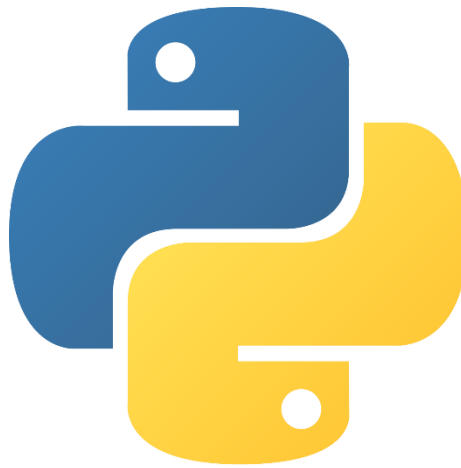


# MODUL BAHASA PEMROGRAMAN PYTHON





## Pengantar

**P**ython adalah bahasa pemrograman interpretatif multiguna. Tidak seperti bahasa lain yang susah untuk dibaca dan dipahami, python lebih menekankan pada keterbacaan kode agar lebih mudah untuk memahami sintaks. Hal ini membuat Python sangat mudah dipelajari baik untuk pemula maupun untuk yang sudah menguasai bahasa pemrograman lain.

Bahasa ini muncul pertama kali pada tahun 1991, dirancang oleh seorang bernama Guido van Rossum. Sampai saat ini Python masih dikembangkan oleh Python Software Foundation. Bahasa Python mendukung hampir semua sistem operasi, bahkan untuk sistem operasi Linux, hampir semua distronya sudah menyertakan Python di dalamnya.

Dengan kode yang simpel dan mudah diimplementasikan, seorang programmer dapat lebih mengutamakan pengembangan aplikasi yang dibuat, bukan malah sibuk mencari syntax error.



# INSTALASI DAN PENGENALAN



## INSTALASI 1

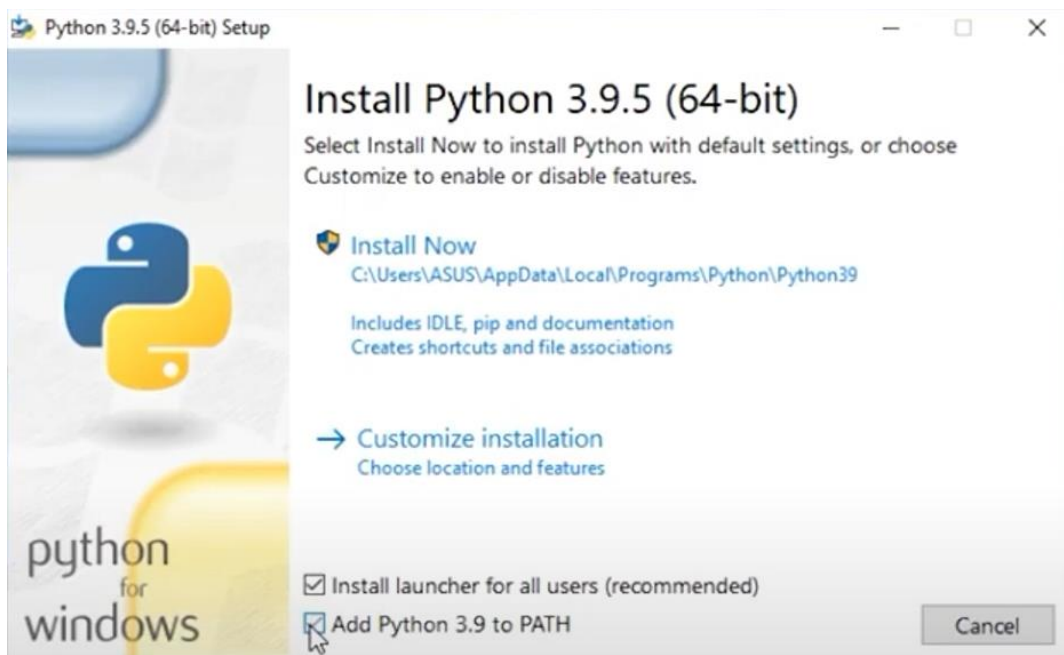
Download dan install python 3.10.4 pada link berikut : <https://www.python.org/downloads/>



Setelah python terunduh ke komputer atau laptop yang digunakan, lanjut proses install python.

Langkah-langkah instalasi:

- Buka folder tempat setup python yang telah didownload sebelumnya
- Klik 2x pada file setup untuk memulai proses instalasi
- Jika muncul pop up seperti pada gambar berikut, klik centang **"Add Python to PATH"**

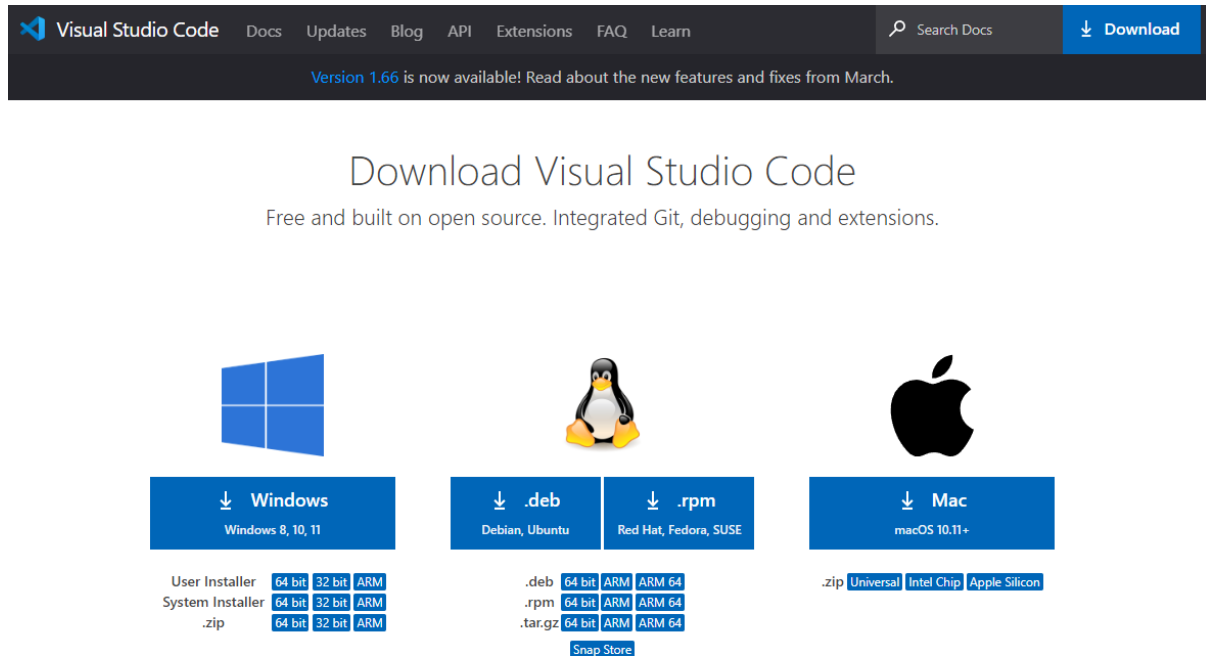


- Kemudian klik **Install Now**

## INSTALASI 2

Install Visual Studio Code pada link berikut: <https://code.visualstudio.com/download>

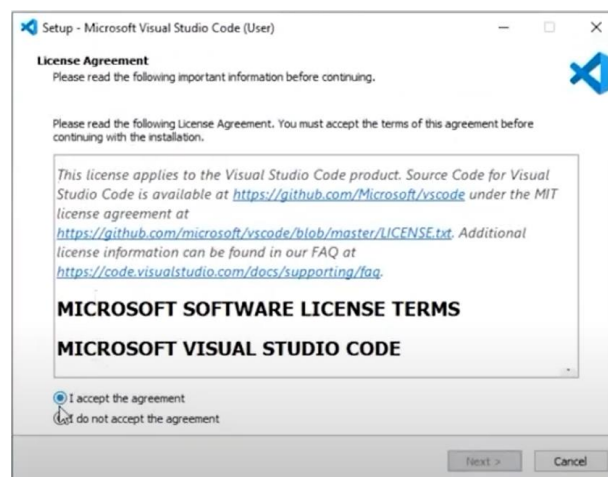
Maka akan muncul pilihan seperti berikut.



Download aplikasi sesuai sistem operasi yang digunakan.

Setelah aplikasi telah terunduh, lanjutkan install aplikasi visual studio code dengan klik 2x setup visual studio code tersebut.

Maka akan muncul tampilan seperti ini:

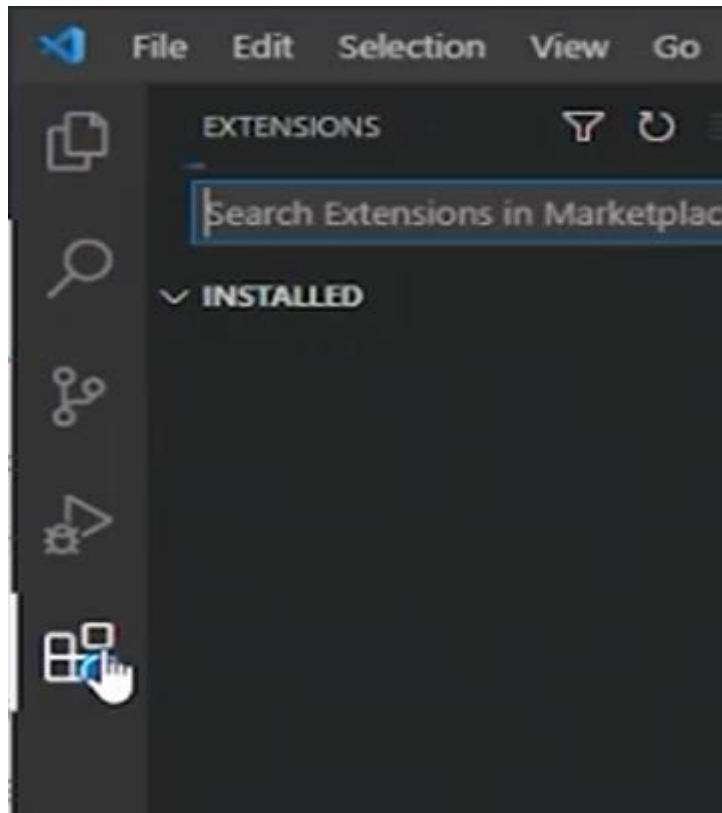


Pilih **"I accept the agreement"**. Selanjutnya klik "next".

Jika muncul tampilan folder untuk penginstallan, klik next untuk default.

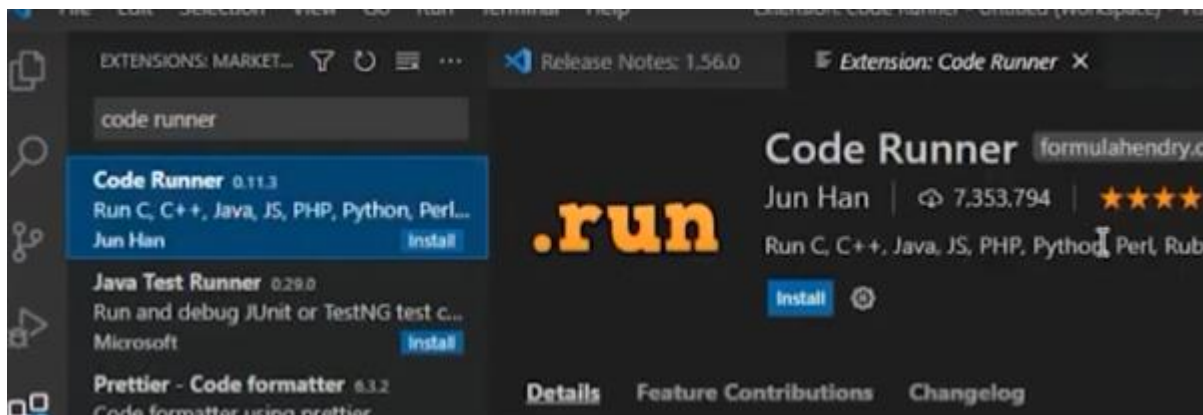
Centang yang diperlukan, klik "next". Jika proses install selesai, maka lanjut ke proses berikut.

Buka aplikasi visual studio code. Proses selanjutnya akan menginstall ekstensi yang diperlukan.



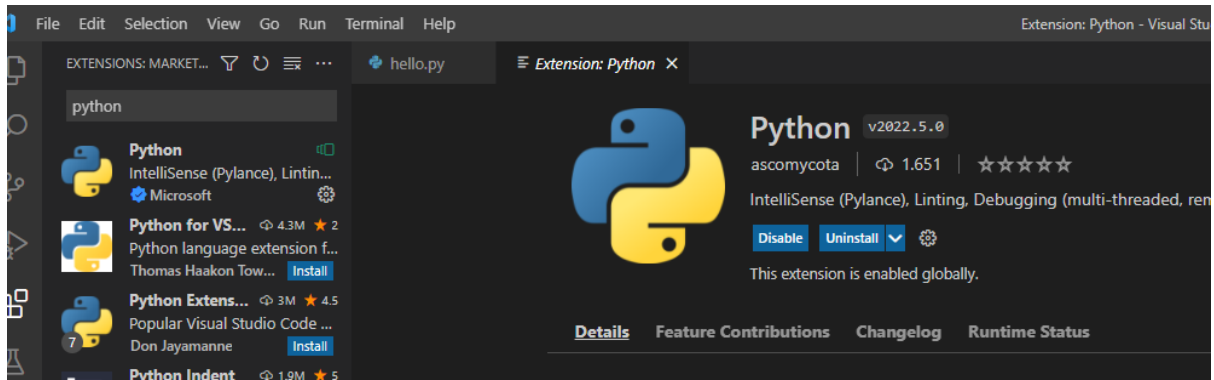
Klik menu seperti pada gambar. Klik kolom search ketikan “**Code Runner**”.

