

MODUL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK

Minggu 1



PENGENALAN DAN DASAR PEMROGRAMAN PYTHON I

Disusun Oleh :

| | |
|------------------------|-----------|
| Andhika Putra Pratama | 119140224 |
| Andhika Wibawa B. | 118140028 |
| Nurul Aulia Larasati | 119140088 |
| Ihza Fajrur Rachman H. | 119140130 |
| Enrico Johanes S. | 119140021 |
| M. Ammar Fadhila R. | 119140029 |

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK ELEKTRO, INFORMATIKA, DAN SISTEM FISIS
INSTITUT TEKNOLOGI SUMATERA

2022

1. Pengenalan Bahasa Pemrograman Python

Bahasa pemrograman python dibuat oleh Guido Van Rossum pada tahun 1980-an akhir di *Centrum Wiskunde & Informatica* Belanda. Python mendukung banyak paradigma pemrograman seperti object-oriented, functional dan structured. Bahasa Python menjadi populer sekarang ini dikarenakan sintaks nya yang mudah, didukung oleh library(modul) yang berlimpah dan Python bisa dipakai untuk pemrograman desktop maupun mobile, CLI, GUI, web, otomatisasi, hacking, IoT, robotika, dan lain sebagainya.

Dalam dokumen *The Zen of Python* (PEP 20), disitu tertera filosofi inti dari bahasa pemrograman Python :

- *Beautiful is better than ugly.*
- *Explicit is better than implicit.*
- *Simple is better than complex.*
- *Complex is better than complicated.*
- *Readability counts.*

Python sangat mementingkan *readability* pada kode, untuk mengimplementasikan filosofi itu Python tidak menggunakan kurung kurawal({}) atau *keyword* (ex. **start**, **begin**, **end**) sebagai gantinya menggunakan spasi(*white space*) untuk memisahkan blok-blok kode

Berikut ini cara untuk melakukan instalasi python:

- Linux/Unix

Secara default Python sudah terinstal pada sistem operasi linux, untuk melihat apakah sudah terinstal, ketikkan perintah dibawah pada terminal

```
fadila-amin@fadilaamin:~$ python3 --version
Python 3.6.9
```

jika belum silahkan install dengan

```
fadila-amin@fadilaamin:~$ sudo apt install python3
```

- Windows

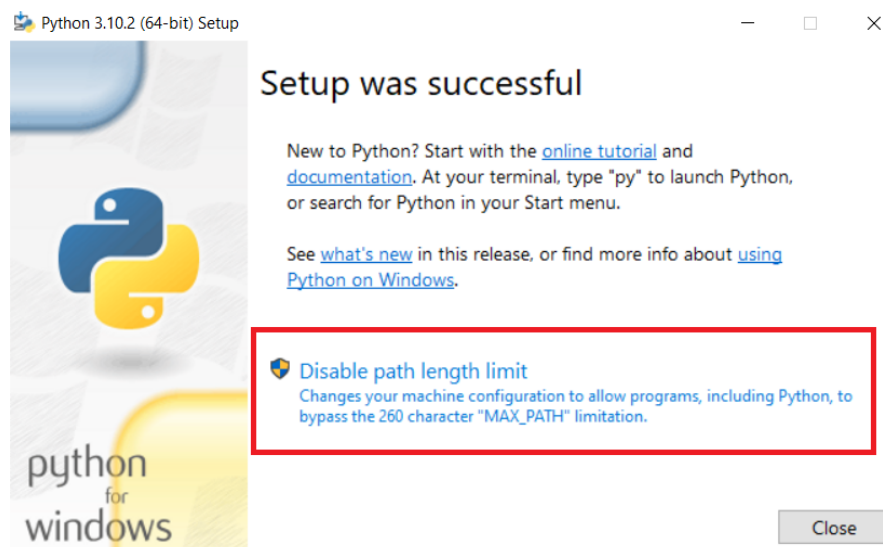
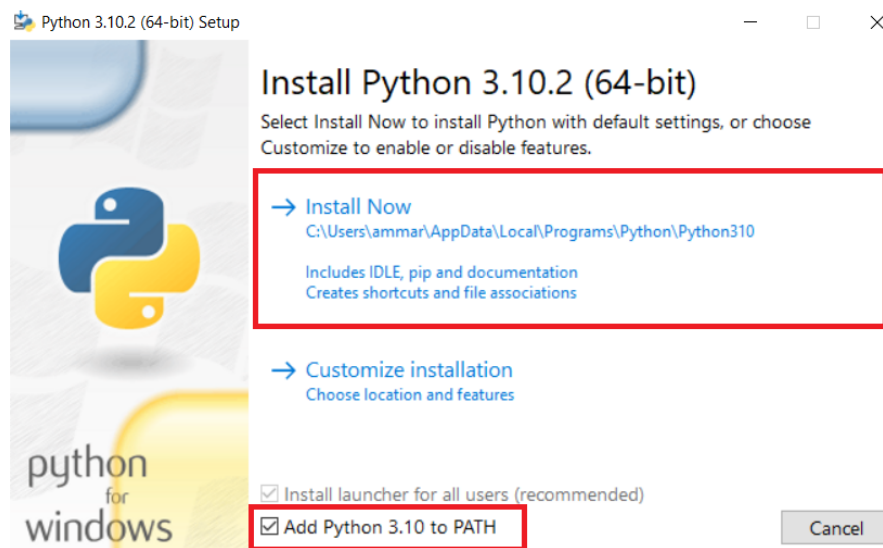
1. download python pada link berikut:

