x is y)
8 print(x is not y)
```

Dalam kode tersebut, terdapat pernyataan print yang mengevaluasi ekspresi boolean. misalkan  $x = 1$  dan  $y = 0$ , maka keluarannya akan berupa ekspresi boolean.



```
Code Runner X
Code Run is done.
False
True
False
False
True
True
False
True
```

- "x == y" berarti x sama dengan y
- "x != y" berarti x tidak sama dengan y
- "x > y" berarti x lebih besar dari y
- "x >= y" berarti x lebih besar atau sama dengan y
- "x < y" berarti x lebih kecil dari y
- "x <= y" berarti x lebih kecil atau sama dengan y
- "x is y" berarti x adalah y
- "x is not y" berarti x adalah bukan y

Boolean sangat penting dalam percabangan (IF-ELSE) karena digunakan untuk menilai kondisi yang menentukan jalannya program. Dengan hasil evaluasi yang diberikan oleh boolean, program dapat memutuskan untuk menjalankan atau tidak menjalankan blok kode tertentu. Selain itu, dalam perulangan (FOR dan WHILE), boolean juga berperan penting. Dalam hal ini, boolean digunakan untuk mengatur aliran perulangan.

## MATERI 2

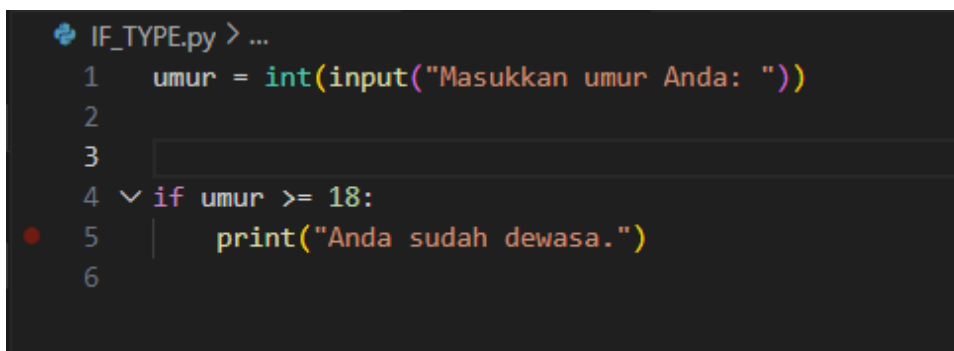
### IF Types

Dalam percabangan python memiliki 3 Tipe

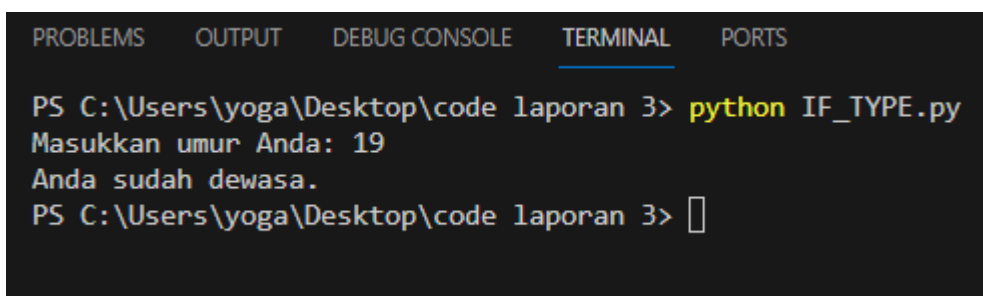
Tipe pertama: **Conditional**

Tipe Conditional pertama dalam Python hanya menggunakan pernyataan IF untuk mengevaluasi kondisi. Jika kondisi tersebut benar, blok kode yang terkait dieksekusi; namun, jika kondisi salah, tidak ada tindakan yang diambil atau blok kode yang dieksekusi.

Contohnya: If umur  $\geq$  18: maka akan di eksekusi ("anda sudah dewasa")



```
IF_TYPE.py > ...
1  umur = int(input("Masukkan umur Anda: "))
2
3
4  if umur >= 18:
5      print("Anda sudah dewasa.")
6
```



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\yoga\Desktop\code laporan 3> python IF_TYPE.py
Masukkan umur Anda: 19
Anda sudah dewasa.
PS C:\Users\yoga\Desktop\code laporan 3> 
```

## Tipe kedua: **Alternative Conditional**

Tipe Conditional kedua menggunakan if dan else.