Selanjutnya akan muncul tampilan seperti berikut.



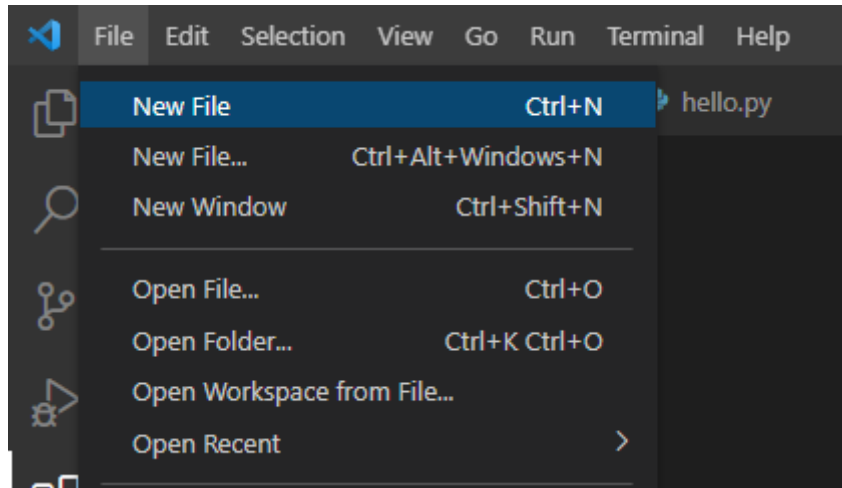
Klik tombol install, lalu tunggu hingga proses selesai.

Kemudian klik kolom search lagi lalu ketikan python. Maka akan tampilan halaman seperti berikut.

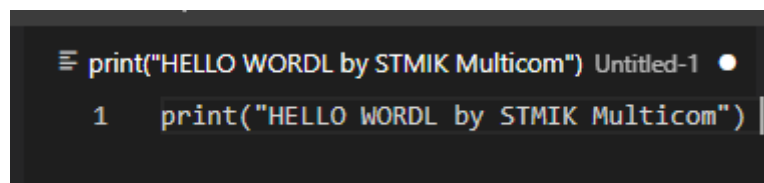


Klik tombol install. Lalu tunggu hingga proses selesai.

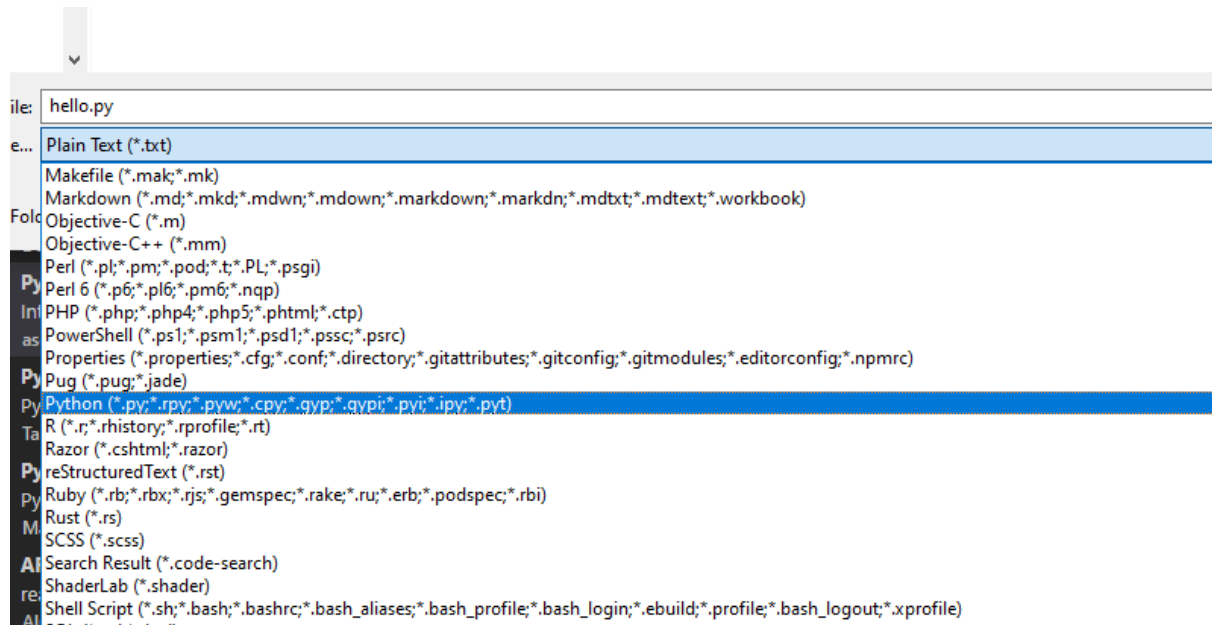
Selanjutnya uji coba buat 1 file python dengan cara klik menu file > new file. Seperti pada tampilan berikut.



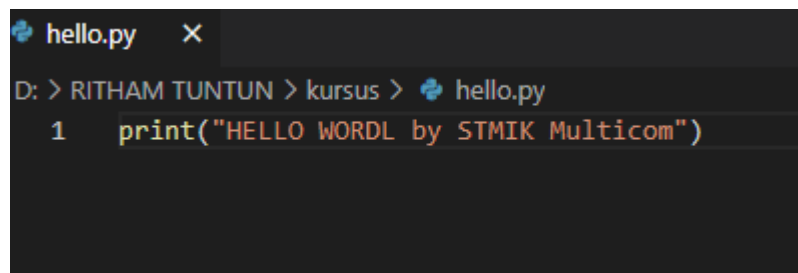
Kemudian ketikkan kode hello world seperti berikut.



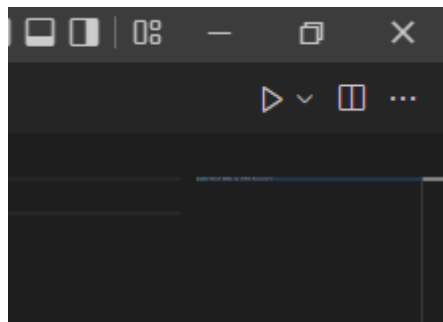
Kemudian save tekan tombol ctrl+s. Simpan dengan nama hello.py. seperti pada tampilan berikut.



Maka tampilan kode akan menjadi seperti berikut.

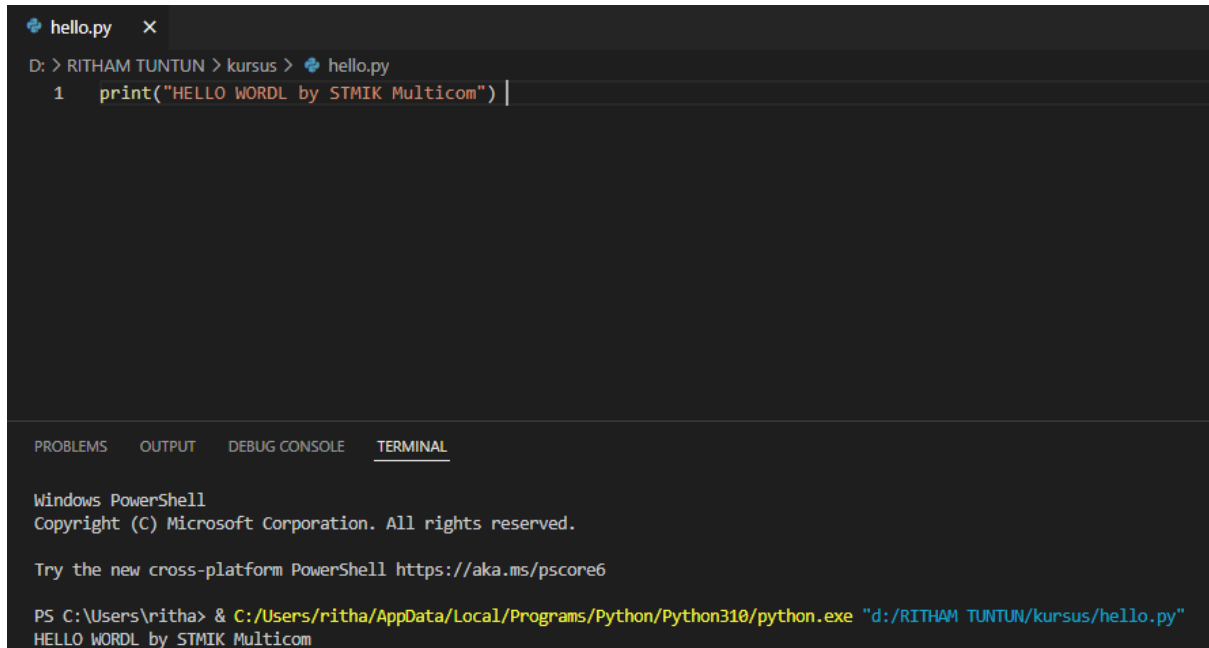


Selanjutnya coba run kode program dengan mengklik menu run (simbol play) yang ada pada kanan atas tampilan aplikasi seperti berikut.



Maka akan muncul hasil seperti berikut.





```
hello.py x
D: > RITHAM TUNTUN > kursus > hello.py
1 print("HELLO WORDL by STMIK Multicom")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\ritha> & C:/Users/ritha/AppData/Local/Programs/Python/Python310/python.exe "d:/RITHAM TUNTUN/kursus/hello.py"
HELLO WORDL by STMIK Multicom
```

Program hello world telah selesai.

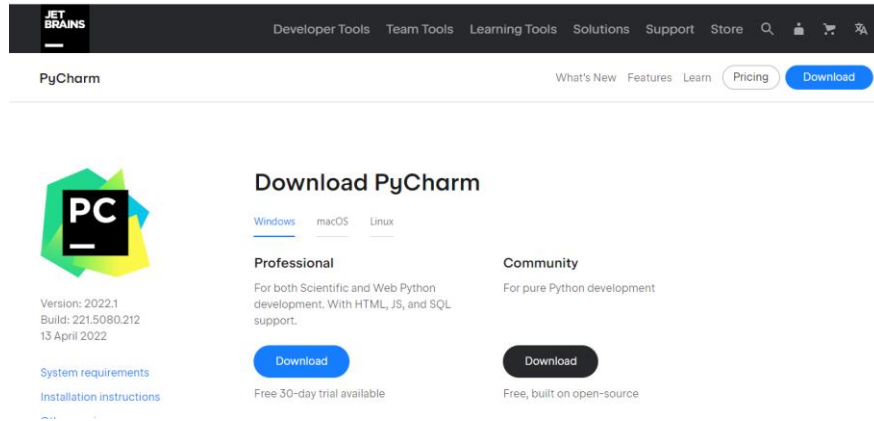
Selain visual studio code, Anda juga bisa menggunakan aplikasi Pycharm. Langkah-langkah untuk download dan instalasi aplikasi pycharm dapat dilihat pada **INSTALASI 3**.

### INSTALASI 3

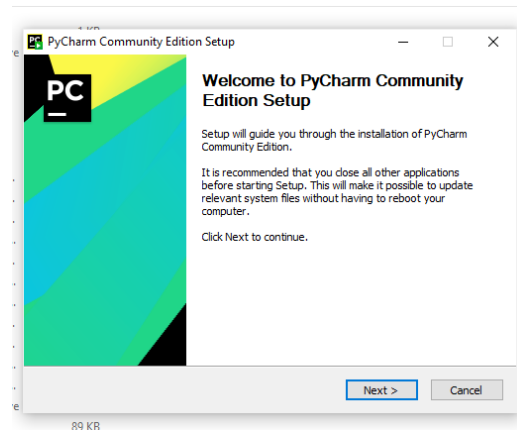
Download aplikasi pycharm pada link berikut:

<https://www.jetbrains.com/pycharm/download/#section=windows>

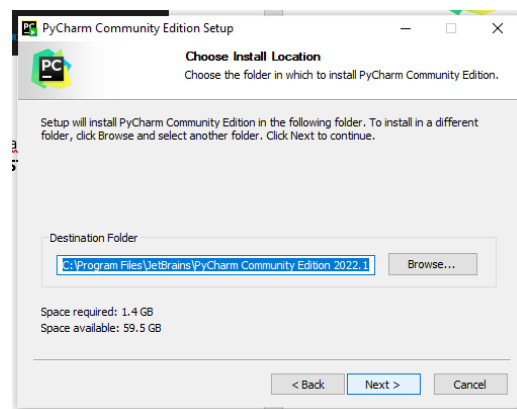
Anda dapat memilih **"Free, Built on open source"**. Seperti pada tampilan berikut.



Tunggu hingga proses download selesai, kemudian klik 2x pada setup aplikasi pycharm.



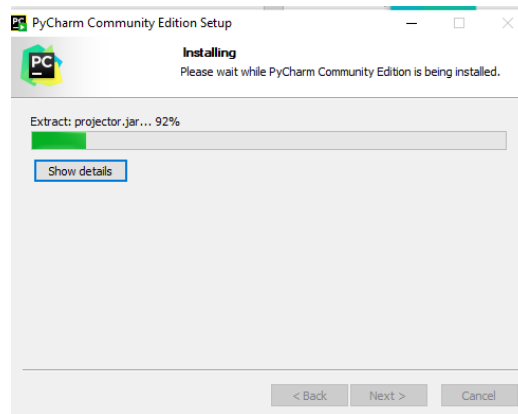
Klik next untuk melanjutkan.



