

MODUL MATA KULIAH

BAHASA PEMROGRAMAN DASAR

PG168 – 3 SKS



**FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BUDI LUHUR**

**JAKARTA
SEPTEMBER 2021**

TIM Penyusun

Agus Umar Hamdani, M.Kom
Tri Ika Jaya Kusumawati, M.Kom¹



MODUL PERKULIAHAN #6 KONTROL PERCABANGAN

Capaian Pembelajaran	:	Mahasiswa Mampu: <ol style="list-style-type: none">1. Memahami bentuk umum kondisi percabangan If dan Switch CASE in Python.2. Memahami penggunaan kontrol kondisional dan logika menggunakan kontrol percabangan .
Sub Pokok Bahasan	:	<ol style="list-style-type: none">1. Kontrol percabangan IF, IF-Else, Multiple IF, IF-ELIF-ELSE dan Nested-IF2. Kontrol Switch Case di dalam Python.3. Operator Perbandingan dan Logika di dalam kontrol IF.
Daftar Pustaka	:	<ol style="list-style-type: none">1. Zarman, Wendi dan Wicaksono, Mochamad Fajar. <i>"Implementasi Algoritma dalam bahasa Python"</i>. Edisi Pertama. Bandung : Penerbit Informatika. 2020.2. Kurniawati, Arik. <i>"Algoritma dan Pemrograman menggunakan Python"</i>. Edisi Pertama. Yogyakarta : Depublish. 2016.3. Ismah. <i>"Pemrograman Komputer Dasar-dasar Python"</i>. Jakarta : Fakultas Ilmu Pendidikan Universitas Muhammadiyah Jakarta. 2017.4. Irfani, M. Haviz dan Dafid. <i>"Modul Praktikum Dasar Pemrograman dengan bahasa Python"</i>. Palembang : Sekolah Tinggi Manajemen Informatika Global Informatika Multidata Palembang. 2016.5. Fikri, Rijalul. <i>"Praktikum Algoritma dan Pemrograman Komputer"</i>. Surabaya : Program Studi Teknik Komputer dan Telematika Jurusan Teknik Elektro Fakultas Teknologi Industri Institut Teknologi Sepuluh Nopember. 2010.6. Sugiana, Owo. <i>Membuat Aplikasi Bisnis menggunakan bahasa Python dan database berbasis SQL</i>. Jakarta. 2003.7. Septian, Ridwan Fadjar. <i>Buku Serial Open Source Belajar Pemrograman Python Dasar</i>. Versi 1. Bandung : POSS-UPI. 2013.8. Hendri. <i>Cepat Mahir Python</i>. Ilmu Komputer.com. 2003.9. Herho, Sandy H.S. <i>Tutorial Pemrograman Python 2 Untuk pemula</i>. Bandung: WCPL Press. 2017.10. <i>Welcome to Python.org</i> (diakses pada 29 September 2021 pukul : 21.00 WIB)11. <i>Programiz.com</i>. Python Operators (https://www.programiz.com/python-programming/operators) diakses pada 29 September 2021 pukul 21.32 WIB)

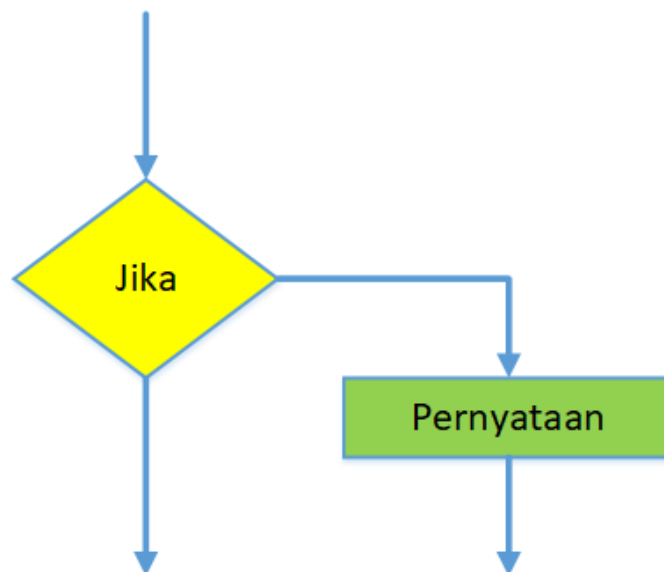
PRAKTIKUM 6

KONTROL PERCABANGAN

3.1 Teori Singkat

Struktur Kontrol Percabangan adalah ekspresi yang perintah atau aksinya dikerjakan, jika kondisi yang menjadi syaratnya terpenuhi [1]. Dengan demikian kita dapat mengatur bagaimana suatu program bekerja sesuai dengan kebutuhan yang kita inginkan. Pemilihan kondisi tersebut harus dilakukan jika kondisi yang dihadapi menghasilkan nilai benar (True). Disebut percabangan, dikarenakan untuk menggambarkan alur program yang bercabang. Pada model algoritma menggunakan Flowchart, logika "**Jika ... , Maka ...**" digambarkan dalam bentuk cabang.

Gambar 6.1 merupakan struktur blok untuk kontrol IF.



Gambar 6.1 Flowchart Blok Kontrol IF

Selain percabangan, struktur ini juga disebut *Kontrol Flow*, *Decision*, struktur kondisi, Struktur IF, dsb. Percabangan akan mampu membuat program berpikir dan menentukan tindakan sesuai dengan logika/kondisi yang kita berikan. Terdapat beberapa kontrol percabangan yang dapat digunakan dalam bahasa Python, antara lain : IF, IF-ELSE, ELIF.

Kontrol IF mendukung penggunaan kondisi Logika aritmatika, seperti :

Tabel 6.1 Operator Perbandingan dalam Kontrol IF

Operator Perbandingan	Deskripsi
$a == b$	Sama dengan
$a != b$	Tidak sama dengan
$a < b$	Nilai a lebih kecil dari nilai b
$a \leq b$	Nilai a lebih kecil atau sama dengan dari nilai b
$a > b$	Nilai a lebih besar dari nilai b
$a \geq b$	Nilai a lebih besar atau sama dengan dari nilai b

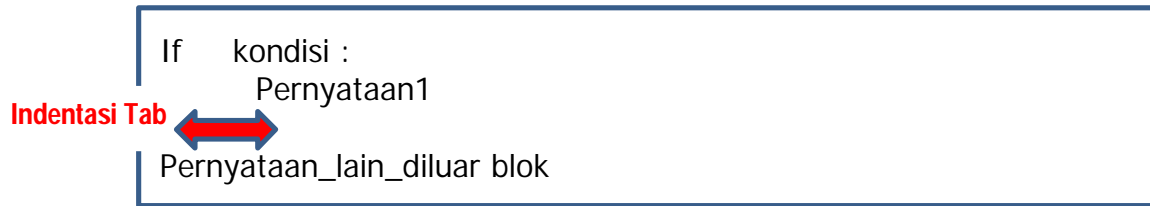
Tabel 6.2 Operator Logika dalam Kontrol IF

Operator Logika	Deskripsi
$a > b$ <u>AND</u> $c > a$	Nilai a lebih besar dari nilai b <u>DAN</u> nilai c lebih besar dari nilai a
$a > b$ <u>OR</u> $c > a$	Nilai a lebih besar dari nilai b <u>ATAU</u> nilai c lebih besar dari nilai a
<u>NOT</u> $c > a$	Nilai a lebih besar dari nilai b <u>ATAU</u> nilai c lebih besar dari nilai a

a. Kontrol IF

Python memiliki cara yang berbeda, gaya penulisan sintak sangat berbeda dan unik, jika pada bahasa pemrograman C++, PHP maupun java menggunakan tanda pengenal {}, berbeda dengan python yang tidak mengenal tanda tersebut. Untuk mengelompokkan suatu blok kode dibuat dengan cara melakukan pergeseran kode terhadap perintah-perintah terkait ke arah kanan.

Berikut ini sintak umum penulisan percabangan IF di bahasa Python :



Gambar 6.2 Struktur Kontrol IF

Penulisan kondisi di dalam python **TIDAK** diapit dengan tanda () seperti pada pemrograman bahasa pemrograman C++, PHP maupun Java. Cukup menuliskan langsung dan diakhiri dengan tanda : (titik dua). Setiap pernyataan yang akan dieksekusi harus diberikan Indentasi Tab atau spasi 2x sebagaimana aturan penulisan sintak dalam Python. Setiap Perintah didalam Python pun **TIDAK** diakhiri dengan tanda ; (titik koma). Bagian kondisi adalah sebuah variabel / atau nilai yang bertipe data boolean. Baik berupa nilai True/False secara langsung, atau pun sebuah ekspresi logika. Jika kondisi bernilai True, maka bagian pernyataan1 dan pernyataan2 akan dieksekusi oleh komputer.