IF umur kurang dari 18 maka akan di eksekusi ("Anda masih di bawah umur)

ELSE umur lebih dari 18 maka akan di eksekusi ("Anda sudah dewasa")

Memiliki dua kondisi dimana jika input lebih atau kurang dari 18 maka akan mendapat output yang berbeda sesuai kondisi yang di input.

```
IF_TYPE.py > ...
1  umur = int(input("Masukkan umur Anda: "))
2
3
4  ✓ if umur >= 18:
5      print("Anda sudah dewasa.")
6
```

```
else:
    print("Anda masih di bawah umur.")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\yoga\Desktop\code laporan 3> python IF_TYPE.py
Masukkan umur Anda: 16
Anda masih di bawah umur.
PS C:\Users\yoga\Desktop\code laporan 3> 
```

### Tipe ketiga: **Chained Conditional**

ketika satu kondisi tidak terpenuhi, program akan melanjutkan ke kondisi berikutnya hingga menemukan kondisi yang sesuai atau melewati semua kondisi tanpa memenuhi salah satunya.

```
Chained_Conditional.py > ...
1  x = 10
2
3  if x < 0:
4      print("x kurang dari 0")
5  elif x == 0:
6      print("x sama dengan 0")
7  elif x < 10:
8      print("x kurang dari 10")
9  else:
10     print("x lebih besar atau sama dengan 10")
11
```

Dalam contoh ini, jika nilai  $x$  kurang dari 0, maka blok kode pertama akan dieksekusi. Jika tidak, maka kondisi berikutnya akan diuji. Jika  $x$  sama dengan 0, blok kode kedua akan dieksekusi. Demikian seterusnya, hingga kondisi terakhir dalam `else` yang akan dieksekusi jika semua kondisi sebelumnya tidak terpenuhi.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\yoga\Desktop\code laporan 3> python Chained_Conditional.py
x lebih besar atau sama dengan 10
PS C:\Users\yoga\Desktop\code laporan 3> █
```

Hasil outputnya.

### Tipe: Ternary Operator

**nilai1 if kondisi else nilai2**

Operator ternary digunakan sebagai pengganti dari percabangan sederhana yang hanya memiliki satu kondisi. jika kondisi bernilai True, maka nilai1 akan diambil, dan jika kondisi bernilai False, maka nilai2 yang akan diambil.

```
Ternary_Operator.py > ...  
1  x = 5  
2  hasil = "Genap" if x % 2 == 0 else "Ganjil"  
3  print(hasil)  
4
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  
  
PS C:\Users\yoga\Desktop\code laporan 3> python Chained_Conditional.py  
x lebih besar atau sama dengan 10  
PS C:\Users\yoga\Desktop\code laporan 3> 
```

Hasil outputnya.

## MATERI 3:

### User "Mis-Input" Handling

menggunakan loop while True untuk terus meminta input pengguna sampai input yang valid diberikan.

Dalam blok try, program mencoba untuk mengonversi input pengguna menjadi bilangan bulat menggunakan fungsi `int(input())`.

cotohnya:

```
Handling.py > ...
1  while True:
2      try:
3          angka = int(input("Masukkan sebuah angka bulat: "))
4          print("Angka yang dimasukkan:", angka)
5          break
6      except ValueError:
7          print("Input yang dimasukkan bukan angka bulat. Silakan coba lagi.")
8
9  print("Terima kasih!")
10
```

Jika input yang diberikan tidak dapat diubah menjadi bilangan bulat, program akan menangkap kesalahan `ValueError` dan menampilkan pesan kesalahan.

Setelah pesan kesalahan ditampilkan, program akan kembali ke awal loop untuk meminta input dari pengguna lagi.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\yoga\Desktop\code laporan 3> python Handling.py
Masukkan sebuah angka bulat: angka
Input yang dimasukkan bukan angka bulat. Silakan coba lagi.
Masukkan sebuah angka bulat: 2
Angka yang dimasukkan: 2
Terima kasih!
PS C:\Users\yoga\Desktop\code laporan 3> 
```

Jika pengguna memasukkan angka bulat yang valid, program akan mencetak angka tersebut dan keluar dari loop dengan menggunakan `break`.