Klik next untuk lokasi install default.

Lanjutkan hingga ke proses install, lalu klik tombol **"install"**.

Tunggu hingga proses selesai.

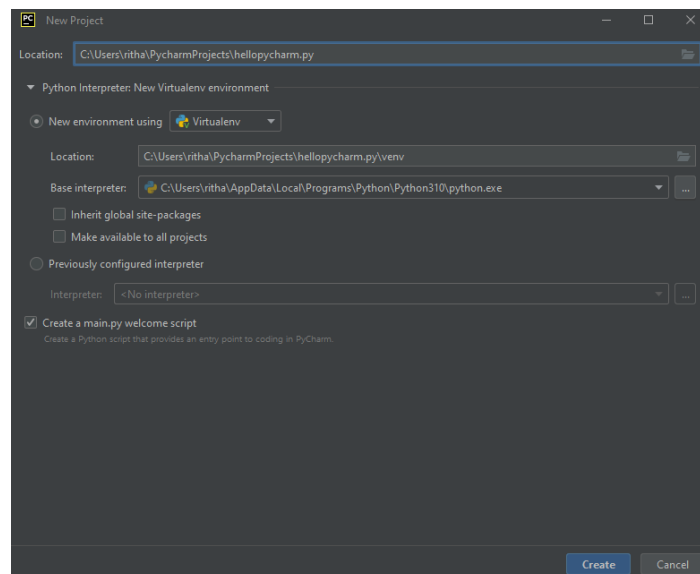


Setelah proses selesai, klik tombol **"Finish"**

Lanjutkan membuat file baru pada pycharm.

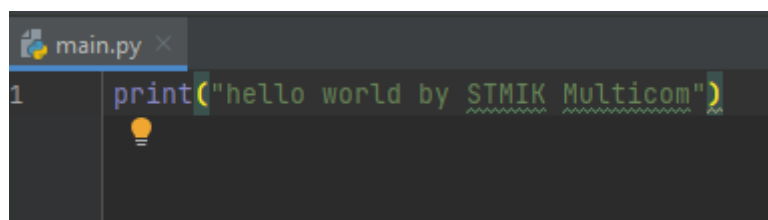
Langkah-langkahnya adalah sebagai berikut.

- Klik menu new project, kemudian beri nama pada project yang akan kita buat seperti pada tampilan berikut.

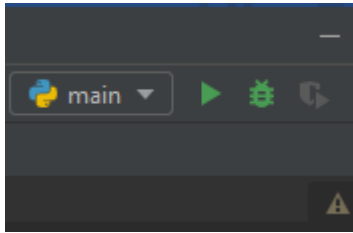


Kemudian klik **"CREATE"**.

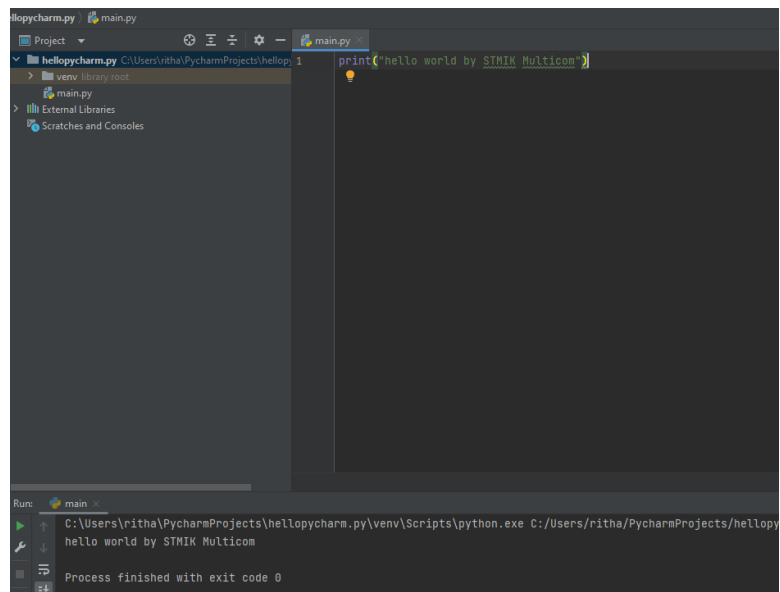
Selanjutnya ketikkan kode program hello world pada aplikasi pycharm seperti berikut.



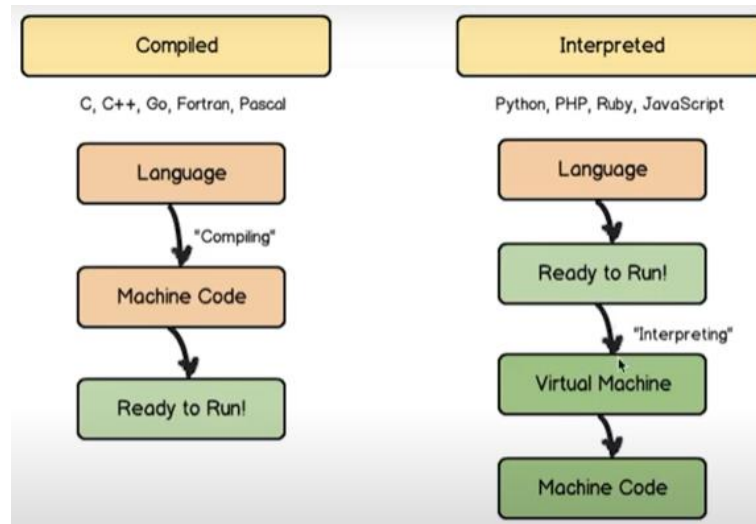
Kemudian untuk menjalankan program. Klik menu run (simbol play) seperti berikut.



Maka hasil yang akan tampil seperti berikut.



Hasil running program akan tampil pada panel bawah aplikasi pycharm.



Kenapa harus python ?

1. Mudah dipelajari. Sintaksnya jelas dan mudah dibaca. Sangat cocok digunakan, terutama sebagai bahasa pemrograman pertama.
2. Simpel tapi powerful. Menulis kodingnya membutuhkan baris perintah yang lebih sedikit dibanding bahasa pemrograman lain.
3. Serbaguna, bisa untuk hampir apa saja kebutuhan atau ketertarikan Anda. Python bisa dipakai untuk pemrograman dekstop maupun mobile, CLI, GUI, web, otomatisasi, hacking, IoT, robotik, dan lain sebagainya.
4. Sangat populer. Rangking 3 di TIOBE index tahun 2020. No. 4 bahasa pemrograman paling banyak digunakan menurut Stackoverflow survey 2020. Merupakan no.1 bahasa pemrograman yang paling cepat perkembangannya dan paling diinginkan beberapa tahun belakangan menurut Stackoverflow.
5. Siapa yang memakai python? Python banyak dipakai perusahaan-perusahaan besar dan top di dunia. Google menggunakannya di mesin pencari, di youtube dan lain-lain, microsoft, dropbox, instagram, pinterest, dan lain-lain.
6. Portable, bisa jalan di multi platform, windows, linux, mac OS, Virtual Machine Java dan .NET.
7. Python adalah bahasa pemrograman yang 'kekinian' karena sedang naik daunnya bidang data science dan cabangnya seperti AI, machine learning, dan big data. Python paling populer dan banyak digunakan karena memiliki library yang lengkap untuk itu seperti sklearn, pytorch, tensorflow, dan lain sebagainya.
8. Open source. Python akan terus berkembang karena didukung oleh komunitas yang besar dan Lembaga Python Software Foundation (PSF) yang tiap tahunnya mengadakan konferensi internasional.
9. Kode python bisa dijalankan secara interaktif (mode interactive) untuk testing, langsung nampak hasilnya.
10. Kode python bisa di-embed ke bahasa lain seperti C dan Java, atau sebaliknya, dari bahasa C atau Java ke Python.

## VARIABEL DAN TIPE DATA Dalam Python.

- Sebuah variable akan menggunakan memory untuk menyimpan suatu nilai.
- Kita dapat membuat sebuah variable dengan menggunakan tanda sama dengan (=).

contoh variabel pada program python.

```
main.py x
1  nama = "Multicom"
2  print('Selamat datang di ', nama)
```

Variabel yang dibuat adalah ' **nama** ' ;

Simbol ' = ' adalah operator.

"**Multicom**" adalah nilai dari variabel ' **nama** '.

## TIPE DATA

Name	Type	Description
Integers	int	Whole numbers, such as: 3 300 200
Floating point	float	Numbers with a decimal point: 2.3 4.6 100.0
Strings	str	Ordered sequence of characters: "hello" 'Sammy' "2000" "楽しい"
Lists	list	Ordered sequence of objects: [10,"hello",200.3]
Dictionaries	dict	Unordered Key:Value pairs: {"mykey": "value", "name": "Frankie"}
Tuples	tup	Ordered immutable sequence of objects: (10,"hello",200.3)
Sets	set	Unordered collection of unique objects: {"a","b"}
Booleans	bool	Logical value indicating True or False

Tipe data harus diperhatikan dalam membuat program karena kekeliruan dalam menentukan tipe data akan membuat operasi dalam suatu program error.

## **BAB I**

# **VARIABEL DAN TIPE DATA**



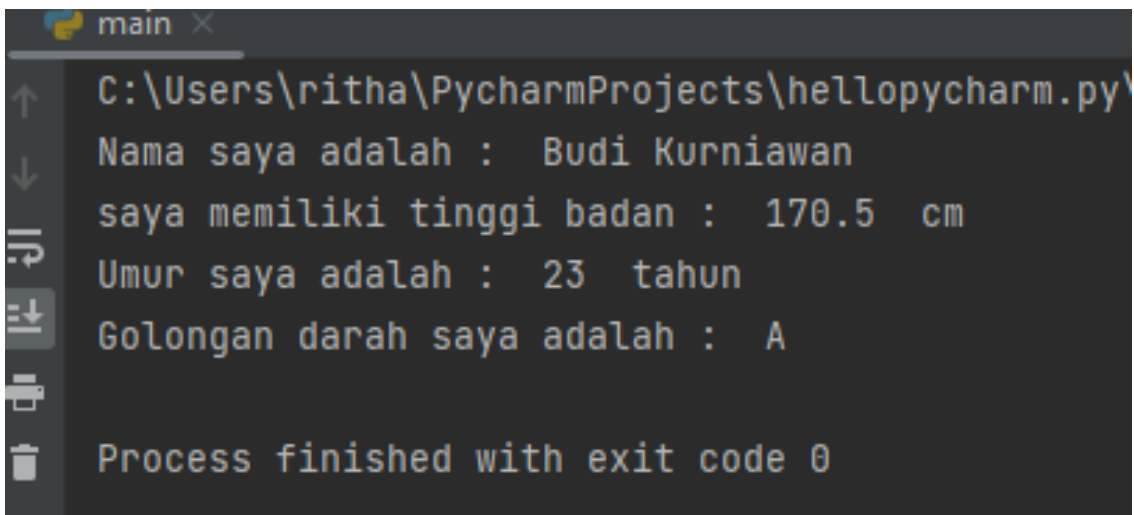
Membuat program yang berisi biodata singkat.

Contoh kode program:

```
nama = "Budi Kurniawan"
tinggi_badan = 170.5
umur = 23
golongan_darah = 'A'

print("Nama saya adalah : ", nama)
print("saya memiliki tinggi badan : ", tinggi_badan, " cm")
print("Umur saya adalah : ", umur, " tahun")
print("Golongan darah saya adalah : ", golongan_darah)
```

Hasil:



```
main x
C:\Users\ritha\PycharmProjects\hellopycharm.py
Nama saya adalah : Budi Kurniawan
saya memiliki tinggi badan : 170.5 cm
Umur saya adalah : 23 tahun
Golongan darah saya adalah : A
Process finished with exit code 0
```

Penjelasan:

Variabel '**nama**' tipe datanya *string*

Variabel '**tinggi\_badan**' tipe datanya *float*

Variabel '**umur**' tipe datanya *integer*

Variabel '**golongan\_darah**' tipe datanya char/string (char = 1 karakter, string = kumpulan karakter)

Penggunaan tanda ( , ) atau koma digunakan untuk memisahkan kalimat string dengan variabel yang akan dipanggil dan nilainya ditampilkan.



Membuat program yang berisi biodata singkat dengan nilai yang diinput oleh user saat menjalankan program.

```
1  nama = input("masukan nama anda : ")
2  tinggi_badan = input("masukan tinggi badan anda : ")
3  umur = input("masukan umur anda : ")
4  golongan_darah = input("masukan golongan darah anda : ")
5
6  print("Nama saya adalah : ",nama)
7  print("saya memiliki tinggi badan : ",tinggi_badan," cm")
8  print("Umur saya adalah : ",umur," tahun")
9  print("Golongan darah saya adalah : ",golongan_darah)
```

Hasil 1:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py
masukan nama anda : Budi Kurniawan
masukan tinggi badan anda : 170.5
masukan umur anda : 23
masukan golongan darah anda : A
Nama saya adalah : Budi Kurniawan
saya memiliki tinggi badan : 170.5 cm
Umur saya adalah : 23 tahun
Golongan darah saya adalah : A

Process finished with exit code 0
```

Hasil 2:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py
masukan nama anda : Eko yulianto
masukan tinggi badan anda : 167.5
masukan umur anda : 30
masukan golongan darah anda : B
Nama saya adalah : Eko yulianto
saya memiliki tinggi badan : 167.5 cm
Umur saya adalah : 30 tahun
Golongan darah saya adalah : B

Process finished with exit code 0
```

Terdapat 2 hasil yaitu hasil 1 dan hasil 2 dengan tampilan output yang berbeda. Dengan menggunakan kode program ini, memungkinkan user bisa memasukkan

(input) nilai yang berbeda pada setiap kali menjalankan program. (ket: karakter yang berwarna hijau pada tampilan hasil adalah inputan atau masukan dari user).