Studi Kasus 6.1

Program pada gambar 6.3 merupakan contoh penerapan struktur kontrol IF. Program tersebut mendemonstrasikan cara menyeleksi kelulusan mahasiswa menurut nilai ujiannya.

```
1  ▶  
2  nilai_akhir = int(input("Masukan Nilai : "))  
3  if nilai_akhir >= 60 :  
4      keterangan = "Lulus"  
5  print("Anda dinyatakan : ", keterangan, "dengan nilai : ", nilai_akhir)  
6  
7  
8
```

Gambar 6.3 Contoh Program dengan Kontrol IF (1)

Perhatikan gambar 6.3, program akan menunggu kita untuk memasukkan nilai. Nilai yang diinput akan dikonversi ke dalam tipe Integer dan disimpan

pada variabel nilai_akhir. Nilai yang terdapat pada variabel nilai_akhir akan diperiksa, apakah bernilai lebih atau sama dengan (\geq) 60 atau tidak. Jika variabel nilai_akhir bernilai lebih besar atau sama dengan 60 (≥ 60), maka variabel keterangan = "Lulus" akan dicetak. Sebaliknya, jika nilai_akhir bernilai kurang dari 60 (< 60), maka program akan berakhir.

Setelah kode program diatas dijalankan dan diinput dengan nilai 77, maka hasil keluaran yang ditampilkan sebagai berikut :

```
Masukan Nilai : 75
Anda dinyatakan : Lulus dengan nilai : 75
```

Gambar 6.4 Contoh Hasil Keluaran Program

Studi Kasus 6.2

Program pada gambar 6.5 merupakan contoh program untuk menyeleksi kategori suatu bilangan bulat, apakah termasuk kategori bilangan Genap atau bilangan Ganjil.

```
1  ▶
2  keterangan = "Ganjil"
3  x = int(input("Masukan Nilai : "))
4  if x % 2 == 0 :
5      keterangan = "genap"
6
7  print(x, " merupakan bilangan : ", keterangan)
8
```

Gambar 6.5 Contoh Program dengan Kontrol IF (2)

Perhatikan gambar 6.5, variabel keterangan didefinisikan bertipe **String** dan berisi nilai awal "**Ganjil**". Program tersebut akan menunggu kita untuk memasukkan nilai. Nilai tersebut akan dikonversi ke tipe **Integer** dan disimpan ke variabel x. untuk mengetahui sebuah bilangan itu termasuk ke dalam kategori **Genap** atau **Ganjil**, maka kita perlu menyeleksi sisa bagi nilai x setelah dibagi 2. Jika sisa bagi 2 terhadap nilai x adalah 0, maka keterangan = "**Genap**" akan dieksekusi. Dan sebaliknya, jika sisa bagi 2 terhadap nilai x adalah 1, maka program akan mengeksekusi variabel keterangan = "**Ganjil**".

Setelah kode program diatas dijalankan dan diinput dengan nilai $x = 35$, maka hasil keluaran yang ditampilkan sebagai berikut :

```
Masukan Nilai : 35
35 merupakan bilangan : Ganjil
```

Gambar 6.6 Contoh Hasil Keluaran Program

Jika nilai yang akan diperiksa hanya terdiri atas satu kondisi aja, maka dapat menuliskan kode programnya dalam satu baris (Penyederhanaan kontrol If ... Else), seperti terlihat pada gambar 6.7.

```
1  ▶ a = int(input("Masukan nilai a : "))
2
3  b = int(input("Masukan nilai b : "))
4
5  print("A") if a > b else print("B")
6
```

Gambar 6.7 Contoh Penyederhanaan Kontrol If..ELSE

Setelah kode program diatas dijalankan dan diinput dengan nilai-nilai :

a = 50

b = 10

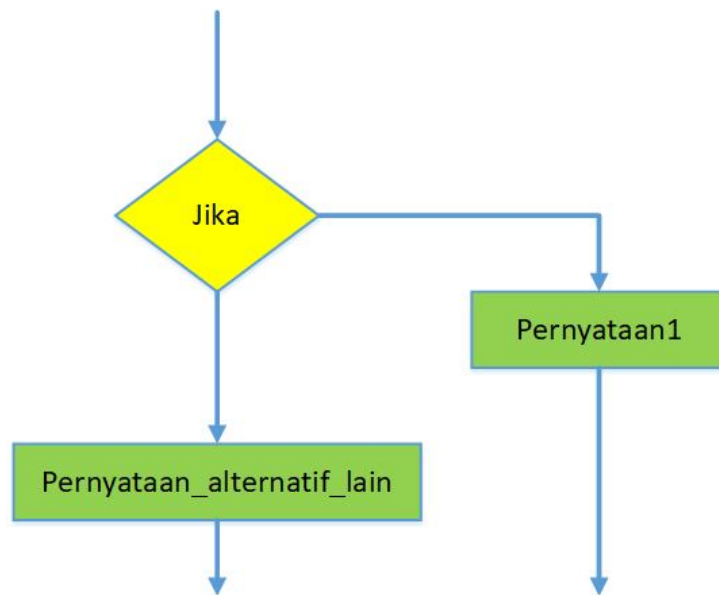
maka hasil keluaran yang ditampilkan sebagai berikut :

```
Masukan nilai a : 50
Masukan nilai b : 10
A
```

Gambar 6.8 Contoh Hasil Keluaran Program

b. Kontrol IF-ELSE.

Struktur kontrol IF-ELSE sebenarnya hamper sama dengan pernyataan IF. Perbedaanya adalah struktur kontrol IF-ELSE memiliki lebih dari satu aksi. Blok IF-ELSE ini biasa dinamakan percabangan, karena memiliki setidaknya 2 cabang. Satu aksi akan dikerjakan, jika kondisinya bernilai benar (TRUE) dan satu aksi lagi akan dikerjakan, jika kondisinya bernilai salah (FALSE). Gambar 6.9 merupakan struktur blok untuk kontrol IF-ELSE.



Gambar 6.10 Flowchart Blok Kontrol IF-ELSE

Berikut ini sintak umum penulisan percabangan IF-ELSE di bahasa Python :

```
If    kondisi :  
    Pernyataan1  
else  
    Pernyataan_alternatif2
```

Gambar 6.11 Struktur Kontrol IF-ELSE

Studi Kasus 6.3

Program pada gambar 6.12 merupakan contoh program untuk menyeleksi apakah IPK yang diinput termasuk ke dalam kategori “Berhak Mendapatkan Predikat Cumlaude” atau “Tidak Berhak Mendapatkan Predikat Cumlaude”.


```

1  ▶
2
3  x = float(input("Masukan IPK anda : "))
4  if x >= 3.5 :
5      keterangan = "Berhak mendapatkan predikat Cumlaude"
6  else :
7      keterangan = "Tidak Berhak mendapatkan predikat Cumlaude"
8  print("IPK : ", x, keterangan)
9

```

Gambar 6.12 Contoh Program dengan Kontrol IF-ELSE (1)

Perhatikan gambar 6.12, variabel x didefinisikan bertipe **Float** digunakan untuk menampung nilai IPK. Program tersebut akan menunggu kita untuk memasukkan nilai x. setelah diinput, maka nilai x akan diperiksa, apakah lebih besar atau sama dengan 3.5 (≥ 3.5) atau tidak. Jika nilai $x \geq 3.5$, maka keterangan = "Berhak mendapatkan predikat Cumlaude" akan dicetak. Sebaliknya, jika nilai $x < 3.5$, maka keterangan = "Tidak Berhak mendapatkan predikat Cumlaude" akan dicetak.