Program mencetak "Terima kasih!" setelah keluar dari loop, menandakan bahwa proses input telah selesai.



## BAGIAN 2: LATIHAN MANDIRI (60%)

### SOAL 1

Implementasikan penanganan kesalahan input pengguna dari program-program.

#### Contoh A.

```
Latihan1.py > ...  
1  try:  
2      nilai = int(input("Masukkan nilai anda: "))  
3      if nilai >= 50:  
4          print("Anda sudah lulus.")  
5      else:  
6          print("Anda tidak lulus.")  
7  except ValueError:  
8      print("Gunakan angka sebagai input.")  
9
```

Hasil output:

```
Masukkan nilai anda: 40  
Anda tidak lulus.  
PS C:\Users\yoga\Desktop\code laporan 3> python latihan1.py
```

```
Masukkan nilai anda: 80  
Anda sudah lulus.  
PS C:\Users\yoga\Desktop\code laporan 3> 
```

```
Gunakan angka sebagai input.  
PS C:\Users\yoga\Desktop\code laporan 3> 
```

## Contoh B.

```
Latihan_B.py > [?] e
1  try:
2      panjang = float(input("Masukkan panjang (dalam meter): "))
3      lebar = float(input("Masukkan lebar (dalam meter): "))
4
5      if panjang <= 0 or lebar <= 0:
6          raise ValueError("Panjang dan lebar harus lebih besar dari nol.")
7
8      luas = panjang * lebar
9      print("Luas persegi panjang adalah:", luas, "meter persegi.")
10
11 except ValueError as e:
12     print("Error:", e)
13
```

Hasil outputnya:

```
▼ TERMINAL
PS C:\Users\yoga\Desktop\code laporan 3> python Latihan_B.py
Masukkan panjang (dalam meter): 5
Masukkan lebar (dalam meter): 4
Luas persegi panjang adalah: 20.0 meter persegi.
PS C:\Users\yoga\Desktop\code laporan 3> 
```

## Contoh C.

```

Latihan_C.py > ...
1  try:
2      angka1 = float(input("Masukkan angka pertama: "))
3      angka2 = float(input("Masukkan angka kedua: "))
4      angka3 = float(input("Masukkan angka ketiga: "))
5
6      hasil = angka1 + angka2 + angka3
7      print("Hasil penjumlahan ketiga angka:", hasil)
8
9  except ValueError:
10     print("Input yang dimasukkan harus berupa angka.")
11

```

Hasil outputnya:

```

PS C:\Users\yoga\Desktop\code laporan 3> python Latihan_C.py
Masukkan angka pertama: 1
Masukkan angka kedua: 2
Masukkan angka ketiga: 3
Hasil penjumlahan ketiga angka: 6.0
PS C:\Users\yoga\Desktop\code laporan 3> 

```

Diatas adalah tiga contoh program yang menggunakan menggunakan blok try dan except. Untuk mengatasi kesalahan pengguna dalam menginput.

## SOAL 2

Implementasikan percabangan

```
Soal2_A.py > ...
1  try:
2      angka = float(input("Masukkan sebuah angka: "))
3
4      status = "positif" if angka > 0 else "negatif" if angka < 0 else "nol"
5
6      print("Angka yang dimasukkan adalah", status)
7
8  except ValueError:
9      print("Input yang dimasukkan harus berupa angka.")
10
```

Outputnya:

```
PS C:\Users\yoga\Desktop\code laporan 3> python Soal2_A.py
Masukkan sebuah angka: 2
Angka yang dimasukkan adalah positif
PS C:\Users\yoga\Desktop\code laporan 3> python Soal2_A.py
Masukkan sebuah angka: 0
Angka yang dimasukkan adalah nol
PS C:\Users\yoga\Desktop\code laporan 3> █
```

Program meminta pengguna untuk memasukkan sebuah angka. Input dari pengguna dikonversi menjadi bilangan floating point.

Menggunakan ternary operator untuk menentukan apakah angka tersebut positif, negatif, atau nol.

Jika angka lebih besar dari 0, status akan diatur sebagai "positif". Jika angka kurang dari 0, status akan diatur sebagai "negatif". Jika angka sama dengan 0, status akan diatur sebagai "nol".