Selanjutnya jika,

Hasil 3 :

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\nmasukn nama anda : 45\nmasukn tinggi badan anda : Budi\nmasukn umur anda : seratus tahun\nmasukn golongan darah anda : 20\nNama saya adalah : 45\nsaya memiliki tinggi badan : Budi cm\nUmur saya adalah : seratus tahun tahun\nGolongan darah saya adalah : 20\n\nProcess finished with exit code 0
```

Hasil seperti pada tampilan hasil 3 terjadi karena kesalahan user dalam memasukan data pada setiap variabel, namun data tetap ditampilkan walau tidak sesuai tipe data karena tipe datanya belum ditentukan. Maka dari itu, dengan menambahkan tipe data pada kode program input, akan mengoreksi tipe data yang dimasukan (input) oleh user.

Seperti pada tampilan berikut:

```
1  nama = str(input("masukn nama anda : "))\n2  tinggi_badan = float(input("masukn tinggi badan anda : "))\n3  umur = int(input("masukn umur anda : "))\n4  golongan_darah = str(input("masukn golongan darah anda : "))\n5\n6  print("Nama saya adalah : ", nama)\n7  print("saya memiliki tinggi badan : ", tinggi_badan, " cm")\n8  print("Umur saya adalah : ", umur, " tahun")\n9  print("Golongan darah saya adalah : ", golongan_darah)
```



Hasil 4 : (jika nilai yang dimasukan tidak sesuai dengan tipe data variabel.

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\venv\Scripts\python.exe C:/Users/ritha/
masukan nama anda : Budi Kurniawan
masukan tinggi badan anda : seratus
Traceback (most recent call last):
  File "C:\Users\ritha\PycharmProjects\hellopycharm.py\input.py", line 2, in <module>
    tinggi_badan = float(input("masukan tinggi badan anda : "))
ValueError: could not convert string to float: 'seratus'
```

Terdapat error karena variabel '**tinggi badan**' harus diisi dengan tipe data float, sedangkan yang dimasukan adalah nilai dengan tipe data string. Maka hasilnya adalah error.

Hasil 5: (jika nilai dari variabel dimasukan dengan tipe data yang sesuai)

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\
masukan nama anda : Budi Kurniawan
masukan tinggi badan anda : 165.5
masukan umur anda : 30
masukan golongan darah anda : A
Nama saya adalah : Budi Kurniawan
saya memiliki tinggi badan : 165.5 cm
Umur saya adalah : 30 tahun
Golongan darah saya adalah : A

Process finished with exit code 0
```

**Tandai Selesai**

## **BAB II**

# **OPERATOR, OPERAND, dan ARITMATIKA**



**Operator, Simbol, Fungsi, dan contoh penggunaannya.**

- a. **Operator** adalah simbol tertentu yang digunakan untuk melakukan operasi aritmatika maupun logika.

Nilai yang padanya dilakukan operasi disebut operand.

Misalnya adalah  $2 + 3$ .

Di sini tanda  $+$  adalah operator penjumlahan.

2 dan 3 adalah operand.

- b. **Operand** merupakan nilai asal yang digunakan didalam proses operasi.

(ket: operand sebelah kiri = a, operand sebelah kanan = b).

operator	penjelasan	contoh
<b>+</b>	penjumlahan, menjumlahkan nilai operand sebelah kiri dengan operand sebelah kanan	$a + b$
<b>-</b>	pengurangan, mengurangi nilai operand sebelah kiri dengan operand sebelah kanan	$a - b$
<b>*</b>	perkalian, mengkalikan nilai operand sebelah kiri dengan operand sebelah kanan	$a * b$
<b>/</b>	pembagian, membagi nilai operand sebelah kiri dengan operand sebelah kanan	$a / b$
<b>%</b>	modulus, sisa pembagian operand sebelah kiri dengan sebelah kanan	$a \% b$
<b>**</b>	pemangkatan, nilai operator sebelah kiri dipangkatkan sesuai operand sebelah kanan	$a ** b$
<b>//</b>	pembagian bulat, operasinya seperti pembagian tetapi angka di belakang koma dihilangkan.	$a // b$



operator	penjelasan	contoh
<b>=</b>	mengisikan nilai di sebelah kanan operator ke sebelah kiri operator	a = 5 b = 10
<b>+=</b>	menjumlahkan operand sebelah kiri dengan sebelah kanan, kemudian hasilnya diisikan ke operand sebelah kiri	a += b
<b>-=</b>	mengurangkan operand sebelah kiri dengan sebelah kanan, kemudian hasilnya diisikan ke operand sebelah kiri	a -= b
<b>*=</b>	mengkalikan operand sebelah kiri dengan sebelah kanan, kemudian hasilnya diisikan ke operand sebelah kiri	a *= b
<b>/=</b>	membagikan operand sebelah kiri dengan sebelah kanan, kemudian hasilnya diisikan ke operand sebelah kiri	a /= b
<b>%=</b>	menjumlahkan operand sebelah kiri dengan sebelah kanan, kemudian hasilnya diisikan ke operand sebelah kiri	a %= b
<b>**=</b>	menjumlahkan operand sebelah kiri dengan sebelah kanan, kemudian hasilnya diisikan ke operand sebelah kiri	a **= b
<b>//=</b>	menjumlahkan operand sebelah kiri dengan sebelah kanan, kemudian hasilnya diisikan ke operand sebelah kiri	a //= b

**Contoh program menggunakan Operator.****Contoh 1:**

```
1  a = 10
2  b = 4
3
4  penjumlahan = a + b
5  pengurangan = a - b
6  perkalian = a * b
7  pembagian = a / b
8  modulus = a % b
9  pemangkatan = a ** b
10
11 print("hasil penjumlahan", penjumlahan)
12 print("hasil pengurangan", pengurangan)
13 print("hasil perkalian", perkalian)
14 print("hasil pembagian", pembagian)
15 print("hasil modulus", modulus)
16 print("hasil pemangkatan", pemangkatan)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py
hasil penjumlahan 14
hasil pengurangan 6
hasil perkalian 40
hasil pembagian 2.5
hasil modulus 2
hasil pemangkatan 10000
```

**Contoh 2:**

```
1  a = int(input("masukan nilai variabel a : "))
2  b = int(input("masukan nilai variabel b : "))
3
4  penjumlahan = a + b
5  pengurangan = a - b
6  perkalian = a * b
7  pembagian = a / b
8  modulus = a % b
9  pemangkatan = a ** b
10
11 print("hasil penjumlahan", penjumlahan)
12 print("hasil pengurangan", pengurangan)
13 print("hasil perkalian", perkalian)
14 print("hasil pembagian", pembagian)
15 print("hasil modulus", modulus)
16 print("hasil pemangkatan", pemangkatan)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\
masukan nilai variabel a : 10
masukan nilai variabel b : 4
hasil penjumlahan 14
hasil pengurangan 6
hasil perkalian 40
hasil pembagian 2.5
hasil modulus 2
hasil pemangkatan 10000
```



**Contoh 3:**

```
1 a = int(input("masukan nilai variabel a : "))
2 b = int(input("masukan nilai variabel b : "))
3
4 print("hasil penjumlahan", a+b)
5 print("hasil pengurangan", a-b)
6 print("hasil perkalian", a*b)
7 print("hasil pembagian", a/b)
8 print("hasil modulus", a%b)
9 print("hasil pemangkatan", a**b)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py
masukan nilai variabel a : 10
masukan nilai variabel b : 4
hasil penjumlahan 14
hasil pengurangan 6
hasil perkalian 40
hasil pembagian 2.5
hasil modulus 2
hasil pemangkatan 10000
```

## Menggunakan Library 'math'.

### Contoh:

```
1  import math
2
3  # Using methods
4  print(math.pow(2, 3))
5  print(math.ceil(3.2))
6  print(math.sqrt(25))
7
8  # Using constants
9  print(math.pi)
```

### Hasil:

```
8.0
4
5.0
3.141592653589793
```

## Daftar Rumus yang disediakan library 'math'.

### Konstanta Matematika

Konstanta	Deskripsi
<code>math.e</code>	Mengembalikan bilangan Euler (2,7182 ...)
<code>math.inf</code>	Mengembalikan tak terhingga positif titik mengambang
<code>math.nan</code>	Mengembalikan nilai NaN titik mengambang (Bukan Angka)
<code>math.pi</code>	Mengembalikan PI (3.1415 ...)
<code>math.tau</code>	Mengembalikan tau (6.2831 ...)

### Method Matematika

Method	Deskripsi
<code>math.acos()</code>	Menampilkan kosinus busur sebuah angka
<code>math.acosh()</code>	Mengembalikan kosinus hiperbolik terbalik sebuah angka
<code>math.asin()</code>	Menampilkan sinus busur sebuah angka
<code>math.asinh()</code>	Mengembalikan sinus hiperbolik terbalik sebuah angka
<code>math.atan()</code>	Mengembalikan tangen busur sebuah angka dalam radian
<code>math.atan2()</code>	Mengembalikan tangen busur dari y / x dalam radian
<code>math.atanh()</code>	Mengembalikan tangen hiperbolik terbalik sebuah angka
<code>math.ceil()</code>	Membulatkan angka ke atas hingga bilangan bulat terdekat
<code>math.comb()</code>	Mengembalikan jumlah cara memilih k item dari n item tanpa pengulangan dan urutan
<code>math.copysign()</code>	Mengembalikan float yang terdiri dari nilai parameter pertama dan tanda parameter kedua



<code>math.cos()</code>	Mengembalikan kosinus sebuah angka
<code>math.cosh()</code>	Menampilkan kosinus hiperbolik sebuah angka
<code>math.degrees()</code>	Mengonversi sudut dari radian ke derajat
<code>math.dist()</code>	Mengembalikan jarak Euclidean antara dua titik (p dan q), di mana p dan q adalah koordinat titik itu
<code>math.erf()</code>	Mengembalikan fungsi kesalahan dari sebuah angka
<code>math.erfc()</code>	Mengembalikan fungsi kesalahan komplementer dari sebuah angka
<code>math.exp()</code>	Mengembalikan E yang dipangkatkan dari x
<code>math.expm1()</code>	Mengembalikan $e^x - 1$
<code>math.fabs()</code>	Mengembalikan nilai absolut sebuah angka
<code>math.factorial()</code>	Mengembalikan faktorial sebuah angka
<code>math.floor()</code>	Membulatkan angka ke bawah hingga bilangan bulat terdekat
<code>math.fmod()</code>	Mengembalikan sisa dari $x / y$
<code>math.frexp()</code>	Mengembalikan mantissa dan eksponen, dari bilangan tertentu
<code>math.fsum()</code>	Mengembalikan jumlah dari semua item dalam iterable (tupel, array, daftar, dll.)
<code>math.gamma()</code>	Mengembalikan fungsi gamma pada x
<code>math.gcd()</code>	Menampilkan pembagi persekutuan terbesar dari dua bilangan bulat
<code>math.hypot()</code>	Mengembalikan norma Euclidean
<code>math.isclose()</code>	Memeriksa apakah dua nilai dekat satu sama lain, atau tidak
<code>math.isfinite()</code>	Memeriksa apakah suatu angka terbatas atau tidak
<code>math.isinf()</code>	Memeriksa apakah suatu bilangan tidak terbatas atau tidak



<code>math.isnan()</code>	Memeriksa apakah suatu nilai adalah NaN (bukan angka) atau bukan
<code>math.isqrt()</code>	Membulatkan angka akar kuadrat ke bawah ke bilangan bulat terdekat
<code>math.ldexp()</code>	Mengembalikan inversi dari <code>math.frexp()</code> yang merupakan $x * (2 ** i)$ dari bilangan yang diberikan $x$ dan $i$
<code>math.lgamma()</code>	Mengembalikan nilai <a href="#">log</a> gamma dari $x$
<code>math.log()</code>	Menampilkan logaritma natural sebuah bilangan, atau logaritma bilangan ke basis
<code>math.log10()</code>	Mengembalikan logaritma basis 10 dari $x$
<code>math.log1p()</code>	Mengembalikan logaritma natural dari $1 + x$
<code>math.log2()</code>	Mengembalikan logaritma basis 2 dari $x$
<code>math.perm()</code>	Mengembalikan jumlah cara memilih $k$ item dari $n$ item dengan urutan dan tanpa pengulangan
<code>math.pow()</code>	Mengembalikan nilai $x$ ke pangkat $y$
<code>math.prod()</code>	Mengembalikan produk dari semua elemen dalam iterable
<code>math.radians()</code>	Mengonversi nilai derajat menjadi radian
<code>math.remainder()</code>	Mengembalikan nilai terdekat yang bisa membuat pembilang habis habisnya oleh penyebut
<code>math.sin()</code>	Menampilkan sinus sebuah angka
<code>math.sinh()</code>	Menampilkan sinus hiperbolik sebuah angka
<code>math.sqrt()</code>	Menampilkan akar kuadrat dari sebuah angka
<code>math.tan()</code>	Menampilkan tangen sebuah angka
<code>math.tanh()</code>	Menampilkan tangen hiperbolik sebuah angka
<code>math.trunc()</code>	Mengembalikan bagian bilangan bulat terpotong dari sebuah angka

**LATIHAN SOAL. Membuat Program:****1. Operasi dengan rumus :**

Nama	Rumus
Volume (V)	$V = s \times s \times s$
	$V = s^3$
Luas permukaan (L)	$L = 6 \times s \times s$
	$L = 6 \times s^2$

**2. Operasi dengan rumus :**

Nama	Rumus
Volume (V)	$V = p \times l \times t$
Luas Permukaan (L)	$L = 2 \times (p.l + p.t + l.t)$

**Penyelesaian:****1. Penyelesaian soal 1**