<https://www.python.org/downloads/release/python-3102/>

2. lalu scroll ke halaman terbawah
3. klik windows installer (32 bit) atau windows installer (64 bit) pada tabel

| Files | | | | | |
|---|------------------|--------------------------|----------------------------------|-----------|---------------------|
| Version | Operating System | Description | MD5 Sum | File Size | GPG |
| Gzipped source tarball | Source release | | 67c92270be6701f4a6fed57c4530139b | 25067363 | SIG |
| XZ compressed source tarball | Source release | | 14e8c22458ed7779a1957b26cde01db9 | 18780936 | SIG |
| macOS 64-bit universal2 installer | macOS | for macOS 10.9 and later | edced8c45edc72768f03f66cf4b4fa27 | 39805121 | SIG |
| Windows embeddable package (32-bit) | Windows | | 44875e70945bf45f655f61bb82dba211 | 7541211 | SIG |
| Windows embeddable package (64-bit) | Windows | | f98f8d7dfa952224fca313ed8e9923d8 | 8509629 | SIG |
| Windows help file | Windows | | 342cabb615e5672e38c9906a3816d727 | 9575352 | SIG |
| Windows installer (32-bit) | Windows | | ef91f4e873280d37eb5bc26e7b18d3d1 | 27072760 | SIG |
| Windows installer (64-bit) | Windows | Recommended | 2b4fd1ed6e736f0e65572da64c17e020 | 28239176 | SIG |

4. lalu buka file yang telah didownload
5. ikuti instruksi pada gambar dibawah ini



buka command prompt lalu ketik “python --version” tanpa tanda kutip dan enter

6. lalu akan menampilkan versi python yang terinstall

```
C:\Windows\system32>python --version  
Python 3.10.2
```

Jika versi python yang ditampilkan adalah versi python yang lain, maka python yang terbaru harus ditambah ke PATH secara manual.