Setelah kode program diatas dijalankan dan diinput dengan nilai 77, maka hasil keluaran yang ditampilkan sebagai berikut :

```

Masukan IPK anda : 3.45
IPK : 3.45 Tidak Berhak mendapatkan predikat Cumlaude

```

Gambar 6.13 Contoh Hasil Keluaran Program

Studi Kasus 6.4

Program pada gambar 6.14 merupakan contoh program untuk menyeleksi pegawai yang mendapatkan kenaikan gaji berdasarkan gaji yang diterima. Kenaikan gaji dihitung berdasarkan masa kerja. Untuk masa kerja diatas 10 tahun mendapatkan kenaikan gaji sebesar 20% dari gaji pokok, sedangkan masa kerja lainnya mendapatkan kenaikan gaji sebesar 5% dari gaji pokok.

```

1  ▶
2  masa_kerja = float(input("Masukan masa kerja : "))
3  gaji_pokok = int(input("Masukan gaji pokok : "))
4  if masa_kerja >= 10 :
5      kenaikan_gaji = (20/100) * float(gaji_pokok)
6  else :
7      kenaikan_gaji = (5/100) * float(gaji_pokok)
8
9  print("Jumlah kenaikan gaji : ", kenaikan_gaji)
10 total_gaji = gaji_pokok + kenaikan_gaji
11 print("Total gaji : ", total_gaji)
12

```

Gambar 6.14 Contoh Program dengan Kontrol IF-ELSE (2)

Perhatikan gambar 6.14, variabel `masa_kerja` didefinisikan bertipe **Float** untuk menampung nilai `masa_kerja`. Variabel `gaji_pokok` didefinisikan bertipe **Integer** untuk menampung nilai `gaji_pokok`. Program tersebut akan menunggu kita untuk memasukkan nilai untuk variabel `masa_kerja` dan `gaji_pokok`. Setelah diinput, maka nilai `masa_kerja` akan diperiksa, apakah lebih besar atau sama dengan 10 (tahun) (≥ 10) atau tidak. Jika nilai `masa_kerja` ≥ 10 , maka `kenaikan_gaji` sebesar 20% dari gaji pokok akan dicetak. Sebaliknya, jika nilai `masa_kerja` < 10 , maka `kenaikan_gaji` sebesar 5% dari gaji pokok akan dicetak.

Setelah kode program diatas dijalankan dan diinput dengan nilai sebagai berikut :

Masa kerja = 9.5

Gaji pokok = 3.500.000

Maka : masa kerja tersebut akan terdeteksi sebagai kondisi masa kerja < 10 tahun sehingga hasil keluaran yang ditampilkan sebagai berikut :

```

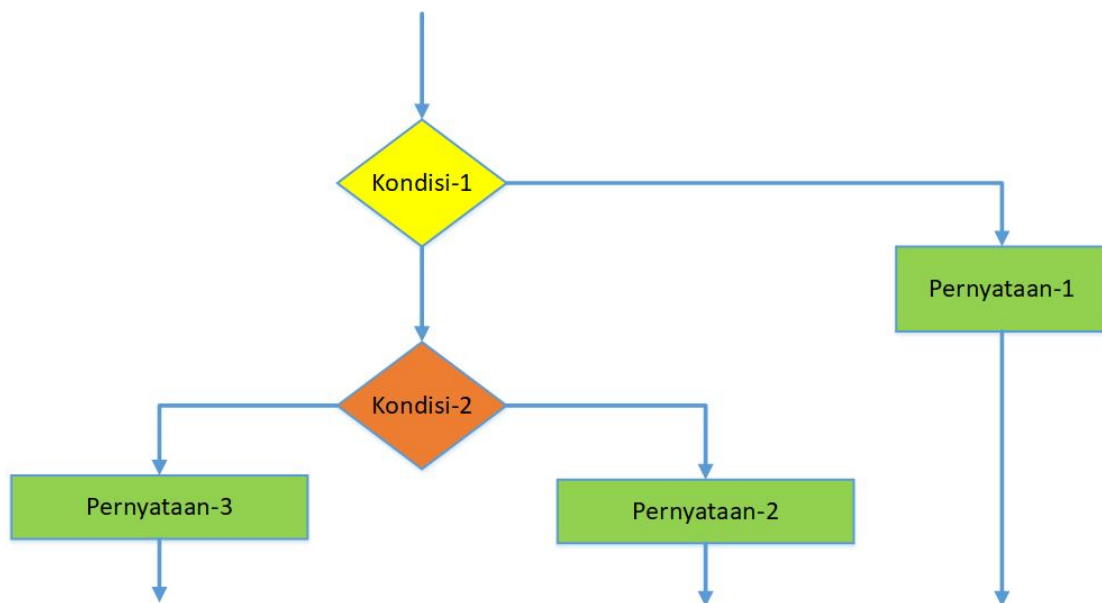
Masukan masa kerja : 9.5
Masukan gaji pokok : 3500000
Jumlah kenaikan gaji : 175000.0
Total gaji : 3675000.0

```

Gambar 6.15 Contoh Hasil Keluaran Program

c. Kontrol Multiple IF

Struktur kontrol Multiple IF merupakan lanjutan/percabangan logika dari "kondisi IF". Dengan Multiple IF kita bisa membuat kode program yang akan menyeleksi beberapa kemungkinan yang bisa terjadi. Hampir sama dengan kondisi "ELSE", bedanya kondisi "IF" bisa banyak dan tidak hanya satu kondisi yang diseleksi.



Gambar 6.16 Flowchart Blok Kontrol Multiple IF

Berikut ini sintak umum penulisan percabangan Multiple IF di bahasa Python :

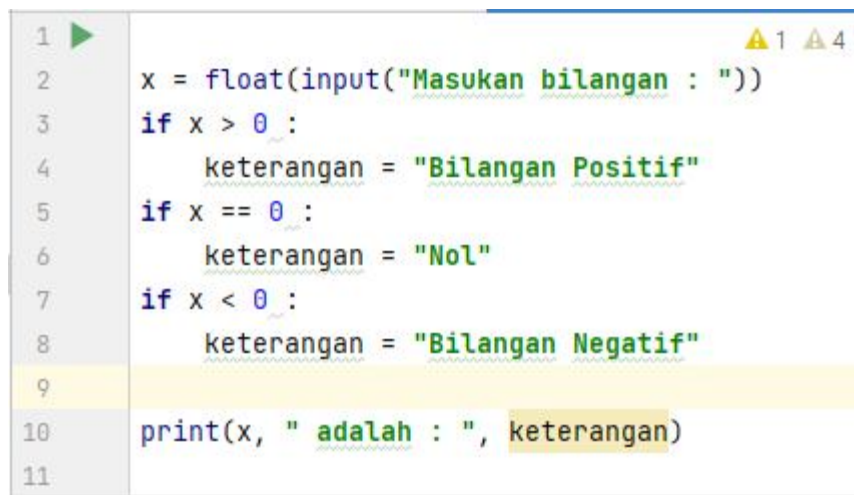
```
If kondisi-1 :  
    Pernyataan1  
If kondisi-2 :  
    Pernyataan_alternatif2  
If kondisi-3 :  
    Pernyataan_alternatif3  
If  
    Pernyataan_alternatif4
```

Gambar 6.17 Struktur Kontrol Multiple IF

Studi Kasus 6.5

Program pada gambar 6.18 merupakan contoh program untuk menyeleksi sebuah nilai, apakah termasuk ke dalam kategori bilangan Positif, Nol atau bilangan Negatif dengan ketentuan sebagai berikut :

1. Jika nilai x lebih besar dari 0 ($x > 0$), maka cetaklah keterangan "Bilangan Positif".
2. Jika nilai x sama dengan 0 ($x == 0$), maka cetaklah keterangan "Nol".
3. Jika nilai x lebih kecil dari 0 ($x < 0$), maka cetaklah keterangan "Bilangan Negatif".



```
1  ▶ x = float(input("Masukan bilangan : "))
2
3  if x > 0 :
4      keterangan = "Bilangan Positif"
5  if x == 0 :
6      keterangan = "Nol"
7  if x < 0 :
8      keterangan = "Bilangan Negatif"
9
10 print(x, " adalah : ", keterangan)
11
```

Gambar 6.18 Contoh Program dengan Kontrol Multiple IF

Perhatikan gambar 6.19, variabel x didefinisikan bertipe **Float** untuk menampung sebuah nilai dari keyboard. Program tersebut akan menunggu kita untuk memasukkan nilai untuk variabel x . Setelah diinput, maka nilai x akan diperiksa, apakah lebih besar dari 0 ($x > 0$), jika benar, maka keterangan "Bilangan Positif" akan dicetak. Jika nilai x sama dengan 0 ($x == 0$), maka keterangan "Nol" akan dicetak. Jika nilai x lebih kecil dari 0 ($x < 0$), maka keterangan "Bilangan Positif" akan dicetak. Setelah kode program diatas dijalankan dan diinput dengan nilai sebagai berikut :