```
1 s = int(input("masukan nilai sisi : "))
2
3 V_cth1 = s*s*s
4 V_cth2 = s**3
5
6 print("hasil rumus pertama : ", V_cth1)
7 print("hasil rumus kedua : ", V_cth2)
```

**2. Penyelesaian soal 2**

```
1 p = int(input("masukan nilai panjang : "))
2 l = int(input("masukan nilai lebar : "))
3 t = int(input("masukan nilai tinggi : "))
4
5 V = p * l * t
6 L = 2 * ((p*l)+(p*t)+(l*t))
7
8 print("hasil Volume : ", V)
9 print("hasil Luas Permukaan : ", L)
```

**Tandai Selesai**☐

## **BAB III**

# **STRINGS DALAM PYTHON**



## STRING

String dideklarasikan dengan cara menggunakan tanda petik satu atau tanda petik dua. Contoh:

```
1
2  nama = "Budi Kurniawan"
3  alamat = 'Kotamobagu'
```

Untuk menampilkan nilainya bisa menggunakan kedua cara berikut:

```
1
2  nama = "Budi Kurniawan"
3  alamat = 'Kotamobagu'
4
5  print("nama saya adalah ", nama, " saya tinggal di ", alamat)
6
7  print("nama saya adalah " + nama + " saya tinggal di " + alamat)
```

Hasilnya adalah seperti berikut:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\venv\Scripts\python.exe
nama saya adalah Budi Kurniawan saya tinggal di Kotamobagu
nama saya adalah Budi Kurniawan saya tinggal di Kotamobagu
```



## ESCAPE CHARACTER

- Escape character dapat dibuat dengan tanda ' \ '
- Fungsinya adalah untuk menyisipkan karakter ilegal ke dalam sebuah string. Misalnya menampilkan tanda kutip pada hasil output atau keluaran program.

Contoh:

```
1 print("Semoga kursus anda lancar")
2 print("\" Semoga kursus anda lancar \"")
```

Maka dapat dilihat perbedaan outputnya pada tampilan berikut ini:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py
Semoga kursus anda lancar
" Semoga kursus anda lancar "
```

## STRING METHODS

String methods adalah methods yang terdapat pada Python yang digunakan untuk mengubah gaya atau tampilan dari suatu string seperti contoh berikut.

```
1 teks = "stmik multicom bolaang mongondow"
2
3 print(teks.upper())
4 print(teks.lower())
5 print(len(teks))
6 print(teks.split(' '))
7 print(teks.capitalize())
8 print(teks.index('m'))
```

Hasilnya adalah sebagai berikut.

```
C:\Users\ritha\PycharmProjects\hellopycharm.py
STMIK MULTICOM BOLAANG MONGONDOW
stmik multicom bolaang mongondow
32
['stmik', 'multicom', 'bolaang', 'mongondow']
Stmik multicom bolaang mongondow
2
```

Penjelasan:

<code>print(teks.upper())</code>	Mengubah teks menjadi huruf kapital
<code>print(teks.lower())</code>	Mengubah teks menjadi huruf kecil
<code>print(len(teks))</code>	Menghitung jumlah karakter pada string
<code>print(teks.split(' '))</code>	Membagi teks berdasarkan karakter. Cth: spasi ' '
<code>print(teks.capitalize())</code>	Mengubah teks awal pada teks menjadi huruf kapital
<code>print(teks.index('m'))</code>	Mendapatkan indeks dari teks

## SLICING STRINGS

Berfungsi untuk mengembalikan karakter yang berada dalam rentang indeks yang diberikan.

P	Y	T	H	O	N
0	1	2	3	4	5
-6	-5	-4	-3	-2	-1

Cat: indeks selalu mulai dari 0 (nol)

Contoh:

P	Y	T	H	O	N
0	1	2	3	4	5
-6	-5	-4	-3	-2	-1

<pre>print(text[0]) print(text[2:]) print(text[:4]) print(text[3:5]) print(text[:]) print(text[-2]) print(text[-4:-1])</pre>	<pre>P thon Pyth ho Python o tho</pre>
--	--

## IN dan NOT IN

Memeriksa teks pada suatu string. Contoh :

```
1  teks = "stmik multicom"
2
3  print("stmik" in teks)
4  print("komputer" in teks)
5  print("stmik" not in teks)
6  print("komputer" not in teks)
```

Hasil:

```
C:\Users\rit
True
False
False
True
```

Penjelasan:

Hasil pertama adalah **True** karena "stmik" ada di dalam string teks

Hasil kedua adalah **False** karena "komputer" tidak ada di dalam string teks

Hasil ketiga adalah **False** karena "stmik" ada di dalam string teks

Hasil keempat adalah **True** karena "komputer" tidak ada di dalam string teks

## CONDITIONAL STATEMENTS

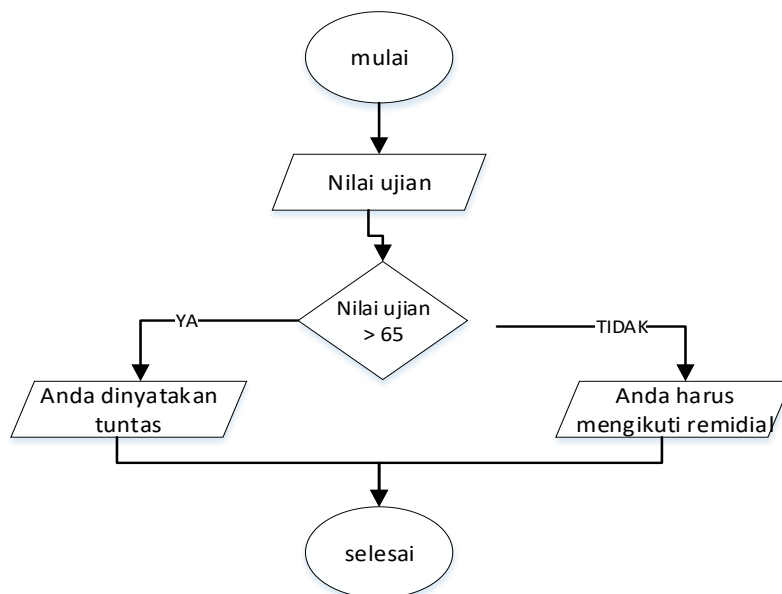
Conditional statement Python adalah aliran yang mengontrol kode program Python berdasarkan pengujian pernyataan bersyarat.

operator	penjelasan	contoh
>	lebih besar, nilai operand sebelah kiri lebih besar dari sebelah kanan	<code>a &gt; b</code>
<	lebih kecil, nilai operand sebelah kiri lebih kecil dari sebelah kanan	<code>a &lt; b</code>
==	sama dengan, membandingkan nilai operand sebelah kiri dengan operand sebelah kanan. Jika nilai operand sebelah kiri sama dengan operand sebelah kanan maka hasilnya True	<code>a == b</code>
!=	Tidak sama dengan, membandingkan nilai operand sebelah kiri dengan operand sebelah kanan. Jika nilai operand sebelah kiri tidak sama dengan nilai operand sebelah kanan maka hasilnya True	<code>a != b</code>
>=	Lebih besar atau sama dengan, membandingkan jika nilai operand sebelah kiri lebih besar atau sama dengan nilai operand sebelah kanan maka hasilnya True	<code>a &gt;= b</code>
<=	Lebih kecil atau sama dengan, membandingkan jika nilai operand sebelah kiri lebih kecil atau sama dengan nilai operand sebelah kanan maka hasilnya True	<code>a &lt;= b</code>

operator	penjelasan	contoh
<b>and</b>	hasilnya adalah True jika kedua operand bernilai True	<pre> a = True b = False print(a and b) print(b and a) print(a and a) print(b and b) </pre>
<b>or</b>	hasilnya adalah True jika salah satu atau kedua nilai operand adalah True	<pre> a = True b = False print(a or b) print(b or a) print(a or a) print(b or b) </pre>
<b>not</b>	membalikan nilai operand yang sebelumnya. Jika nilai operand adalah True maka hasilnya adalah False	<pre> a = True b = False print(not a) print(not b) </pre>

operator	penjelasan	contoh
<b>is</b>	Hasilnya adalah True jika nilai operand sebelah kiri dan sebelah kanan adalah sama	<pre> a = 5 b = 10 print(a is b) print(b is a) print(a is a) print(b is b) </pre>
<b>is not</b>	Hasilnya adalah True jika nilai operand sebelah kiri tidak sama dengan nilai operand sebelah kanan	<pre> a = 5 b = 10 print(a is not b) print(b is not a) print(a is not a) print(b is not b) </pre>

Kondisi atau percabangan adalah suatu pemilihan statemen / pernyataan yang akan dieksekusi dimana pemilihan tersebut didasarkan pada kondisi tertentu. Percabangan digunakan jika ada masalah yang tidak dapat dipecahkan dengan perhitungan yang pasti, inti dari pemecahan masalah jenis ini adalah menganalisis semua kondisi yang mungkin terjadi dan menentukan statmen yang dilakukan untuk setiap kondisi tersebut.



Pengambilan keputusan digunakan untuk mengantisipasi kondisi yang terjadi saat jalanya program dan menentukan tindakan apa yang akan diambil sesuai dengan kondisi. Pada python ada beberapa statement/kondisi yaitu:

1. **if**,
2. **if else**
3. **if elif**

### Contoh 1. Menggunakan 'IF'

Misalnya dalam suatu kondisi jika nilai dari mahasiswa melebihi standar kelulusan (sesuai kondisi), maka akan tampil teks "selamat".

Contoh program :

```
1  nilai = int(input("masukan nilai mahasiswa : "))
2
3  if(nilai>=75):
4      print("SELAMAT !")
```

Hasil 1 (jika sesuai kondisi):

```
C:\Users\ritha\PycharmProjects\hel
masukan nilai mahasiswa : 78
SELAMAT !

Process finished with exit code 0
```

Hasil 2 (jika tidak sesuai kondisi):

```
C:\Users\ritha\PycharmProjects\hel
masukan nilai mahasiswa : 60

Process finished with exit code 0
```

Penjelasan:

Kalimat atau string "SELAMAT !" akan tampil jika nilai yang dimasukan lebih dari 75. (ket: 75 adalah nilai yang ditentukan pada kondisi tapi tidak konstan)

Catatan: Penulisan kode program untuk output sesuai kondisi (print) harus masuk 1 blok ke kanan. Bisa tambahkan tab.



## Contoh 2. Menggunakan 'IF ELSE'

Misalnya dalam suatu kondisi jika nilai dari mahasiswa melebihi standar kelulusan (sesuai kondisi), maka akan tampil teks "Lulus", jika nilai dari mahasiswa kurang dari standar (tidak sesuai kondisi) maka akan tampil teks alternatif "Tidak Lulus".

Contoh program:

```
1  nilai = int(input("masukan nilai mahasiswa : "))
2
3  if(nilai>=75):
4      print("LULUS")
5  else:
6      print("TIDAK LULUS")
```

Hasil 1 (jika sesuai kondisi):

```
C:\Users\ritha\PycharmProjects\hell
masukan nilai mahasiswa : 78
LULUS

Process finished with exit code 0
```

Hasil 2 (jika tidak sesuai kondisi):

```
C:\Users\ritha\PycharmProjects\hell
masukan nilai mahasiswa : 65
TIDAK LULUS

Process finished with exit code 0
```

### Contoh 3. Menggunakan 'IF ELIF'.

Misalnya pada kasus penentuan nilai huruf berdasarkan nilai angka yang dicapai oleh mahasiswa pada suatu mata kuliah. Jika nilai angka yang dicapai 0 sampai 50 maka nilai hurufnya adalah 'D'. Jika nilai angka yang dicapai 51 sampai 80 maka nilai hurufnya adalah 'C'. Jika nilai angka yang dicapai 81 sampai 90 maka nilai hurufnya adalah 'B'. Jika nilai angka yang dicapai 91 sampai 100 maka nilai hurufnya adalah 'A'.

Contoh program:

```
1  nilai = int(input("masukan nilai mahasiswa : "))
2
3  if(nilai<=50):
4      print("D")
5  elif(nilai<=80):
6      print("C")
7  elif(nilai<=90):
8      print("B")
9  elif(nilai<=100):
10     print("A")
```

Hasil 1:

```
C:\Users\ritha\PycharmProjects\hel
masukan nilai mahasiswa : 87
B
Process finished with exit code 0
```

Hasil 2:

```
C:\Users\ritha\PycharmProjects\hel
masukan nilai mahasiswa : 56
C
Process finished with exit code 0
```

Hasil 3:

```
C:\Users\ritha\PycharmProjects\hel
masukan nilai mahasiswa : 96
A
Process finished with exit code 0
```

**LATIHAN SOAL. Membuat Program:**

1. Menampilkan harga makanan berdasarkan masukan nomor menu.
2. Permainan perbandingan poin.

**Penyelesaian:**

1. Menampilkan harga makanan berdasarkan masukan nomor menu.