Hasil status akan dicetak. Jika terjadi kesalahan dalam memproses input (misalnya jika pengguna tidak memasukkan angka), program akan menangkap kesalahan dan mencetak pesan kesalahan yang sesuai.

### SOAL 3

program yang dapat menampilkan jumlah hari dalam suatu bulan di tahun 2020.

```
Soal3.py > ...
1 def jumlah_hari(bulan):
2     bulan = bulan.lower()
3
4     if bulan == "januari" or bulan == "maret" or bulan == "mei" or bulan == "juli" or bulan == "agustus" or bulan == "oktober" or bulan == "desember":
5         return 31
6     elif bulan == "april" or bulan == "juni" or bulan == "september" or bulan == "november":
7         return 30
8     elif bulan == "februari":
9         return 29
10    else:
11        return "Bulan yang dimasukkan tidak valid."
12
13    bulan_input = input("Masukkan nama bulan (Contoh: Januari): ")
14
15    print("Jumlah hari dalam bulan", bulan_input, "adalah:", jumlah_hari(bulan_input))
16
```

outputnya:

```
PS C:\Users\yoga\Desktop\code laporan 3> python soal3.py
Masukkan nama bulan (Contoh: Januari): januari
Jumlah hari dalam bulan januari adalah: 31
PS C:\Users\yoga\Desktop\code laporan 3> 
```

Menggunakan struktur percabangan IF, ELIF dan ELSE dan untuk menentukan jumlah hari berdasarkan nama bulan yang dimasukkan pengguna.

fungsi jumlah\_hari(bulan) yang menerima sebuah string bulan sebagai input dan mengembalikan jumlah hari dalam bulan tersebut. untuk menentukan jumlah hari berdasarkan nama bulan yang dimasukkan pengguna. Program memperhitungkan tahun kabisat untuk bulan Februari, yang memiliki 29 hari.

Setelah pengguna memasukkan nama bulan, program akan menampilkan jumlah hari dalam bulan tersebut.

#### SOAL 4

Sebuah program meminta pengguna memasukkan ketiga panjang sisi suatu segitiga (berarti pengguna memasukkan tiga bilangan)

```
Soal4.py > ...
1 def jenis_segitiga(sisi1, sisi2, sisi3):
2     if sisi1 == sisi2 == sisi3:
3         return "Segitiga sama sisi"
4     elif sisi1 == sisi2 or sisi1 == sisi3 or sisi2 == sisi3:
5         return "Segitiga sama kaki"
6     else:
7         return "Segitiga sembarang"
8
9 try:
10     sisi1 = float(input("Masukkan panjang sisi pertama: "))
11     sisi2 = float(input("Masukkan panjang sisi kedua: "))
12     sisi3 = float(input("Masukkan panjang sisi ketiga: "))
13
14     if sisi1 + sisi2 > sisi3 and sisi1 + sisi3 > sisi2 and sisi2 + sisi3 > sisi1:
15         print("Segitiga dengan panjang sisi-sisi yang diberikan adalah", jenis_segitiga(sisi1, sisi2, sisi3))
16     else:
17         print("Panjang sisi-sisi yang dimasukkan tidak membentuk segitiga.")
18
19 except ValueError:
20     print("Panjang sisi harus berupa bilangan.")
21
```

Outputnya:

```
PS C:\Users\yoga\Desktop\code laporan 3> python Soal4.py
Masukkan panjang sisi pertama: 4
Masukkan panjang sisi kedua: 3
Masukkan panjang sisi ketiga: 4
Segitiga dengan panjang sisi-sisi yang diberikan adalah Segitiga sama kaki
PS C:\Users\yoga\Desktop\code laporan 3> █
```

Program ini meminta pengguna untuk memasukkan panjang sisi-sisi segitiga dan kemudian menentukan jenis segitiga yang terbentuk berdasarkan panjang sisi-sisinya. Ini dilakukan dengan menggunakan fungsi untuk menentukan jenis segitiga (sama sisi, sama kaki, atau sembarang) dan struktur percabangan IF, ELIF dan ELSE untuk menentukan jenis segitiga berdasarkan panjang sisi-sisinya. Program akan mencetak hasil jenis segitiga atau pesan kesalahan jika panjang sisi-sisi yang dimasukkan tidak dapat membentuk segitiga.