$x = 3.4$

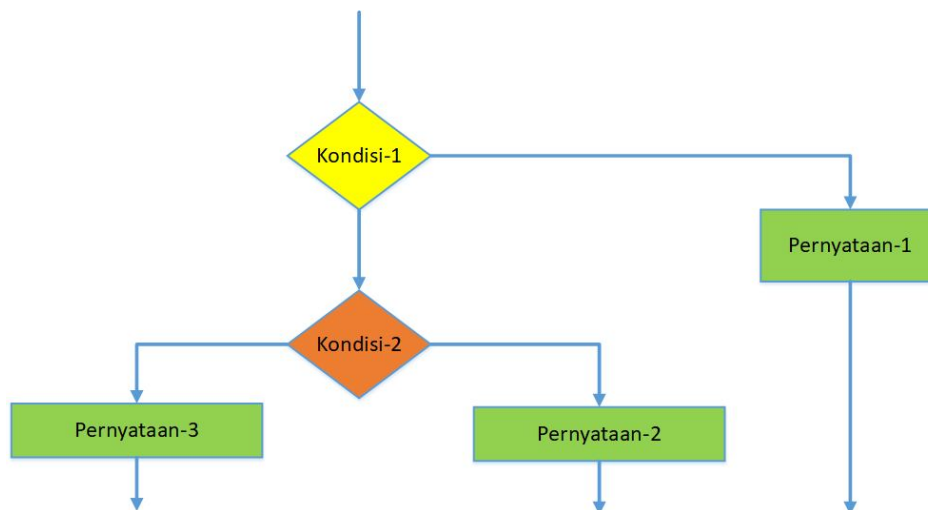
Maka hasil keluaran yang ditampilkan sebagai berikut :

```
Masukan bilangan : 3.4
3.4 adalah : Bilangan Positif
```

Gambar 6.20 Contoh Hasil Keluaran Program

d. Kontrol IF-ELIF-ELSE

Struktur kontrol IF-ELIF-ELSE merupakan lanjutan/percabangan logika dari “kondisi IF”. Dengan IF-ELIF-ELSE kita bisa membuat kode program yang akan menyeleksi beberapa kemungkinan yang bisa terjadi. Hampir sama dengan kondisi “ELSE”, bedanya kondisi “IF-ELIF-ELSE” bisa banyak dan tidak hanya satu kondisi yang diseleksi. Struktur kontrol IF-ELIF-ELSE dapat dilihat pada gambar 6.21.



Gambar 6.21 Flowchart Blok Kontrol IF-ELIF-ELSE

Berikut ini sintak umum penulisan percabangan IF-ELIF-ELSE di bahasa Python :

```
If kondisi-1 :
    Pernyataan1
Elif kondisi-2 :
    Pernyataan_alternatif2
Elif kondisi-3 :
    Pernyataan_alternatif3
Else
    Pernyataan_alternatif4
```

Gambar 6.13 Struktur Kontrol IF-ELIF-ELSE

Studi Kasus 6.6

Program pada gambar 6.22 merupakan pengembangan dari contoh program pengembangan dari studi kasus 6.5, tetapi menggunakan perintah IF-ELIF-ELSE.

```
1  ▶ x = float(input("Masukan bilangan : "))
2
3  if x > 0 :
4      keterangan = "Bilangan Positif"
5  elif x == 0 :
6      keterangan = "Nol"
7  else :
8      keterangan = "Bilangan Negatif"
9
10 print(x, " adalah : ", keterangan)
11
```

Gambar 6.22 Contoh Program dengan Kontrol IF-ELIF-ELSE

Perhatikan gambar 6.22, variabel **x** didefinisikan bertipe **Float** untuk menampung sebuah nilai dari keyboard. Program tersebut akan menunggu kita untuk memasukkan nilai untuk variabel **x**. Setelah diinput, maka nilai **x** akan diperiksa, apakah lebih besar dari 0 ($x > 0$), jika benar, maka keterangan "Bilangan Positif" akan dicetak. Jika nilai **x** sama dengan 0 ($x == 0$), maka keterangan "Nol" akan dicetak. Jika nilai **x** lebih kecil dari 0 ($x < 0$), maka keterangan "Bilangan Negatif" akan dicetak.

Setelah kode program diatas dijalankan dan diinput dengan nilai sebagai berikut :

$x = -9$

Maka hasil keluaran yang ditampilkan sebagai berikut :

```
Masukan bilangan : -9
-9.0 adalah : Bilangan Negatif
```

Gambar 6.23 Contoh Hasil Keluaran Program

e. Kontrol SWITCH-CASE di bahasa Python

Kontrol Switch Case digunakan untuk percabangan yang kondisinya banyak. Sedangkan Kontrol IF hanya bisa untuk menangani dua kondisi yaitu TRUE atau FALSE. Pernyataan switch case adalah pengganti pernyataan 'IF-ELSE'.

Sintak penulisan kontrol Switch Case dalam bahasa C++ dapat dilihat pada gambar 6.24.

```
switch (kondisi) {  
  case value-1 :  
    Pernyataan1  
  Case value-2 :  
    Pernyataan_alternatif2  
  default :  
    Pernyataan_alternatif3  
}
```

Gambar 6.24 Struktur Kontrol Switch-Case

Keuntungan menggunakan Pernyataan Switch Case dalam program adalah sebagai berikut:

- 1) Lebih mudah untuk men-debug.
- 2) Lebih mudah untuk membaca program oleh siapa saja kecuali programmer.
- 3) Lebih mudah dipahami dan juga lebih mudah dipelihara.
- 4) Lebih mudah untuk memverifikasi semua nilai yang akan diperiksa untuk ditangani.

Dalam bahasa python tidak menggunakan pernyataan switch case seperti bahasa lain, kita dapat menggunakan 3 (tiga) pengganti untuk mengimplementasikan Switch Case dengan menggunakan cara sebagai berikut :

a. Cara ke-1 : menggunakan IF-ELIF-ELSE

Berikut ini sintak umum penulisan kontrol IF-ELIF di bahasa Python :

```
If    kondisi-1 :  
    Pernyataan1  
Elif  kondisi-2 :  
    Pernyataan_alternatif2  
Elif  kondisi-3 :  
    Pernyataan_alternatif3  
Else  
    Pernyataan_alternatif4
```

Gambar 6.25 Struktur Kontrol IF-ELIF

Studi Kasus 6.6

Lihat Kembali contoh program pada gambar 6.22 sebelumnya yang menggunakan perintah IF-ELIF-ELSE.

b. Cara ke-2 : *Dictionary Mapping*

Dalam bahasa Python terdapat tipe data Dictionary yang menyimpan sekelompok obyek menggunakan pasangan **Key-Value**. Kita dapat menggunakan Dictionary untuk mengganti kontrol Switch-Case. **Key-Value** pada Dictionary bekerja mirip seperti seperti Kontrol Switch-Case.

Sintak penulisan tipe data Dictionary adalah

```
nama_variabel = {  
    "Key1" : "value",  
    "Key2" : "value",  
    ".." : "..."}  
}
```

Studi Kasus 6.7

Program pada gambar 6.26 merupakan contoh implementasi kontrol Switch-Case menggunakan tipe Dictionary.


```

1  ▶ # Implementasi kontrol Switch Case menggunakan Dictionary
2  def Senin():
3      return "Senin"
4  def Selasa():
5      return "Selasa"
6  def Rabu():
7      return "Rabu"
8  def Kamis():
9      return "Kamis"
10 def Jumat():
11     return "Jum'at"
12 def Sabtu():
13     return "Sabtu"
14 def Minggu():
15     return "Minggu"
16 def default():
17     return "Salah Hari"

```

```

20 switcher = {
21     1: Senin,
22     2: Selasa,
23     3: Rabu,
24     4: Kamis,
25     5: Jumat,
26     6: Sabtu,
27     7: Minggu
28 }
29
30 hari = int(input("Masukan angka hari : "))
31
32 def switch(hari):
33     return switcher.get(hari, default)()
34
35 print(switch(hari))
36

```

Gambar 6.26 Contoh Penggunaan Kontrol Switch-Case menggunakan Dictionary

Perhatikan gambar 6.26, terdapat definisi fungsi yang mengembalikan nilai, contoh : **def Senin()** yang mengembalikan nilai **"Senin"** dalam

bentuk String. terdapat variabel switcher bertipe Dictionary yang berisi pasangan **Key-Value**, contoh : **1: Senin**. Kemudian terdapat variabel **hari** untuk menginput data bertipe **Integer** dari keyboard, Kemudian terdapat fungsi yang akan mengembalikan nama hari dari nilai yang diinput dengan memanggil variabel **switcher.get()**.