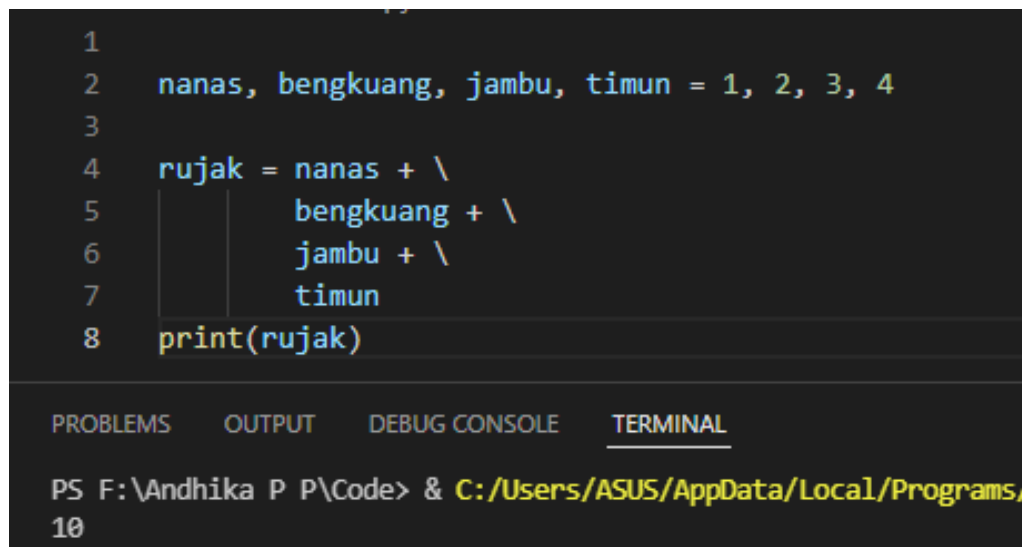
2. Dasar Pemrograman Python

2.1 Sintaks Dasar

- **Statement**

Semua perintah yang bisa dieksekusi Python disebut statement. Pada Python akhir dari sebuah statement adalah baris baru (*newline*) tapi dimungkinkan membuat statement yang terdiri dari beberapa baris menggunakan *backslash* (\).

- **Baris dan Indentasi**

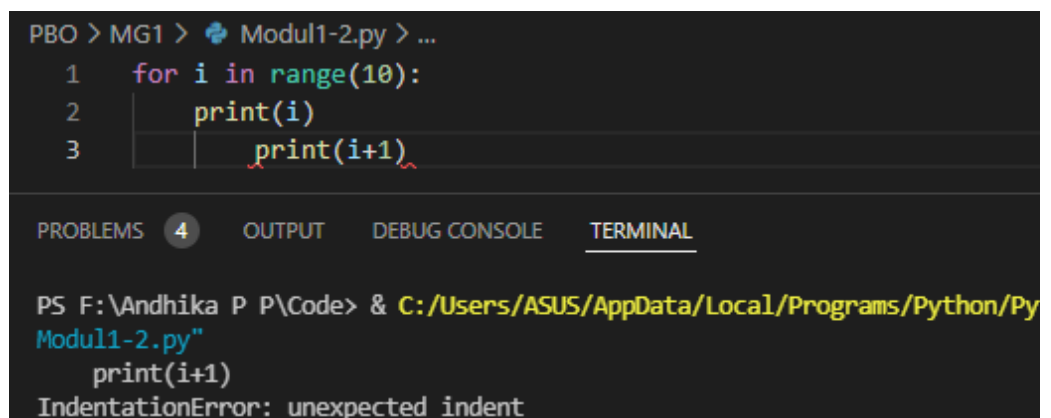


```
1
2  nanas, bengkuang, jambu, timun = 1, 2, 3, 4
3
4  rujak = nanas + \
5         bengkuang + \
6         jambu + \
7         timun
8  print(rujak)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS F:\Andhika P P\Code> & C:/Users/ASUS/AppData/Local/Programs/Python/Python39-64/python.exe F:\Andhika P P\Code\modul1-2.py

Python tidak menggunakan kurung kurawal sebagai *grouping* blok kode melainkan menggunakan spasi ataupun tab (4 spasi). kode yang berada di blok yang sama harus memiliki jumlah spasi yang sama di awal. Contoh salah:



```
PBO > MG1 > Modul1-2.py > ...
1  for i in range(10):
2      print(i)
3      print(i+1)
```

PROBLEMS **4** OUTPUT DEBUG CONSOLE TERMINAL

PS F:\Andhika P P\Code> & C:/Users/ASUS/AppData/Local/Programs/Python/Python39-64/python.exe F:\Andhika P P\Code\modul1-2.py

print(i+1)
IndentationError: unexpected indent

2.2 Variabel dan Tipe Data Primitif

Variabel merupakan lokasi penyimpanan yang berguna untuk menyimpan suatu data atau suatu nilai. Dalam mendeklarasikan suatu variabel dalam pemrograman, perlu diketahui tipe-tipe data yang berhubungan dengan variabel

yang akan dideklarasikan, di bawah ini merupakan tipe data yang ada :

| Tipe Data | Jenis | Nilai |
|-----------|----------------|------------------------|
| bool | Boolean | True atau false |
| int | Bilangan bulat | Seluruh bilangan bulat |
| float | Bilangan real | Seluruh bilangan real |
| string | Teks | Kumpulan karakter |

Contoh pendeklarasian variabel dalam Python :