```
1  nomor = int(input("masukan nomor menu : "))
2
3  if(nomor==1):
4      print("Nama Menu : Nasi Goreng")
5      print("Harga Menu : Rp. 15.000 ")
6  elif(nomor==2):
7      print("Nama Menu : Nasi Kuning")
8      print("Harga Menu : Rp. 10.000 ")
9  elif(nomor==3):
10     print("Nama Menu : Nasi campur")
11     print("Harga Menu : Rp. 20.000 ")
12 elif(nomor==4):
13     print("Nama Menu : Nasi putih")
14     print("Harga Menu : Rp. 5.000 ")
```

Untuk lebih kompleks lagi, kita tambahkan perhitungan ke dalam tampilan menunya. Seperti pada contoh berikut.

```
1  nomor = int(input("masukan nomor menu : "))
2
3  if(nomor==1):
4      print("Nama Menu : Nasi Goreng")
5      print("Harga Menu : Rp. 15.000 ")
6      harga=15000
7      jumlah = int(input("masukan jumlah porsi : "))
8      total = harga * jumlah
9      print("Total yang harus dibayar adalah : ",total)
10     uang = int(input("masukan jumlah pembayaran : "))
11     kembalian = uang - total
12     print("Jumlah kembaliannya adalah : ",kembalian)
13 elif(nomor==2):
14     print("Nama Menu : Nasi Kuning")
15     print("Harga Menu : Rp. 10.000 ")
16     harga=10000
17     jumlah = int(input("masukan jumlah porsi : "))
18     total = harga * jumlah
19     print("Total yang harus dibayar adalah : ",total)
20     uang = int(input("masukan jumlah pembayaran : "))
21     kembalian = uang - total
22     print("Jumlah kembaliannya adalah : ",kembalian)
23 elif(nomor==3):
24     print("Nama Menu : Nasi campur")
25     print("Harga Menu : Rp. 20.000 ")
26     harga=20000
27     jumlah = int(input("masukan jumlah porsi : "))
28     total = harga * jumlah
29     print("Total yang harus dibayar adalah : ",total)
30     uang = int(input("masukan jumlah pembayaran : "))
31     kembalian = uang - total
32     print("Jumlah kembaliannya adalah : ",kembalian)
33 elif(nomor==4):
34     print("Nama Menu : Nasi putih")
35     print("Harga Menu : Rp. 5.000 ")
36     harga=5000
37     jumlah = int(input("masukan jumlah porsi : "))
38     total = harga * jumlah
39     print("Total yang harus dibayar adalah : ",total)
40     uang = int(input("masukan jumlah pembayaran : "))
41     kembalian = uang - total
42     print("Jumlah kembaliannya adalah : ",kembalian)
```

## 2. Permainan perbandingan poin

```
1 player = int(input("masukan poin player : "))
2 musuh1 = int(input("masukan poin musuh : "))
3 if(player>musuh1):
4     print("player menang, lanjut ke level 2")
5     player += musuh1
6     print("poin player sekarang", player)
7     musuh2 = int(input("masukan poin musuh : "))
8     if(player>musuh2):
9         print("player menang, lanjut ke level 3")
10        player += musuh2
11        print("poin player sekarang", player)
12        musuh3 = int(input("masukan poin musuh : "))
13        if (player > musuh3):
14            print("player menang")
15            player += musuh3
16            print("poin player sekarang", player)
17            print("petualangan selesai")
18        else:
19            print("poin player kalah")
20            print("game over")
21    else:
22        print("poin player kalah")
23        print("game over")
24 else:
25     print("poin player kalah")
26     print("game over")
```

<b>Tandai Selesai</b>	
-----------------------	--

## **BAB IV**

# **LOOP STATEMENTS**



Secara umum, pernyataan pada bahasa pemrograman akan dieksekusi secara berurutan. Pernyataan pertama dalam sebuah fungsi dijalankan pertama, diikuti oleh yang kedua, dan seterusnya. Tetapi akan ada situasi dimana Anda harus menulis banyak kode, dimana kode tersebut sangat banyak. Jika dilakukan secara manual maka Anda hanya akan membuang-buang tenaga dengan menulis beratus-ratus bahkan beribu-ribu kode. Untuk itu Anda perlu menggunakan pengulangan di dalam bahasa pemrograman Python.

Kenapa harus Loop ?

Bandingkan kedua program di bawah ini.

```
print("Looping!")  
print("Looping!")  
print("Looping!")  
print("Looping!")  
print("Looping!")  
print("Looping!")
```

```
for i in range(6):  
    print("Looping!")
```

Kedua program tersebut memiliki tujuan yang sama yaitu menampilkan output berupa kata ' **Looping!** '. tapi program pertama memakan lebih banyak kode program dibandingkan program kedua. Maka alasan adanya looping statement untuk mengatasi masalah seperti pada gambar tersebut.

## RANGE

Range memiliki parameter yaitu : Start, stop, step.

Secara default parameter range adalah: range(start=0, stop, step=1)

```
for i in range(6):  
    print("Looping!")
```

Start = 0, stop = 6, step 1.

Contoh 1:

```
1   for i in range(0, 6, 1):  
2       print("Looping!")
```

Hasil 1:

```
C:\Users\ritha\PycharmProjects  
Looping  
Looping  
Looping  
Looping  
Looping  
Looping
```

Contoh 2:

```
1   for i in range(0, 6, 2):  
2       print("Looping!")
```

Hasil 2:

```
C:\Users\ritha\PycharmProjects  
Looping!  
Looping!  
Looping!
```



Contoh 3:

```
for i in range(0, 6, 1):  
    print(i)
```

hasil :

```
C:\Users\ritha\  
0  
1  
2  
3  
4  
5
```

Contoh 4:

```
for i in range(0, 6, 2):  
    print(i)
```

Hasil:

```
C:\Users\riti  
0  
2  
4
```

Contoh 5:

```
for i in range(5, 30, 5):  
    print(i)
```

Hasil:

```
C:\Users\ritha\  
5  
10  
15  
20  
25
```

## BREAK AND CONTINUE

- a. Dengan break statement, kita dapat menghentikan sebuah loop walaupun loop sedang berjalan.

Contoh:

```
for i in range(10, 20, 3):  
    if i == 19:  
        break  
  
    print(i)
```

Hasil:

```
10  
13  
16
```

- b. Dengan continue statement, kita dapat melewati operasi yang kita lakukan dan langsung lanjut ke iterasi berikutnya.

Contoh:

```
for i in range(10, 20, 3):  
    if i == 13:  
        continue  
  
    print(i)
```

Hasil:

```
10  
16  
19
```

Misalnya pada kasus menampilkan bilangan ganjil.

```
1 for i in range(1, 10, 1):  
2     if i % 2 == 0:  
3         continue  
4     print(i)
```

Hasil:

```
C:\Users\ritha\Pycharm  
1  
3  
5  
7  
9
```

Penjelasan:

Start pada range yang dibuat adalah dari 1, Lalu akan stop (berhenti) di 10, step yang ditentukan adalah 1. Seharusnya hasil output adalah berupa menampilkan angka 1, 2, 3, sampai 9. Tapi karena terdapat kondisi **'if'** yang menyatakan jika 'i' modulus 2 == 0 (nilai dari i habis dibagi 2), maka **'continue'** atau angka tersebut akan dilewati.

Cat : angka genap selalu habis jika dibagi dengan 2 (tidak ada sisa atau modulus).

## WHILE STATEMENT

While statement akan mengeksekusi operasi di dalamnya selama kondisinya bernilai **'True'**.

Contoh:

```
1      nilai = 50
2
3      while(nilai < 70):
4          print("nilai sekarang", nilai)
5          nilai += 5
6
7      print("sudah batas maksimal")
```

Hasil:

```
C:\Users\ritha\PycharmProjects\I
nilai sekarang 50
nilai sekarang 55
nilai sekarang 60
nilai sekarang 65
sudah batas maksimal
```

Penjelasan:

Selama masih dalam kondisi (nilai kurang dari 70), perintah untuk menampilkan nilai tetap akan dijalankan.

Setelah kondisi selesai, maka perintah yang akan dieksekusi adalah perintah selanjutnya (menampilkan string "sudah batas maksimal").

**SOAL Latihan Membuat program:****1. Menampilkan output jumlah putaran seperti berikut:**

Putaran ke-1  
Putaran ke-2  
Putaran ke-3  
Putaran ke-4  
Putaran ke-5  
Putaran selesai!

**Penyelesaian:**

```
putaran = int(input("masukan jumlah putaran : "))
putaran += 1
for i in range(1, putaran):
    print("putaran ke-", i)
print("putanan selesai!")
```

**2. Permainan perbandingan poin menggunakan 'while'.**

```
player = int(input("masukan poin player :"))
musuh = int(input("masukan poin musuh :"))
while player > musuh:
    print("player menang")
    player += musuh
    print("poin player sekarang", player)
    musuh = int(input("masukan poin musuh "))
print("=====")
print("Game Over")
```

<b>Tandai Selesai</b>	
-----------------------	--

## **BAB V**

# **PYTHON COLLECTION DATA TYPES**



## LIST

List adalah sebuah koleksi yang terurut dan dapat diganti. List bisa menampung elemen yang duplikat. Tiap nilai dalam list memiliki indeks yang dimulai dari 0 (nol).

Contoh:

```
listExample = ['Python', 42, 3.9, True, 42]
print(listExample[0])
print(listExample[1:4])
print(listExample[-1])
print(listExample)
```

Hasil:

```
Python
[42, 3.9, True]
42
['Python', 42, 3.9, True, 42]
```

Penjelasan:

- Tampil pada baris pertama adalah nilai list dengan indeks 0 (nol)
- Tampil pada baris kedua adalah nilai list dengan indeks 1 sampai 3.
- Tampil pada baris ketiganya adalah nilai list dengan indeks -1. (dihitung mulai dari kanan)
- Tampil pada baris keempat adalah seluruh nilai dalam list.

## LIST OPERATION (CREATE, READ, UPDATE, DELETE)

### 1. Create (insert / append)

Misalnya terdapat list dengan nilai nama-nama hari seperti senin, Selasa, Rabu, Kamis, Sabtu, Minggu.

Lalu kita ingin menambahkan nilai pada list tersebut dengan nilai nama hari 'Jumat' dengan posisi berada di indeks 4. Maka kode programnya adalah sebagai berikut:

```
1 list = ['senin', 'Selasa', 'Rabu', 'Kamis', 'Sabtu', 'Minggu']
2 list.insert(4, 'Jumat')
3
4 print(list)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\venv\Scripts\python
['senin', 'Selasa', 'Rabu', 'Kamis', 'Jumat', 'Sabtu', 'Minggu']

Process finished with exit code 0
```

Dapat kita lihat perintah **insert** nilai dari list bertambah di index 4 dengan nilai 'Jumat'.

Cotoh 2:

```
1 list = ['senin', 'Selasa', 'Rabu', 'Kamis', 'Sabtu', 'Minggu']
2 list.append('Jumat')
3
4 print(list)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\venv\Scripts\python
['senin', 'Selasa', 'Rabu', 'Kamis', 'Sabtu', 'Minggu', 'Jumat']
```

Perintah **append** akan menambahkan nilai pada posisi indeks paling terakhir.



## 2. Remove (hapus)

- Perintah ini digunakan untuk menghapus nilai yang ada pada list. Contoh:

```
1 list = ['senin', 'selasa', 'rabu', 'kamis', 'sabtu', 'minggu']
2 print(list)
3 list.remove('minggu')
4 print(list)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\venv\Script
['senin', 'selasa', 'rabu', 'kamis', 'sabtu', 'minggu']
['senin', 'selasa', 'rabu', 'kamis', 'sabtu']
```

Nilai pada list dengan nilai **'minggu'** terhapus dari dalam list.

- Bisa juga menggunakan. Contoh 2:

```
1 list = ['senin', 'selasa', 'rabu', 'kamis', 'sabtu', 'minggu']
2 print(list)
3 del list[3]
4 print(list)
```

Hasil

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\venv\Script
['senin', 'selasa', 'rabu', 'kamis', 'sabtu', 'minggu']
['senin', 'selasa', 'rabu', 'sabtu', 'minggu']
```

Menghapus nilai list menggunakan perintah del.

- Menghapus seluruh nilai. Contoh 3:

```
1 list = ['senin', 'selasa', 'rabu', 'kamis', 'sabtu', 'minggu']
2 print(list)
3 list.clear()
4 print(list)
```

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\venv\Script
['senin', 'selasa', 'rabu', 'kamis', 'sabtu', 'minggu']
[]
```

Nilai dalam list terhapus semua setelah menggunakan perintah **clear()**.

### 3. READ

#### a. List iteration

Melihat nilai pada list.

```
listExample = [40, 50, 20, 70, 80]

for item in listExample:
    print(item)
```

Hasil:

```
40
50
20
70
80
```

Berbeda dengan print list berdasarkan indeks, list iteration dapat ditambahkan kondisi percabangan.

Contoh:

```
for item in listExample:
    if item % 2 == 0:
        print(item)
```

Hasil:

```
40
20
80
```

Angka yang tampil adalah angka genap yang ada dalam list.

- b. Check item dalam sebuah list.

Digunakan untuk mengecek eksistensi item di dalam sebuah list.

```
listExample = [40, 55, 20, 75, 80]

if 40 in listExample:
    print("Terdapat angka 40 di dalam listExample")
```

Hasil:

```
collections.py
Terdapat angka 40 di dalam listExample
```

Muncul output berupa string sesuai dengan kondisi.

- c. Copy

Digunakan untuk mengubah dan menduplikat sebuah list.

```
listExample = [40, 55, 20, 75, 80]
listExample2 = listExample.copy()
listExample2[0] = 100
print(listExample)
print(listExample2)
```

Hasil:

```
[40, 55, 20, 75, 80]
[100, 55, 20, 75, 80]
```

## d. Concat

Digunakan untuk menggabungkan nilai dari beberapa list.