Setelah kode program diatas dijalankan dan diinput dengan nilai sebagai berikut :

hari = 3

Maka hasil keluaran yang ditampilkan sebagai berikut :

```
Masukan angka hari : 3
Rabu
```

Gambar 6.27 Contoh Hasil Keluaran Program

c. Cara ke-3 : Menggunakan kelas (akan dipelajari di pertemuan berikutnya).

f. Kontrol Nested IF di bahasa Python

Ada kalanya, kita ingin memeriksa kondisi lain setelah kondisi bernilai benar. Untuk hal seperti itu, maka kita dapat menggunakan kontrol If Bersarang (Nested IF). Disebut demikian, karena terdapat sebuah kondisi lain yang akan dijalankan, apabila kondisi utamanya bernilai benar (True). Struktur control Nested IF bisa dilihat pada gambar 6.28.

```
If kondisi-1 :
    Pernyataan1
    If kondisi-2 :
        Pernyataan_alternatif2
    Elif kondisi-3 :
        Pernyataan_alternatif3
    Elif kondisi4 :
        Pernyataan_alternatif4
    Else
        Pernyataan_alternatif5
Else
    Pernyataan_alternatif6
```

Gambar 6.28 Struktur Kontrol NESTED IF

Studi Kasus 6.8

Program pada gambar 6.29 merupakan contoh implementasi kontrol Nested IF.

```
1  ▶ x = int(input("Masukkan sebuah bilangan : "))
2
3  if x < 200:
4      print(x, " : Nilai kurang dari 200")
5      if x == 150:
6          keterangan = " : Sama dengan 150"
7          print(x, keterangan)
8      elif x == 100:
9          keterangan = " : Sama dengan 100"
10         print(x, keterangan)
11     elif x == 50:
12         keterangan = " : Sama dengan 50"
13         print(x, keterangan)
14     elif x < 50:
15         keterangan = " : Kurang dari 50"
16         print(x, keterangan)
17 else:
18     keterangan = "Nilai yang dimasukkan salah"
19     print(keterangan)
20
21 print("Selesai")
22
```

Gambar 6.29 Contoh Penggunaan Kontrol Nested IF

Perhatikan gambar 6.28, terdapat definisi variabel *x* dengan tipe Integer dengan nilai yang diinput dari keyboard. Kemudian terdapat kontrol IF yang memeriksa nilai *x* apakah kurang dari 200 (***x* < 200**). Jika kondisi Benar (**TRUE**), maka program akan mencetak keterangan = **"Nilai kurang dari 200"**, beserta pemeriksaan kondisi berikutnya yang sesuai dengan nilai *x*. apabila *x* bernilai lebih dari 200 (***x* > 200**), maka kondisinya adalah **False**, sehingga tercetak keterangan = **"Nilai yang dimasukkan salah"** dan **"Selesai"**.

Setelah kode program diatas dijalankan dan diinput dengan nilai-nilai sebagai berikut :

x = 175, kemudian input Kembali nilai x menjadi 250,

Maka hasil keluaran yang ditampilkan sebagai berikut :

```
Masukkan sebuah bilangan : 175
175 : Nilai kurang dari 200
Selesai

Masukkan sebuah bilangan : 250
Nilai yang dimasukkan salah
Selesai
```

Gambar 6.30 Contoh Hasil Keluaran Program

g. Operator AND di dalam kontrol IF

Kita dapat memeriksa beberapa kondisi menggunakan operator **AND**. Kedua kondisi harus terpenuhi agar pernyataan yang diberikan dapat dieksekusi oleh Interpreter Python. Berikut ini sintak umum penulisan kontrol IF dengan operator Logika **AND** di bahasa Python :

```
If kondisi-1 AND kondisi-2:
    Pernyataan1
Elif kondisi-2 AND kondisi-3:
    Pernyataan_alternatif2
Else
    Pernyataan_alternatif3
```

Gambar 6.31 Struktur Kontrol IF-ELIF dengan operator AND

Kita dapat menggunakan struktur kontrol IF, Multiple IF dan IF-ELIF untuk mengimplementasikan penggunaan operator AND menggunakan struktur kontrol diatas.

Tabel 6.3 Rumus Untuk Operator AND

Kondisi-1	Kondisi-2	Hasil : Kondisi-1 AND Kondisi-2
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

Studi Kasus 6.9

Program pada gambar 6.32 merupakan contoh implementasi penggunaan operator AND pada kontrol IF.

```

1  ►
2      x = input("Masukkan nilai x : ")
3      y = int(input("Masukkan nilai y : "))
4
5      # Memeriksa nilai x dan y, apakah keduanya bertipe Integer
6      if type(x) is int and type(y) is int:
7          keterangan = x + y
8          print(keterangan)
9      else:
10         keterangan = "x atau y tidak bertipe Integer"
11         print(keterangan)
12

```

Gambar 6.32 Struktur Kontrol IF-ELIF dengan operator AND

Perhatikan gambar 6.32, terdapat definisi variabel *x* dengan tipe String dengan nilai yang diinput dari keyboard dan variabel *y* bertipe Integer dengan nilai yang diinput dari keyboard. Kemudian terdapat dua kondisi pemeriksaan, yaitu **type(x) is int** dan **type(y) is int**. apabila kedua kondisi tersebut terpenuhi (**True**), maka **keterangan** akan diisi dengan **jumlah antara x dan y**. tetapi jika, salah satu kondisi tidak terpenuhi (**False**), maka **keterangan** akan disini dengan “**x atau y tidak bertipe Integer**”.

Setelah kode program diatas dijalankan dan diinput dengan nilai-nilai sebagai berikut :

x = 5, kemudian input nilai y = 25,

Maka hasil keluaran yang ditampilkan sebagai berikut :

```
Masukkan nilai x : 5
Masukkan nilai y : 25
x atau y tidak bertipe Integer
```

Gambar 6.33 Contoh Hasil Keluaran Program

Perhatikan gambar 6.33, kondisi pertama **type(x) is Integer** tidak terpenuhi (**False**), dikarenakan variabel x bertipe **String** dan kondisi kedua **type(y) is Integer** terpenuhi (**True**), sehingga hasil akhirnya adalah **False** menggunakan operator **AND**, sehingga Python akan mencetak keterangan = **"x atau y tidak bertipe Integer"**.

h. Operator OR di dalam kontrol IF

Kita juga dapat memeriksa beberapa kondisi menggunakan operator OR. Kedua kondisi harus terpenuhi agar pernyataan yang diberikan dapat dieksekusi oleh Interpreter Python. Berikut ini sintak umum penulisan kontrol IF dengan operator Logika **OR** di bahasa Python dapat dilihat pada gambar 6.34.

```
If kondisi-1 OR kondisi-2:
    Pernyataan1
Elif kondisi-2 OR kondisi-3:
    Pernyataan_alternatif2
Else
    Pernyataan_alternatif3
```

Gambar 6.34 Struktur Kontrol IF-ELIF dengan operator OR

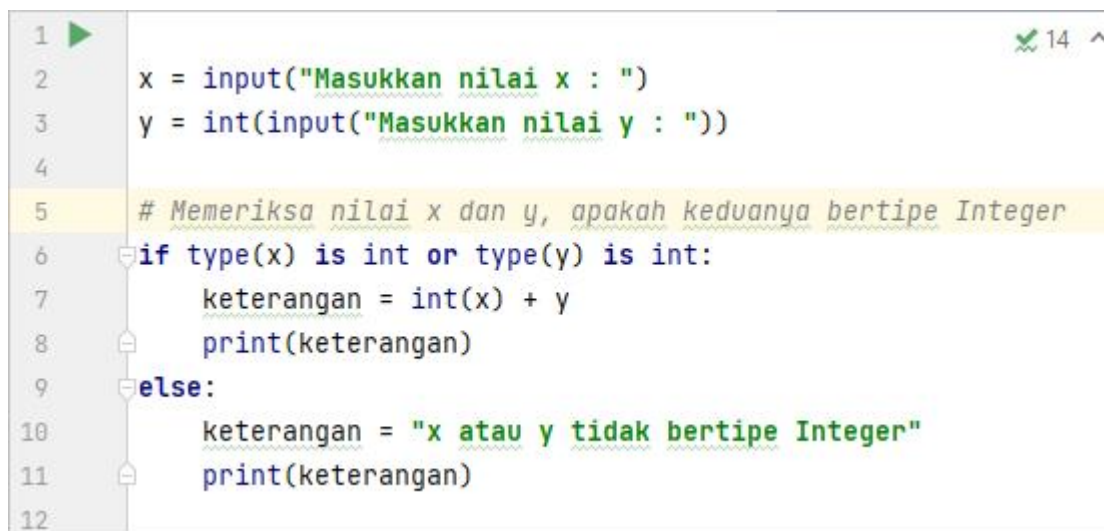
Kita dapat menggunakan struktur kontrol IF, Multiple IF dan IF-ELIF untuk mengimplementasikan penggunaan operator OR menggunakan struktur kontrol diatas.