```
var1 = True # Boolean
angka1 = 10 # Integer
desimal = 3.14 # Float
huruf1 = 'Praktikum Pemrograman Python' # String
```

2.3 Operator

Python memiliki sejumlah operator, yaitu :

2.3.1 Operator Aritmatika

Operator aritmatika adalah operator yang digunakan untuk melakukan operasi matematika, seperti penjumlahan, pengurangan, perkalian, pembagian, dan sebagainya. Tabel berikut menunjukkan jenis operator aritmatika:

| Operator | Nama dan Fungsi | Contoh |
|----------|--|---------|
| + | Penjumlahan, menjumlahkan 2 buah operand | $x + y$ |
| - | Pengurangan, mengurangi 2 buah operand | $x - y$ |
| * | Perkalian, mengalikan 2 buah operand | $x * y$ |
| / | Pembagian, membagi 2 buah operand | x / y |

| | | |
|----|---|--------|
| ** | Pemangkatan, memangkatkan bilangan | x **y |
| // | Pembagian bulat, menghasilkan hasil bagi tanpa koma | x // y |
| % | Modulus, menghasilkan sisa pembagian 2 bilangan | x % y |

contoh :

```

1  Num1 = int(input("Angka Pertama: "))
2  Num2 = int(input("Angka Kedua: "))
3  print("Hasil: ", Num1 + Num2)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

Angka Pertama: 10
Angka Kedua: 20
Hasil: 30

```

```

1  Num1 = int(input("Angka Pertama: "))
2  Num2 = int(input("Angka Kedua: "))
3  print("Hasil: ", Num1 // Num2)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

Angka Pertama: 32
Angka Kedua: 5
Hasil: 6

```

2.3.2 Operator Perbandingan

Operator perbandingan adalah operator yang digunakan untuk membandingkan 2 buah nilai. Hasil perbandingannya adalah True atau False tergantung kondisi.

| Operator | Nama dan Fungsi | Contoh |
|----------|-----------------|--------|
|----------|-----------------|--------|

| | | |
|----|---|----------|
| > | Lebih besar dari – Hasilnya True jika nilai sebelah kiri lebih besar dari nilai sebelah kanan | $x > y$ |
| < | Lebih kecil dari – Hasilnya True jika nilai sebelah kiri lebih kecil dari nilai sebelah kanan | $x < y$ |
| == | Sama dengan – Hasilnya True jika nilai sebelah kiri sama dengan nilai sebelah kanan | $x == y$ |
| != | Tidak sama dengan – Hasilnya True jika nilai sebelah kiri tidak sama dengan nilai sebelah kanan | $x != y$ |
| >= | Lebih besar atau sama dengan – Hasilnya True jika nilai sebelah kiri lebih besar atau sama dengan nilai sebelah kanan | $x >= y$ |
| <= | Lebih kecil atau sama dengan – Hasilnya True jika nilai sebelah kiri lebih kecil atau sama dengan nilai sebelah kanan | $x <= y$ |

Contoh:

```

1
2  Num1 = int(input("Angka Pertama: "))
3  Num2 = int(input("Angka Kedua: "))
4  lebih_kecil = Num1 < Num2
5  print("Hasil: ", lebih_kecil)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

Angka Pertama: 2
Angka Kedua: 3
Hasil: True

```


2.3.3 Operator Penugasan

Operator penugasan adalah operator yang digunakan untuk memberi nilai ke variabel. $a = 7$ adalah contoh operator penugasan yang memberi nilai 7 di kanan ke variabel a yang ada di kiri.

| Operator | Penjelasan | Contoh |
|----------|--|--|
| $=$ | Menugaskan nilai yang ada di kanan ke operand di sebelah kiri | $c = a + b$ menugaskan $a + b$ ke c |
| $+=$ | Menambahkan operand yang di kanan dengan operand yang ada di kiri dan hasilnya ditugaskan ke operand yang di kiri | $c += a$ sama dengan $c = c + a$ |
| $-=$ | Mengurangi operand yang di kanan dengan operand yang ada di kiri dan hasilnya ditugaskan ke operand yang di kiri | $c -= a$ sama dengan c $= c - a$ |
| $*=$ | Mengalikan operand yang di kanan dengan operand yang ada di kiri dan hasilnya ditugaskan ke operand yang di kiri | $c *= a$ sama dengan $c = c * a$ |
| $/=$ | Membagi operand yang di kanan dengan operand yang ada di kiri dan hasilnya ditugaskan ke operand yang di kiri | $c /= a$ sama dengan c $= c / a$ |
| $**=$ | Memangkatkan operand yang di kanan dengan operand yang ada di kiri dan hasilnya ditugaskan ke operand yang di kiri | $c **= a$ sama dengan $c = c ** a$ |
| $//=$ | Melakukan pembagian bulat operand di kanan terhadap operand di kiri dan hasilnya disimpan di operand yang di kiri | $c //= a$ sama dengan $c = c // a$ |
| $\%=$ | Melakukan operasi sisa bagi operand di | $c \% = a$ sama dengan |

| | | |
|--|--|------------------|
| | kanan dengan operand di kiri dan hasilnya disimpan di operand yang di kiri | c = c % a |
|--|--|------------------|

Contoh:

```
1 Num1 = int(input("Angka Pertama: "))
2 Num2 = 10
3 Num2 -= Num1
4 print("Hasil: ", Num2)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Angka Pertama: 30
Hasil: -20

2.3.4 Operator Logika

Operator logika adalah operator yang digunakan untuk melakukan operasi logika.

| Operator | Penjelasan | Contoh |
|----------|---|---------|
| and | Hasilnya adalah True jika kedua operandnya bernilai benar | x and y |
| or | Hasilnya adalah True jika salah satu atau kedua operandnya bernilai benar | x or y |
| not | Hasilnya adalah True jika operandnya bernilai salah (kebalikan nilai) | not x |

Contoh:

```

1  #Operator Logika
2  X = 10
3  Y = 20
4  Operator1 = Y and Y<15
5  print("Hasil Kondisi Operator1: ", Operator1)
6
7  Operator2 = X or Y>15
8  print("Hasil Kondisi Operator2: ", Operator2)
9
10 Operator3 = X and Y>15 or X and Y<11
11 print("Hasil Kondisi Operator3: ", Operator3)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

Hasil Kondisi Operator1: False
Hasil Kondisi Operator2: 10
Hasil Kondisi Operator3: True

```

2.3.5 Operator Bitwise

Operator bitwise adalah operator yang melakukan operasi bit terhadap operand. Operator ini beroperasi bit per bit sesuai dengan namanya. Sebagai misal, angka 2 dalam bit ditulis 10 dalam notasi biner dan angka 7 ditulis 111. Pada tabel di bawah ini, misalkan $x = 10$ (0000 1010) dalam biner dan $y = 4$ (0000 0100) dalam biner.

| Operator | Nama | Contoh |
|----------|---------------------|-------------------------------|
| & | Bitwise AND | $x \& y = 0$ (0000 0000) |
| | Bitwise OR | $x y = 14$ (0000 1110) |
| ~ | Bitwise NOT | $\sim x = -11$ (1111 0101) |
| ^ | Bitwise XOR | $x \wedge y = 14$ (0000 1110) |
| >> | Bitwise right shift | $x \gg 2 = 2$ (0 000 0010) |
| << | Bitwise left shift | $x \ll 2 = 40$ (0010 1000) |

Contoh:

```
1
2  Num1 = int(input("Angka Pertama: "))
3  Num2 = int(input("Angka Kedua: "))
4  bitwise = Num1 & Num2
5  print("Hasil: ", bitwise)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Angka Pertama: 10
Angka Kedua: 12
Hasil: 8

2.3.6 Operator Identitas

Operator identitas adalah operator yang memeriksa apakah dua buah nilai (atau variabel) berada pada lokasi memori yang sama.

| Operator | Penjelasan | Contoh |
|----------|--|----------------------|
| is | True jika kedua operand identik (menunjuk ke objek yang sama) | x is True |
| is not | True jika kedua operand tidak identik (tidak merujuk ke objek yang sama) | x is not True |

Contoh:

```
1  A='Praktikum'
2  B='Praktikum'
3  C='PBO'
4
5  print(" A is B: ", A is B)
6  print(" A is not C: ", A is not C)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

A is B: True
A is not C: True

2.3.7 Operator Keanggotaan

Operator keanggotaan adalah operator yang digunakan untuk memeriksa apakah suatu nilai atau variabel merupakan anggota atau ditemukan di dalam suatu data (string, list, tuple, set, dan dictionary).