```
list = ['senin', 'selasa']  
list2 = ['kedua', 'kecil']  
list3 = ['tiga', 'ketiga']  
  
print(list+list2+list3)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\venv\Scr  
['senin', 'selasa', 'kedua', 'kecil', 'tiga', 'ketiga']
```

## e. Extend

Digunakan untuk menduplikasi dan menambahkan isi dari sebuah list ke dalam list lain.

```
list = ['senin', 'selasa']  
list2 = ['kedua', 'kecil']  
list3 = ['tiga', 'ketiga']  
list.extend(list2)  
print(list)  
print(list2)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\hellopy  
['senin', 'selasa', 'kedua', 'kecil']  
['kedua', 'kecil']
```

## f. Sort (urutan)

```
list = ['senin', 'rabu', 'selasa']
listangka = [3, 6, 1, 2, 8, 10, 4, 7]
print("sebelum disort")
print(list)
print(listangka)
list.sort()
listangka.sort()
print("setelah disort")
print(list)
print(listangka)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\he
sebelum disort
['senin', 'rabu', 'selasa']
[3, 6, 1, 2, 8, 10, 4, 7]
setelah disort
['rabu', 'selasa', 'senin']
[1, 2, 3, 4, 6, 7, 8, 10]
```

## g. Reverse

Digunakan untuk membalikan isi dari list (termasuk index)

```
list = ['senin', 'rabu', 'selasa']
listangka = [3, 6, 1, 2, 8, 10, 4, 7]
print("sebelum direverse")
print(list)
print(listangka)
list.reverse()
listangka.reverse()
print("setelah direverse")
print(list)
print(list[1])
print(listangka)
```

```
C:\Users\ritha\PycharmProjects\hellop
sebelum disort
['senin', 'rabu', 'selasa']
[3, 6, 1, 2, 8, 10, 4, 7]
setelah disort
['selasa', 'rabu', 'senin']
rabu
[7, 4, 10, 8, 2, 1, 6, 3]

Process finished with exit code 0
```

### D. Update (ubah)

Digunakan untuk mengubah nilai dari list berdasarkan indeks dari nilai yang ingin diubah.

Contoh:

```
list = ['senin', 'rabu', 'selasa']  
print(list)  
list[1] = 'minggu'  
print(list)
```

Hasil:

```
['senin', 'rabu', 'selasa']  
['senin', 'minggu', 'selasa']
```

Nilai pada list indeks ke 1 ('rabu') diubah menjadi 'minggu'.

## TUPLE

Tuple hampir mirip dengan List, tapi perbedaannya dengan list adalah nilai dalam Tuple tidak dapat diubah.

```
tuple = ('senin', 'selasa', 'rabu')  
print(tuple)  
print(tuple[1])
```

Hasil:

```
C:\Users\ritha\PycharmProjects\  
('senin', 'selasa', 'rabu')  
selasa
```

## DICTIONARY

Digunakan untuk menyimpan data dalam bentuk key:value

```
mahasiswa = {  
    "nim": 1517227,  
    "nama": "Budi Kurniawan",  
    "semester": 6  
}  
  
print(mahasiswa)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\helloycharm.py\venv\Scripts>python.exe C:\Users\ritha\PycharmProjects\helloycharm.py/  
{'nim': 1517227, 'nama': 'Budi Kurniawan', 'semester': 6}  
  
Process finished with exit code 0
```

## Menyimpan Dictionary ke dalam List.

```
mahasiswa = {  
    "nim": 1517227,  
    "nama": "Budi Kurniawan",  
    "semester": 6  
}  
  
mahasiswa2 = {  
    "nim": 1517228,  
    "nama": "Reza Rahadian",  
    "semester": 7  
}  
  
list = [mahasiswa, mahasiswa2]  
print(list)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\helloycharm.py\venv\Scripts>python.exe C:\Users\ritha\PycharmProjects\helloycharm.py/  
[{'nim': 1517227, 'nama': 'Budi Kurniawan', 'semester': 6}, {'nim': 1517228, 'nama': 'Reza Rahadian', 'semester': 7}]
```



**Print dictionary berdasarkan key.**

Contoh:

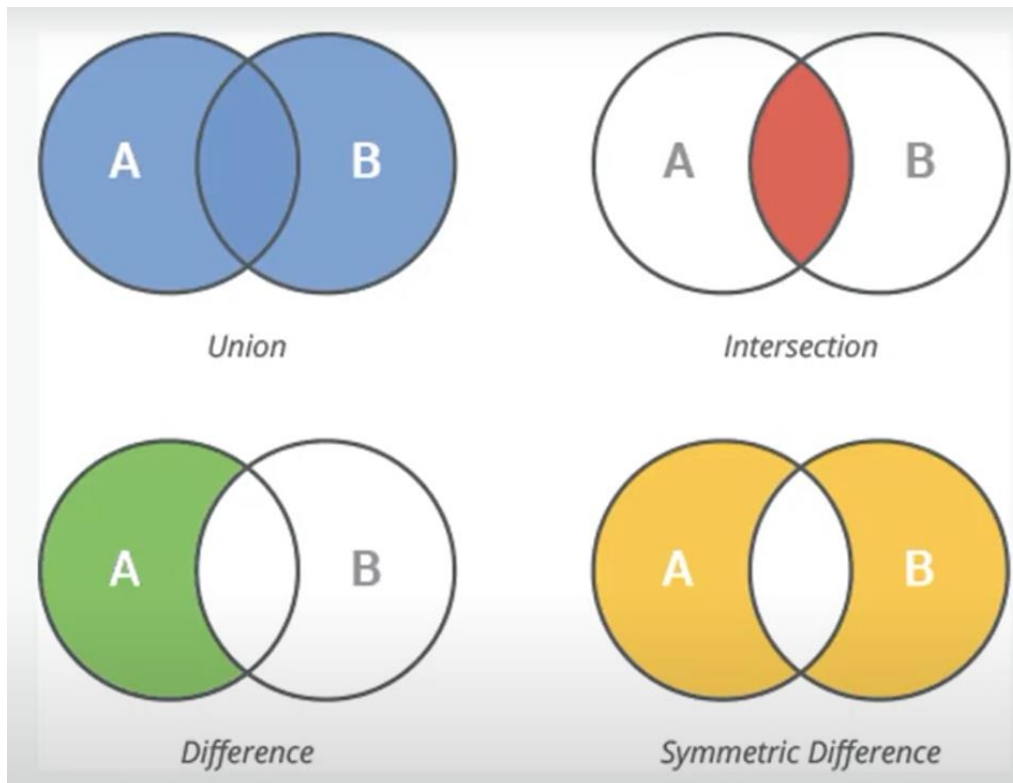
```
mahasiswa = {  
    "nim": 1517227,  
    "nama": "Budi Kurniawan",  
    "semester": 6  
}  
  
print(mahasiswa["nim"])  
print(mahasiswa["nama"])
```

Hasil:

```
C:\Users\ritha\PycharmProj  
1517227  
Budi Kurniawan
```

## SET

Set adalah kumpulan elemen unik yang tidak memiliki urutan.



Union = Mencari semua elemen yang ada di A dan B

Intersection = Mencari elemen yang sama antara A dan B

Difference = Mencari elemen yang dimiliki A tapi tidak dimiliki B, dan sebaliknya.

Symmetric Difference = Mencari elemen yang tidak sama antara A dan B.

## a. Program set (union)

```
A = {1, 3, 4, 7, 12}
B = {2, 3, 5, 8, 12}

var_union = A.union(B)
print(var_union)
```

Hasil:

```
C:\Users\ritha\PycharmProjects\
{1, 2, 3, 4, 5, 7, 8, 12}
Process finished with exit code
```

Cat: nilai duplikat hanya akan muncul 1 kali.

## b. Program set (intersection)

```
A = {1, 3, 4, 7, 12}
B = {2, 3, 5, 8, 12}

var_intersection = A.intersection(B)
print(var_intersection)
```

Hasil:

```
C:\Users\ritha\PycharmP
{3, 12}
```

## c. Program set Difference

```
A = {1, 3, 4, 7, 12}
B = {2, 3, 5, 8, 12}

var_difference1 = A.difference(B)
print(var_difference1)
var_difference2 = B.difference(A)
print(var_difference2)
```

Hasil:

```
C:\Users\rith
{1, 4, 7}
{8, 2, 5}
```

## d. Program set Symmetric Difference

```
A = {1, 3, 4, 7, 12}
B = {2, 3, 5, 8, 12}

sym_difference = A.symmetric_difference(B)
print(sym_difference)
```

Hasil:

```
C:\Users\ritha\PycharmPro
{1, 2, 4, 5, 7, 8}
```

**Tandai Selesai**

## **BAB VI**

# **FUNCTION**



## FUNCTION

Memungkinkan suatu kumpulan kode untuk digunakan kembali dalam suatu program.

Python menyediakan beberapa built-in function seperti len(), print(), dan type().

### Function syntax.

- Function name: nama dari function
- Argument: nilai yang diberikan ke function
- Function body: isi dari sebuah function
- Return value: hasil yang dikembalikan oleh function

### Create and Call

Dalam python kita menggunakan def() untuk membuat sebuah fungsi.

```
def greet():  
    print("Hello, stranger!")  
  
greet()
```

Hasil:

```
Hello, stranger!
```

## PASSING ARGUMENTS

Contoh 1:

```
def greet(name):  
    print("Hello, " + name)  
  
greet("Budi")  
greet("Ahmad")  
greet("Cecep")
```

Hasil:

```
Hello, Budi  
Hello, Ahmad  
Hello, Cecep
```

Contoh 2:

```
def greet(nama, umur):  
    print("Selamat datang ", nama)  
    print("umur anda adalah", umur)  
  
greet("Budi Kurniawan", 25)
```

Hasil 2:

```
C:\Users\ritha\PycharmProjects\he  
Selamat datang Budi Kurniawan  
umur anda adalah 25
```

## LAMBDA FUNCTION

Memungkinkan untuk mempersingkat kode program suatu function

Contoh:

```
nilai = lambda x: x+2
~~~~~
print(nilai(2))
```

hasil:

```
C:\Users\ritha
4
```

Contoh:

```
bilangan = 10
~~~~~
nilai = lambda x: x+2
~~~~~
print(nilai(bilangan))
```

hasil:

```
C:\Users\ritha
12
```



**SOAL LATIHAN Membuat Program.****1. Harga menu menggunakan function**

```
def a():  
    harga = 10000  
    jumlah = int(input("masukan jumlah pembelian :"))  
    total = harga * jumlah  
    print("total pembayaran adalah :", total)  
  
def b():  
    harga = 15000  
    jumlah = int(input("masukan jumlah pembelian :"))  
    total = harga * jumlah  
    print("total pembayaran adalah :", total)  
  
def c():  
    harga = 20000  
    jumlah = int(input("masukan jumlah pembelian :"))  
    total = harga * jumlah  
    print("total pembayaran adalah :", total)  
  
print("====daftar menu====")  
pilihan = int(input("masukan pilihan : "))  
if pilihan==1:  
    a()  
elif pilihan==2:  
    b()  
elif pilihan==3:  
    c()  
else:  
    print("tidak tersedia")
```

**Tandai Selesai**

## **BAB VII**

### **CLASS dan OBJECT**



## Class

Python adalah sebuah bahasa pemrograman Object Oriented Language. Hampir semua hal di python merupakan sebuah objek yang memiliki property dan method.

Class adalah blueprint untuk membuat sebuah object.

### Contoh 1:

```
class Person:
    nama = "Budi"

obj = Person()
print(type(obj))
print(obj.nama)
```

```
<class '__main__.Person'>
Budi
```

### Contoh 2:

```
class hard:
    warna = "merah"
    tipe = "baja"
    power_level = 1000

class medium:
    warna = "biru"
    tipe = "batu"
    power_level = 500

class easy:
    warna = "kuning"
    tipe = "kayu"
    power_level = 200

level_1 = easy()
print(level_1.warna)
print(level_1.tipe)
print(level_1.power_level)
```

```
C:\Users\rit
kuning
kayu
200
```

## `__init__()`

Fungsi `__init__()` akan otomatis dipanggil ketika kita membuat sebuah object. Dengan fungsi ini, kita dapat mendeklarasi properti yang dimiliki oleh sebuah class.

Contoh 1:

```
class person:
    def __init__(self, nama, umur, skor):
        self.nama = nama
        self.umur = umur
        self.skor = skor

p1 = person("Budi", 23, 230)
p2 = person("Reza", 25, 300)

print(p1.__dict__)
print(p2.nama)
print(p2.umur)
print(p2.skor)
```

Contoh 2:

```
C:\Users\ritha\PycharmProjects\hellopychar
{'nama': 'Budi', 'umur': 23, 'skor': 230}
Reza
25
300
```

```
class person:
    def __init__(self, nama, umur, skor):
        self.nama = nama
        self.umur = umur
        self.skor = skor
    def salam(self):
        print("Selamat datang", self.nama, " di kelas kursus. skor kamu adalah ", self.skor)

p1 = person("Budi", 23, 230)
p2 = person("Reza", 25, 300)

p1.salam()
```

Hasil:

```
C:\Users\ritha\PycharmProjects\hellopycharm.py\venv\Scripts\python.exe
Selamat datang Budi di kelas kursus. skor kamu adalah 230
```

## INHERITANCE

Inheritance adalah sebuah proses dimana sebuah class mengambil semua properti dan semua metode dari kelas lain.

```
class hewan: #PARENT CLASS
    def __init__(self, nama, umur):
        self.nama = nama
        self.umur = umur
    def panggil(self):
        print("Cuss")

class kucing(hewan): #CHILD CLASS
    def __init__(self, nama, umur, bulu, berat):
        super().__init__(nama, umur)
        self.bulu = bulu
        self.berat = berat

class sapi(hewan): #CHILD CLASS
    def __init__(self, nama, umur, tipe):
        super().__init__(nama, umur)
        self.tipe = tipe

p1 = hewan("squidword", 25)
p2 = kucing("Patrick", 20, "pink", 5)
p3 = sapi("otis", 20, "belang")

p1.panggil()
p2.panggil()
p2.panggil()
print(p1.nama, p1.umur)
print(p2.nama, p2.umur)
print(p3.nama, p3.umur)
```

Hasil:

```
C:\Users\ritha\Pychar
Cuss
Cuss
Cuss
squidword 25
Patrick 20
otis 20
```



Penjelasan:

**P2** dan **P3** (dalam hal ini kucing), tetap bisa menginputkan nama dan umur walaupun kedua class tersebut tidak menginisialisasikan nama dan umur karena telah ter-*inheritance* dari kelas **hewan**.