Tabel 6.4 Rumus Untuk Operator OR

Kondisi-1	Kondisi-2	Hasil : Kondisi-1 OR Kondisi-2
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

Studi Kasus 6.9

Program pada gambar 6.35 merupakan contoh implementasi penggunaan operator OR pada kontrol IF.



```

1  x = input("Masukkan nilai x : ")
2  y = int(input("Masukkan nilai y : "))
3
4
5  # Memeriksa nilai x dan y, apakah keduanya bertipe Integer
6  if type(x) is int or type(y) is int:
7      keterangan = int(x) + y
8      print(keterangan)
9  else:
10     keterangan = "x atau y tidak bertipe Integer"
11     print(keterangan)
12

```

Gambar 6.35 Struktur Kontrol IF-ELIF dengan operator OR

Perhatikan gambar 6.35, terdapat definisi variabel *x* dengan tipe String dengan nilai yang diinput dari keyboard dan variabel *y* bertipe Integer dengan nilai yang diinput dari keyboard. Kemudian terdapat dua kondisi pemeriksaan, yaitu **type(x) is int** dan **type(y) is int**. Apabila salah satu kondisi terpenuhi, maka **keterangan** akan diisi dengan **jumlah antara x dan y**. Tetapi jika, kedua kondisi tidak terpenuhi, maka **keterangan** akan disini dengan "**x atau y tidak bertipe Integer**" akan dicetak.

Setelah kode program diatas dijalankan dan diinput dengan nilai-nilai sebagai berikut :

x = 5, kemudian input nilai y = 25,

Maka hasil keluaran yang ditampilkan sebagai berikut :

```
Masukkan nilai x : 5
Masukkan nilai y : 25
30
```

Gambar 6.36 Contoh Hasil Keluaran Program

Perhatikan gambar 6.36, kondisi pertama **type(x) is Integer** tidak terpenuhi (**False**), dikarenakan variabel x bertipe **String** dan kondisi kedua **type(y) is Integer** terpenuhi (**True**), sehingga hasil akhirnya adalah **TRUE** dikarenakan menggunakan operator **OR**, sehingga aka Python akan mencetak keterangan = "**int(x) + y**" dengan hasil 30.

i. Operator NOT di dalam kontrol IF

Kita juga dapat memeriksa beberapa kondisi menggunakan operator NOT. Operator NOT digunakan untuk membalikkan kondisi TRUE dan FALSE suatu pengujian logika. Operator NOT dapat dikombinasikan dengan operator perbandingan, operator AND dan OR.

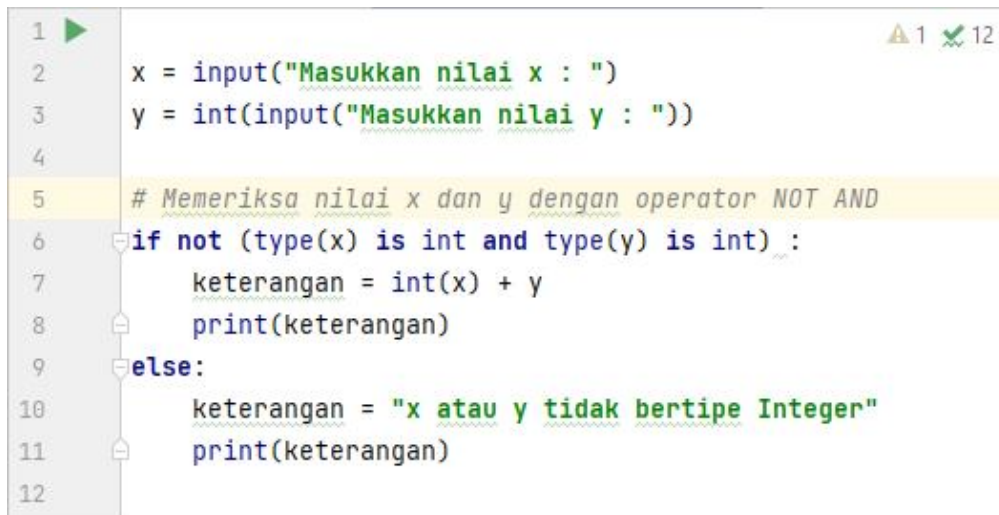
Berikut ini sintak umum penulisan kontrol IF dengan operator Logika **OR** di bahasa Python dapat dilihat pada gambar 6.37.

```
If NOT kondisi-1 :
    Pernyataan1
Else
    Pernyataan_alternatif2
```

Gambar 6.37 Struktur Kontrol IF-ELIF dengan operator NOT

Studi Kasus 6.10

Program pada gambar 6.38 merupakan contoh implementasi penggunaan operator AND dan NOT pada kontrol IF-ELSIF.



```

1  ▶
2  x = input("Masukkan nilai x : ")
3  y = int(input("Masukkan nilai y : "))
4
5  # Memeriksa nilai x dan y dengan operator NOT AND
6  if not (type(x) is int and type(y) is int) :
7      keterangan = int(x) + y
8      print(keterangan)
9  else:
10     keterangan = "x atau y tidak bertipe Integer"
11     print(keterangan)
12

```

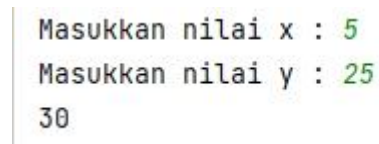
Gambar 6.38 Struktur Kontrol IF-ELSE dengan operator NOT AND

Perhatikan gambar 6.38, terdapat definisi variabel *x* dengan tipe String dengan nilai yang diinput dari keyboard dan variabel *y* bertipe Integer dengan dengan nilai yang diinput dari keyboard. Kemudian terdapat dua kondisi pemeriksaan, yaitu **not (type(x) is int and type(y) is int)**. Hasil dari pemeriksaan **not** tersebut akan bernilai **True** sehingga **keterangan = int(x) + y** akan dikerjakan dan dicetak nilainya. Apabila hasil pemeriksaan **not** tersebut tidak terpenuhi, maka **keterangan "x atau y tidak bertipe Integer"** akan dikerjakan dan dicetak.