| Operator | Penjelasan | Contoh |
|----------|--|------------|
| in | True jika nilai/variabel ditemukan di dalam data | 5 in x |
| not in | True jika nilai/variabel tidak ada di dalam data | 5 not in x |

contoh:

```

1  A='Praktikum'
2  B='Praktikum PBO'
3  C='PBO'
4
5  print(" A in B: ", A in B)
6  print(" A not in C: ", A not in C)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

A in B:  True
A not in C:  True

```

2.4 Tipe Data Bentukan

Ada 4, dengan perbedaan penggunaan:

- List
Sebuah kumpulan data yang teratur, dapat diubah, dan memungkinkan ada anggota yang sama
- Tuple
Sebuah kumpulan data yang teratur, tidak dapat diubah, dan memungkinkan ada anggota yang sama
- Set
Sebuah kumpulan data yang tidak berurutan, tidak terindeks, dan tidak memungkinkan ada anggota yang sama
- Dictionary
Sebuah kumpulan data yang tidak berurutan, dapat diubah, tidak memungkinkan ada anggota yang sama

| Tipe data dasar | Contoh nilai | Penjelasan |
|-----------------|--|---|
| List | [1, 2, 3, 4, 5] atau ['apple', 'banana', 'cherry'] atau ['xyz', 768, 2.23] | Data untaian yang menyimpan berbagai tipe data dan isinya bisa diubah-ubah |
| Tuple | ('xyz', 1, 3.14) | Data untaian yang menyimpan berbagai tipe data tapi isinya tidak bisa diubah |
| Dictionary | { 'firstName': 'Joko', 'lastName': 'Widodo' } | Data untaian yang menyimpan berbagai tipe data berupa pasangan penunjuk dan nilai |
| Set | { 'apple', 'banana', 'cherry' } | Data untaian yang menyimpan berbagai tipe data dan elemen datanya harus unik |

2.5. Percabangan

Dalam bahasa pemrograman Python, terdapat beberapa percabangan yaitu IF, IF-ELSE, dan IF-ELSE-IF.

a. Percabangan IF

Sebagai contoh, akan dibuat suatu program dengan bahasa Python untuk menentukan bilangan negatif.

```

1  #menentukan bilangan positif dan negatif
2
3  print("Masukkan nilai N: ", end="")
4  N = int(input())
5
6  if(N<0):
7      print(str(N) + " bilangan negatif")

```

Kode di atas hanya bisa menentukan bilangan negatif saja, tanpa menentukan bilangan lainnya yaitu positif dan nol. Jika pengguna memasukkan angka positif atau nol, program tidak mengeluarkan keluaran

apapun

b. Percabangan IF-ELSE

Untuk mengeluarkan output ketika user memasukkan selain angka negatif, perlu ditambahkan percabangan lagi. Kita dapat mengimplementasikan percabangan IF-ELSE.

```
1 #menentukan bilangan positif dan negatif
2
3 print("Masukkan nilai N: ", end="")
4 N = int(input())
5
6 if(N<0):
7     print(str(N) + " bilangan negatif")
8 else:
9     print(str(N) + " bilangan positif")
```

Penambahan line 8-9 akan mengeluarkan keluaran untuk bilangan positif. Namun, kode di atas tidak dapat mengeluarkan keluaran bila pengguna memasukkan bilangan nol. Agar program bisa menentukan bilangan nol, kita dapat menggunakan percabangan IF-ELSE-IF.

c. Percabangan IF-ELSE-IF

Kita dapat menggunakan percabangan IF-ELSE-IF untuk melengkapi program sebelumnya.

```
1 #menentukan bilangan positif dan negatif
2
3 print("Masukkan nilai N: ", end="")
4 N = int(input())
5
6 if(N<0):
7     print(str(N) + " bilangan negatif")
8 elif(N==0):
9     print(str(N) + " bilangan nol")
10 else:
11     print(str(N) + " bilangan positif")
```

d. Nested IF

Selain menggunakan code seperti gambar diatas untuk menentukan bilangan positif, negatif atau bilangan nol, kita dapat mengubah format code menjadi percabangan bersarang (nested if) dengan menempatkan percabangan di dalam percabangan. Kode program dapat dilihat seperti gambar dibawah ini.

```

1 #menentukan bilangan positif dan negatif
2
3 print("Masukkan nilai N: ", end="")
4 N = int(input())
5
6 if(N<=0):
7     if(N==0):
8         print(str(N) + " bilangan nol")
9     else:
10        print(str(N) + " bilangan negatif")
11
12 else:
13     print(str(N) + " bilangan positif")

```

2.6 Perulangan

Dalam python terdapat dua jenis perulangan , yaitu perulangan for dan perulangan while. Dalam implementasi nya perulangan digunakan jika ingin mengulang sesuatu sebanyak n kali.

2.6.1 Perulangan For

Pada perulangan for biasa digunakan untuk iterasi pada urutan berupa list, tuple, atau string. Sintaks dasar pada perulangan for di python:

```

for i in (kondisi):
    #perintah

```

contoh :

- Perulangan for pada list

```

1 #contoh perulangan for pada list
2 namaMahasiswa = ["Joko", "Budi", "Bambang", "Eka"]
3 for i in namaMahasiswa:
4     print(i)

```

output :

```

Joko
Budi
Bambang
Eka

```

- Perulangan for pada string

```

1 #contoh perulangan for pada string
2 for i in "Mengoding":
3     print(i)

```


output:

```
M  
e  
n  
g  
o  
d  
i  
n  
g
```

- Perulangan for pada tuple

```
1 #contoh perulangan for pada tuple  
2 bunga = ("mawar", "melati", "kamboja")  
3 for i in bunga:  
4     print(i)
```

output :

```
mawar  
melati  
kamboja
```

Untuk menggunakan perulangan for dapat juga dilakukan dengan menggunakan indeks atau range. Dimana range akan mengembalikan nilai dari 0 kemudian bertambah 1 sampai batas nilai yang ditentukan.

Sintaks umum penggunaan range pada for :

1. Menggunakan satu parameter

```
for i in range(x):  
    #lakukan sesuatu
```

keterangan : Perulangan akan dilakukan dari indeks 0 sampai kurang dari x.

2. Menggunakan 2 parameter

```
for i in range(x,y):  
    #lakukan sesuatu
```

keterangan : Perulangan akan dilakukan dari indeks ke-x sampai kurang dari y.

3. Menggunakan 3 parameter

```
for i in range(x,y,z):  
    #lakukan sesuatu
```

keterangan : Perulangan akan dilakukan dari indeks ke-x

sampai kurang dari y dengan indeks bertambah/increment sejumlah z.

contoh (mengakses tuple dengan menggunakan indeks) :

```
1 #contoh penggunaan range
2 bunga = ("mawar", "melati", "kamboja")
3 for i in range(len(bunga)):
4     print("bunga ke", i, "adalah " , bunga[i])
```

output :

```
bunga ke 0 adalah mawar
bunga ke 1 adalah melati
bunga ke 2 adalah kamboja
```

Dalam for juga dikenal penggunaan break dan continue. Statement break digunakan ketika ingin keluar dari baris perulangan sedangkan continue digunakan ketika ingin melewati kondisi tertentu dan melanjutkan ke baris program selanjutnya.

```
1 #contoh penggunaan break
2 bunga = ("mawar", "melati", "kamboja")
3
4 for i in range(len(bunga)):
5     if(bunga[i]=="melati"):
6         print("ada bunga melati")
7         break
8 #cek indeks terakhir
9 print("indeks terakhir yang diakses :", i)
```

contoh penggunaan break (mencari bunga “melati” pada tuple bunga):

output:

```
ada bunga melati
indeks terakhir yang diakses : 1
```

contoh penggunaan continue (melewati spasi pada kalimat “Teknik Informatika”):

```
1 #contoh penggunaan continue
2
3 for huruf in "Teknik Informatika":
4     if(huruf==" "):
5         continue
6     print(huruf)
```

output :

```
T  
e  
k  
n  
i  
k  
I  
n  
f  
o  
r  
m  
a  
t  
i  
k  
a
```