Setelah kode program diatas dijalankan dan diinput dengan nilai-nilai sebagai berikut :

x = 5, kemudian input nilai *y* = 25,

Maka hasil keluaran yang ditampilkan sebagai berikut :



```

Masukkan nilai x : 5
Masukkan nilai y : 25
30

```

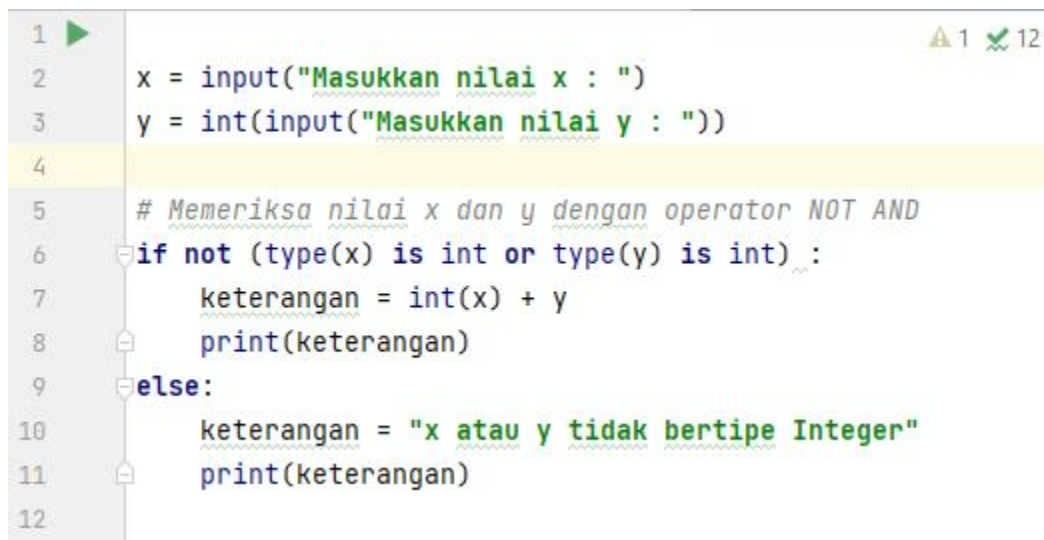
Gambar 6.39 Contoh Hasil Keluaran Program

Perhatikan gambar 6.39, kondisi pertama **type(x) is Integer** tidak memenuhi (**False**), dikarenakan variabel *x* bertipe **String** dan kondisi kedua **type(y) is Integer** terpenuhi (**True**), sehingga kedua kondisi tersebut bernilai **False** dalam operator AND. Kemudian terdapat operator **NOT** yang

berjalan pada operator AND sehingga hasilnya adalah **TRUE** sehingga Python akan mencetak keterangan = **"int(x) + y"** dengan hasil 30.

Studi Kasus 6.11

Program pada gambar 6.40 merupakan contoh implementasi penggunaan operator OR dan NOT pada kontrol IF-ELSIF.



```
1  ▶ x = input("Masukkan nilai x : ")
2
3  y = int(input("Masukkan nilai y : "))
4
5  # Memeriksa nilai x dan y dengan operator NOT AND
6  if not (type(x) is int or type(y) is int) :
7      keterangan = int(x) + y
8      print(keterangan)
9  else:
10     keterangan = "x atau y tidak bertipe Integer"
11     print(keterangan)
12
```

Gambar 6.40 Struktur Kontrol IF-ELSE dengan operator NOT OR

Perhatikan gambar 6.40, terdapat definisi variabel *x* dengan tipe String dengan nilai yang diinput dari keyboard dan variabel *y* bertipe Integer dengan dengan nilai yang diinput dari keyboard. Kemudian terdapat dua kondisi pemeriksaan, yaitu **not (type(x) is int or type(y) is int)**. Hasil dari pemeriksaan **not** tersebut akan bernilai **True** sehingga **keterangan = int(x) + y** akan dikerjakan dan dicetak nilainya. Apabila hasil pemeriksaan **not** tersebut tidak terpenuhi (**False**), maka **keterangan "x atau y tidak bertipe Integer"** akan dikerjakan dan dicetak.

Setelah kode program diatas dijalankan dan diinput dengan nilai-nilai sebagai berikut :

x = 5, kemudian input nilai y = 25,

Maka hasil keluaran yang ditampilkan sebagai berikut :

```
Masukkan nilai x : 5
Masukkan nilai y : 30
x atau y tidak bertipe Integer
```

Gambar 6.41 Contoh Hasil Keluaran Program

Perhatikan gambar 6.41, kondisi pertama **type(x) is Integer** tidak memenuhi (**False**), dikarenakan variabel x bertipe **String** dan kondisi kedua **type(y) is Integer** terpenuhi (**True**), sehingga kedua kondisi tersebut bernilai **True** dalam operator **AND**. Kemudian terdapat operator **NOT** yang berjalan pada operator **AND** sehingga hasilnya adalah **False** sehingga Python akan mencetak keterangan = “**x atau y tidak bertipe Integer**”.

3.2 Praktikum

Langkah-langkah Praktikum

1. Buka Editor Python (IDLE / Pycharm / VSCode).
2. Buatlah file baru dengan membuka menu File > New > Source File atau dengan shortcut Ctrl + N.
3. Tulislah kode program berikut ini :

Program 6.1 : Praktikum61. Py

1. Buatlah program menggunakan bahasa Python untuk mencetak informasi total harga barang yang dibeli dalam jumlah tertentu.

Algoritma / Inisiasi Persoalan :

Variabel input :

kode barang, nama_barang, harga_satuan, jumlah_beli, total_harga

Proses :

total_harga = harga_satuan * jumlah_beli

Output :

total_harga

Aturan bisnis :

- a. Jika total harga lebih besar atau sama dengan Rp. 200.000,-, akan mendapatkan potongan harga sebesar 3% dari total harga.

b. Total bayar dihitung dari total harga dikurangi dengan potongan harga.
Untuk menjawab persoalan diatas, maka tuliskan Program 6.1 berikut ini pada editor :

```

1  ► # deklarasi variabel input
2  kode_barang = input("Masukan kode barang : ")
3  nama_barang = input("Masukan nama barang : ")
4  harga_satuan = input("Masukan harga satuan : ")
5  jumlah_beli = input("Masukan jumlah beli : ")
6
7  # proses
8  total_harga = int(harga_satuan) * int(jumlah_beli)
9  if total_harga >= 200000 :
10     potongan_harga = (3/100) * total_harga
11 else :
12     potongan_harga = 0
13
14 total_bayar = total_harga - potongan_harga
15
16 # Output : menampilkan semua nilai ke layar
17 print("\nData Barang")
18 print("=====")
19 print("Kode barang : ", kode_barang)
20 print("Nama barang : ", nama_barang)
21 print("Harga satuan : ", harga_satuan)
22 print("Jumlah beli : ", jumlah_beli)
23 print("Total harga : ", total_harga)
24 print("Potongan : ", potongan_harga)
25 print("Total bayar : ", total_bayar)
26

```

Gambar 6.42 Keluaran Program Praktikum 61

2. Simpan Program ini dengan nama Praktikum61.py
3. Jalankan program praktikum61 di atas, kemudian input nilai-nilai berikut ini :

Kode barang	: B001
Nama barang	: Kemeja
Harga satuan	: 75000
Jumlah beli	: 5

4. Jalankan program praktikum61 di atas, kemudian tuliskan apa yang tercetak

--

5. Jalankan program praktikum61 di atas, kemudian input nilai-nilai berikut ini :

Kode barang	: B001
Nama barang	: Pensil
Harga satuan	: 7500
Jumlah beli	: 350

6. Jalankan program praktikum61 di atas, lalu tuliskan apa yang tercetak di layar.

--

7. Apakah hasil keluaran program di nomor 4 dan nomor 6 memiliki hasil yang sama, jelaskan!.

--

Program 6.2 : Praktikum62. Py

1. Buatlah program menggunakan bahasa Python untuk mencetak informasi gaji pegawai beserta dengan komisi yang diterima.

Algoritma / Inisiasi Persoalan :

Variabel input :

nip, nama_pegawai, gaji_pokok, golongan, lama_kerja

Proses :

komisi = ..??

total_gaji = gaji_pokok + komisi

Output :

Komisi

Total_gaji

Aturan bisnis :

- a. Jika golongan pegawai adalah "B", "C" atau "D" **dan** lama kerja lebih dari 10 tahun, maka akan mendapatkan komisi sebesar 15% dari total gaji yang diterima.
- b. Jika tidak memenuhi kondisi (a), maka tidak mendapat komisi.

Untuk menjawab persoalan diatas, maka tuliskan Program 6.2 berikut ini pada editor :

```
1  ▶
2  # deklarasi variabel input
3  NIP = input("Masukan id number : ")
4  nama_pegawai = input("Masukan nama pegawai : ")
5  gaji_pokok = int(input("Masukan gaji pokok : "))
6  lama_kerja = int(input("Masukan lama kerja : "))
7  golongan = input("Masukan golongan kerja : ")
8
9  # proses
10 if golongan in("A", "B", "C") and lama_kerja > 10:
11     komisi = (15/100) * gaji_pokok
12 else :
13     komisi = 0
14
15 total_gaji = float(gaji_pokok) + komisi
16
17 # Output : menampilkan semua nilai ke layar
18 print("\nData Gaji Pegawai")
19 print("=====")
20 print("NIP : ", NIP)
21 print("Nama pegawai : ", nama_pegawai)
22 print("Gaji pokok : ", gaji_pokok)
23 print("Komisi : ", komisi)
24 print("Total gaji : ", total_gaji)
25
```

Gambar 6.43 Kode Program Praktikum62

2. Jalankan program praktikum62 di atas, kemudian input nilai-nilai berikut ini :

NIP	: P001
Nama pegawai	: Bambang Jeger
Gaji pokok	: 3500000
Lama Kerja	: 15
Golongan	: B

3. Jalankan program praktikum62 di atas, kemudian tuliskan apa yang tercetak

4. Jalankan program praktikum62 di atas, kemudian input nilai-nilai berikut ini :

Kode barang	: ID321
Nama barang	: Bagus Sajiwo
Gaji pokok	: 4500000.288
Lama Kerja	: 8
Golongan	: B

5. Jalankan program praktikum62 di atas, lalu tuliskan apa yang tercetak di layar.

6. Apakah hasil keluaran program di nomor 3 dan nomor 5 memiliki hasil yang sama, jelaskan!.

Program 6.2 : Praktikum63. Py

7. Buatlah program menggunakan bahasa Python untuk mencetak informasi gaji pegawai beserta dengan komisi yang diterima.

Algoritma / Inisiasi Persoalan :

Variabel input :

nip, nama_pegawai, gaji_pokok, golongan, lama_kerja

Proses :

total_gaji = gaji_pokok + komisi

komisi = ..??

Output :

Total_gaji

Komisi

Aturan bisnis :

- c. Jika golongan pegawai adalah "B", "C" atau "D" atau lama kerja lebih dari 10 tahun, maka akan mendapatkan komisi sebesar 15% dari total gaji yang diterima.
- d. Jika tidak memenuhi kondisi (a), maka tidak mendapat komisi.

Untuk menjawab persoalan diatas, maka tuliskan Program 6.3 berikut ini pada editor :

```
1  # deklarasi variabel input
2  NIP = input("Masukan id number : ")
3  nama_pegawai = input("Masukan nama pegawai : ")
4  gaji_pokok = int(input("Masukan gaji pokok : "))
5  lama_kerja = int(input("Masukan lama kerja : "))
6  golongan = input("Masukan golongan kerja : ")
7
8  # proses
9  if golongan in("C", "D") :
10     if lama_kerja >= 20:
11         komisi = (10/100) * gaji_pokok
12     elif lama_kerja >= 10:
13         komisi = (5/100) * gaji_pokok
14     else :
15         komisi = 0
16  elif golongan in ("A", "B") :
17     if lama_kerja >= 20:
18         komisi = (5/100) * gaji_pokok
19     elif lama_kerja >= 10:
20         komisi = (3/100) * gaji_pokok
21     else :
22         komisi = 0
23  else :
24     komisi = 0
25
26  total_gaji = float(gaji_pokok) + komisi
```



```

27
28 # Output : menampilkan semua nilai ke layar
29 print("\nData Gaji Pegawai")
30 print("=====")
31 print("NIP : ", NIP)
32 print("Nama pegawai : ", nama_pegawai)
33 print("Golongan kerja : ", golongan)
34 print("Gaji pokok : ", gaji_pokok)
35 print("Komisi : ", komisi)
36 print("Total gaji : ", total_gaji)
37

```

Gambar 6.44 Kode Program Praktikum52

8. Jalankan program praktikum63 di atas, kemudian input nilai-nilai berikut ini :

NIP	: P001
Nama pegawai	: Bambang Jeger
Gaji pokok	: 3500000
Lama Kerja	: 15
Golongan	: B

9. Jalankan program praktikum63 di atas, kemudian tuliskan apa yang tercetak

10. Jalankan program praktikum63 di atas, kemudian input nilai-nilai berikut ini :

Kode barang	: ID321
Nama barang	: Bagus Sajiwo
Gaji pokok	: 4500000
Lama Kerja	: 8
Golongan	: -

11. Jalankan program praktikum63 di atas, lalu tuliskan apa yang tercetak di layar.

12. Apakah hasil keluaran program di nomor 3 dan nomor 5 memiliki hasil yang sama, jelaskan!.

3.3 Rangkuman

1. Dengan kontrol IF dalam bahasa Python, maka kita dapat membuat perbandingan logis antara nilai dan apa yang diharapkan dengan menguji kondisi dan mengembalikan hasil berupa benar (True) atau salah (False).
2. Kontrol IF sendiri memiliki beberapa struktur kontrol yang berbeda, yaitu IF digunakan jika hanya untuk satu kondisi seleksi saja, IF-Else digunakan bila memiliki dua kondisi seleksi, Multiple IF dan IF-ELIF-ELSE digunakan bila memiliki banyak kondisi.
3. Memungkinkan juga penggunaan kontrol bersarang di dalam IF (Nested IF), dimana terdapat kondisi seleksi lain di dalam sebuah kondisi seleksi.
4. Kontrol SWITCH-CASE tidak tersedia di dalam bahasa Python, tetapi kita bisa mengganti kontrol tersebut dengan kontrol IF-ELIF-ELSE, tipe Dictionary maupun menggunakan kelas (Class).

3.4 Latihan

Buatlah program untuk mencetak nilai akhir yang diperoleh dari nilai presensi, nilai tugas, nilai UTS dan nilai UAS dengan ketentuan sebagai berikut :

- a. Bobot nilai presensi adalah 10%
- b. Bobot nilai tugas adalah 20%
- c. Bobot nilai UTS adalah 30%
- d. Bobot nilai UAS adalah 40%

Nilai akhir dihitung dengan rumus = $(10\% * \text{nilai presensi}) + (20\% * \text{nilai tugas}) + (30\% * \text{nilai UTS}) + (40\% * \text{nilai UAS})$.

Grade ditentukan berdasarkan nilai akhir sebagai berikut :

- Grade A : 85 – 100.
- Grade A- : 80 – 84
- Grade B+ : 75 – 79
- Grade B : 70 – 74
- Grade C : 60 – 69
- Grade D : 0 - 50

Sehingga apabila diinput nilai-nilai diatas, maka akan menghasilkan tampilan berikut :

```
Data Nilai Akhir
=====
Nilai Presensi : 100
Nilai Tugas : 85
Nilai UTS : 70
Nilai UAS : 90
Nilai Akhir : 84.0
Grade : A-
```

Gambar 6.45 Tampilan keluaran program

3.5 Tugas Mandiri

Kerjakan soal-soal berikut :

1. Jelaskan perbedaan kontrol Multiple IF dan IF-ELIF-ELSE?
2. Apakah mungkin membuat kontrol bersarang (Nested IF) di dalam kontrol IF-ELIF-ELSE? Jelaskan!
3. Buatlah program dengan Python untuk mencari harga produk paling murah (terkecil) dari 3 harga yang diinput (harga produk harus bilangan bulat sembarang).

Pertanyaan :

- a. Tentukan input / output prosesnya.
- b. Buatlah kode program untuk menyelesaikan permasalahan tersebut.
- c. Jalankan kode program pada soal (b), temukan dan selesaikan masalah yang terjadi!

Contoh keluaran program :

Harga 1 = 3500

Harga 2 = 4200

Harga 3 = 3800

Harga terkecil adalah harga ke-3 yaitu 3800.

4. Buatlah program dengan Python untuk menghitung harga layanan yang dikenakan kepada member :

Program hanya menerima inputan berupa kode member, nama member, biaya layanan, lama menjadi member. Untuk masing-masing member akan diberikan diskon berdasarkan lama bergabung menjadi member. Diskon ditentukan sebagai berikut :

- Lama member lebih besar atau sama dengan 20 tahun, diberikan diskon 15% dari biaya layanan.
- Lama member lebih besar atau sama dengan 10 tahun, diberikan diskon 10% dari biaya layanan.
- Lama member lebih besar atau sama dengan 5 tahun, diberikan diskon 5% dari biaya layanan.
- Lama member kurang dari 5 tahun, tidak diberikan diskon.

Pertanyaan :

- a. Tentukan input / output prosesnya.
- b. Buatlah kode program untuk menyelesaikan permasalahan tersebut.
- c. Jalankan kode program pada soal (b), temukan dan selesaikan masalah yang terjadi!



FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BUDI LUHUR

Jl. Raya Ciledug, Petukangan Utara, Pesanggrahan
Jakarta Selatan, 12260

Telp: 021-5853753 Fax : 021-5853752

<http://fti.budiluhur.ac.id>