2.6.2 Perulangan While

Dengan menggunakan while maka dapat dilakukan perulangan selama kondisi tertentu terpenuhi. sintaks umum pada perulangan

```
while(kondisi):  
    #lakukan sesuatu
```

while :

contoh (menghitung karakter c sebelum tanda !) :

```
1  #menghitung karakter c sebelum tanda !  
2  
3  count=int(0)  
4  
5  kalimat=(input())  
6  
7  while(kalimat!='!'):  
8  
9      if kalimat=='c':  
10         count+=1  
11         kalimat=(input())  
12  
13  print("jumlah karakter c adalah ", count)  
14
```

output :

```
b
c
c
a
d
!
jumlah karakter c adalah 2
```

2.7 Fungsi

Dengan menggunakan fungsi kita dapat mengeksekusi suatu blok kode tanpa harus menulisnya berulang-ulang. Contoh:

```
def kelulusan(nama, nilai):
    if nilai ≥ 40:
        return f"{nama} lulus dengan nilai {nilai}"
    else:
        return f"{nama} mengulang dengan nilai {nilai}"

list_mhs = [
    ["Bambang", 70], ["Anton", 55], ["Budi", 85],
    ["Rani", 75], ["Siti", 35], ["Aulia", 90]
]

for i in list_mhs:
    print(kelulusan(i[0], i[1]))
```

Hasil:

```
/run/media/dhika/Multi/Project/Praktikum/PBO
Bambang lulus dengan nilai 70
Anton lulus dengan nilai 55
Budi lulus dengan nilai 85
Rani lulus dengan nilai 75
Siti mengulang dengan nilai 35
Aulia lulus dengan nilai 90
```

Dalam fungsi juga dapat digunakan parameter default yang dapat dilihat hasilnya saat pemanggilan fungsi dengan parameter tidak lengkap:

```
def kelulusan(nama, nilai, matkul = "PBO"):
    if nilai ≥ 40:
        return f"{nama} lulus {matkul} dengan nilai {nilai}"
    else:
        return f"{nama} mengulang {matkul} dengan nilai {nilai}"

print(kelulusan("Anton", 80))
print(kelulusan("Budi", 90, "Kewirausahaan"))
```

Hasil:

```
run /run/media/dhika/Multi/Project/Praktikum/PBO
Anton lulus PBO dengan nilai 80
Budi lulus Kewirausahaan dengan nilai 90
```

Fungsi di Python juga memungkinkan pendeteksian banyak parameter secara otomatis melalui parameter spesial `*args` (arguments) dan `**kwargs` (keyworded arguments). Dalam `*args`, parameter fungsi dianggap sebagai satu kesatuan **list**, sedangkan dalam `**kwargs` parameter fungsi dianggap sebagai satu kesatuan **dictionary**.

Contoh penggunaan `*args`:

```
def kelulusan(*args):
    if args[1] >= 40:
        return f"{args[0]} lulus dengan nilai {args[1]}"
    else:
        return f"{args[0]} mengulang dengan nilai {args[1]}"

def jumlah_total(*angka):
    hasil = 0
    for i in angka:
        hasil += i
    return hasil

print(kelulusan("Budi", 90))
# Hasil: Budi lulus dengan nilai 90

print(jumlah_total(5,10,15))
# Hasil: 30
```

Contoh penggunaan `**kwargs`:

```
def jumlah_total(**angka):
    hasil = 0
    for i in angka.values():
        hasil += i
    return hasil

print(jumlah_total(a = 5, b = 10, c = 15))
# Hasil: 30
```

Latihan

1. Buatlah program yang dapat menerima n input bilangan integer dari user kemudian menghasilkan output kotak bintang dengan panjang n, dan lebar n. contoh input dan output:

- input oleh user 5

```
5
*****
*****
*****
*****
*****
```

- input oleh user 3

```
3
***
***
***
```

2. Buatlah program yang dapat menerima inputan username dan password dari user sebanyak 3 kali. Jika percobaan oleh user kurang dari sama dengan 3 kali maka berhasil login, jika lebih dari 3 kali maka akun diblokir. Dengan username yang digunakan ialah seperti berikut:

username = informatika

password = 12345678

contoh input dan output program:

- ketika berhasil

```
Username anda :informatika
Password anda : 12345678
Berhasil login!
```

- ketika gagal

```
Username anda :informa
Password anda : 1234
Username atau password salah coba lagi
Username anda :informatika
Password anda : 12345678
Berhasil login!
```

Referensi:

https://python101.pythonlibrary.org/chapter5_loops.html

https://www.w3schools.com/python/python_while_loops.asp

https://www.w3schools.com/python/python_for_loops.asp

https://www.w3schools.com/python/gloss_python_bitwise_operators.asp