<b>Tandai Selesai</b>	
-----------------------	--

## **BAB VIII**

# **FILE PROCESSING**





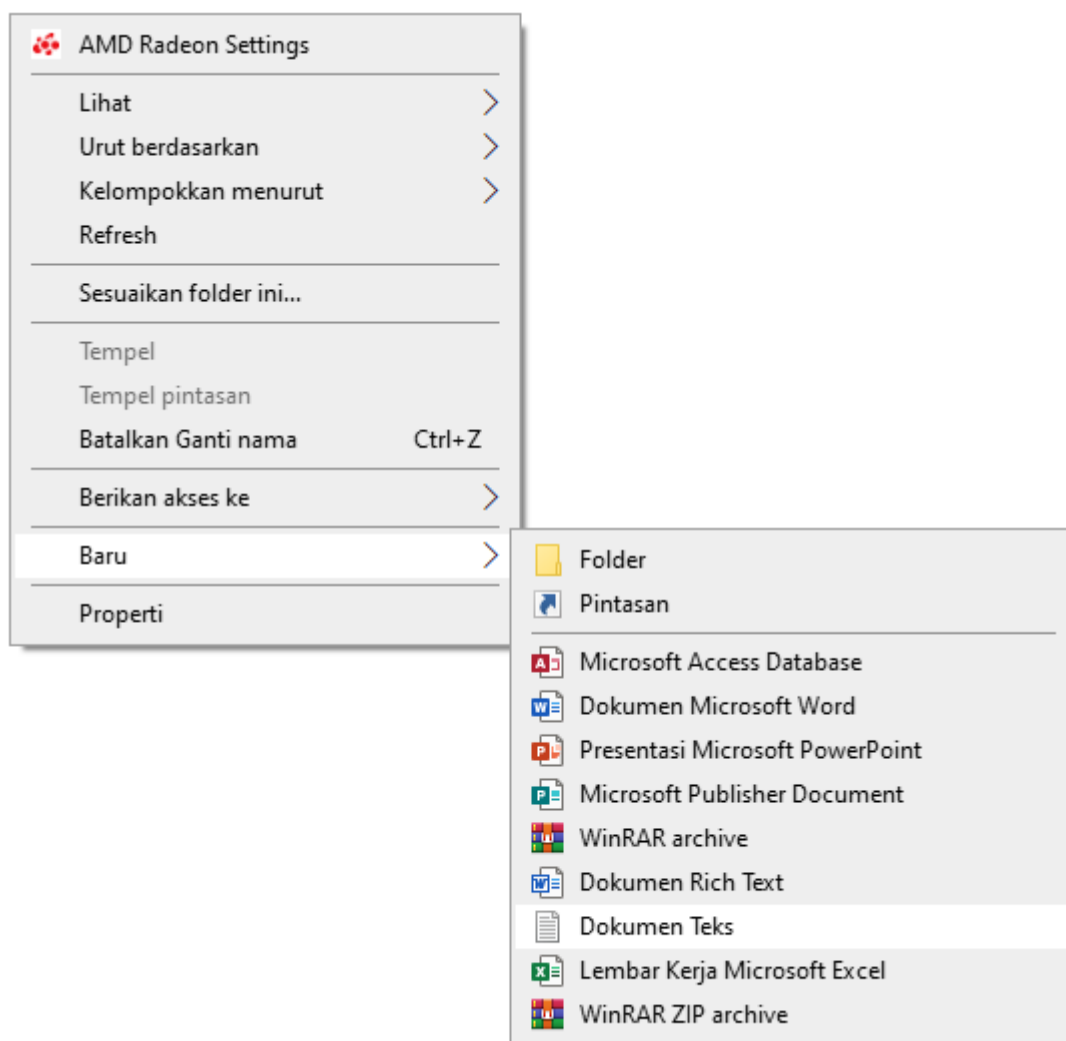
## File Processing

Python mendukung proses file handling seperti membaca dan menulis file. Berbeda dengan pemrograman lain, python memiliki konsep yang sangat sederhana untuk file.

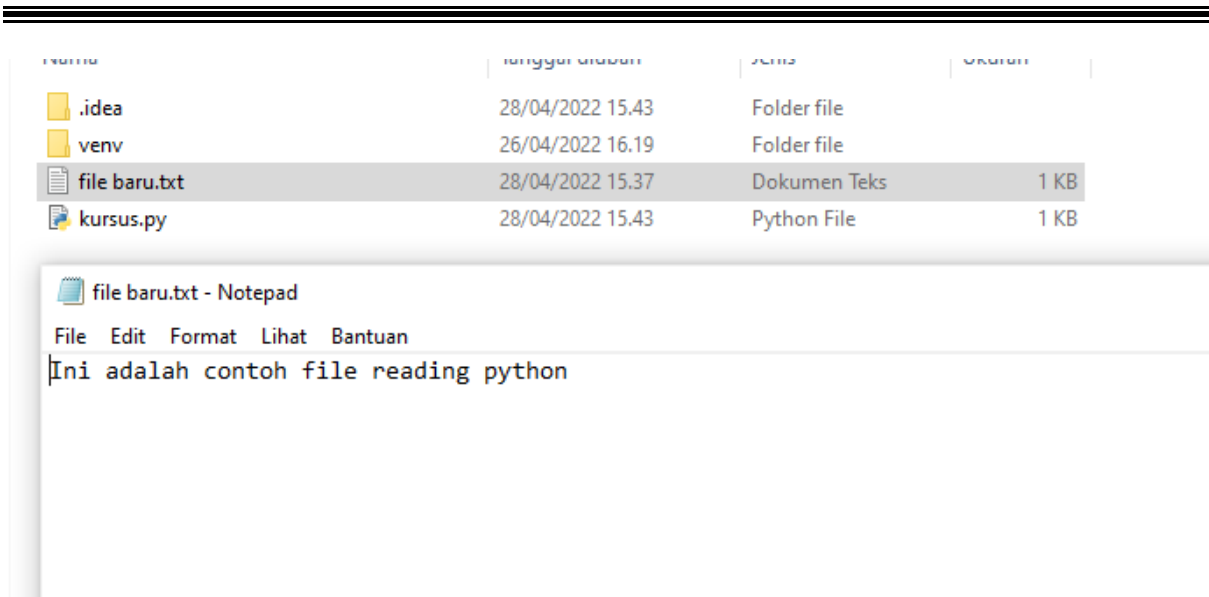
### Open() Function - reading

Contoh:

Buatlah sebuah file baru dengan format txt dalam 1 folder project python Anda. Lalu klik kanan dan pilih new/baru. Kemudian pilih dokumen teks. Seperti pada tampilan berikut.



Lalu ketikan:



Lalu save atau simpan dengan nama apa saja. (dalam contoh: file baru.txt)

Buka kembali code editor Anda lalu ketikkan kode program berikut:

```
file = open("file baru.txt", "r")  
  
print(file.read())
```

Hasil:

```
C:\Users\ritha\PycharmProjects\hellopychar  
Ini adalah contoh file reading python
```

Isi dari file akan tampil.

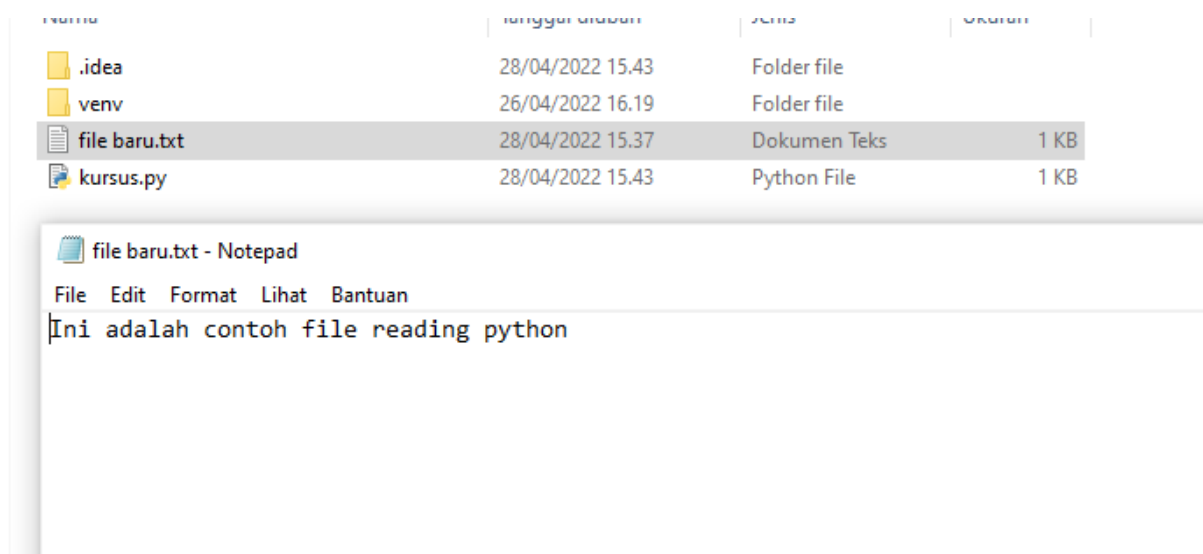
## Open() Function – writing

Digunakan untuk menulis atau menimpa isi dalam file. Contoh :

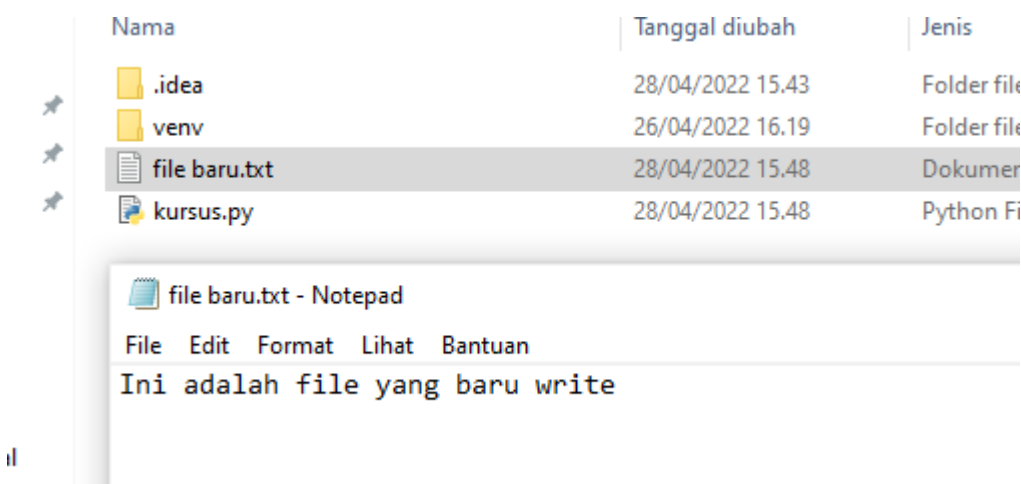
```
file = open("file_baru.txt", "w")  
file.write("Ini adalah file yang baru write")  
file.close()
```

Maka hasilnya dapat dilihat pada file yang sebelumnya dibuat.

Dari teks seperti ini: ↓



Menjadi seperti ini: ↓



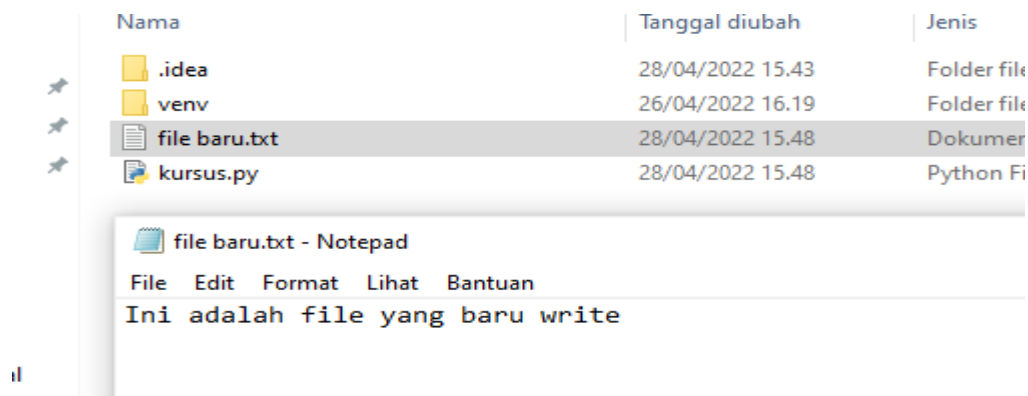
## Open() Function – append

Digunakan untuk menambahkan isi file tanpa menimpa isi teks sebelumnya. Contoh:

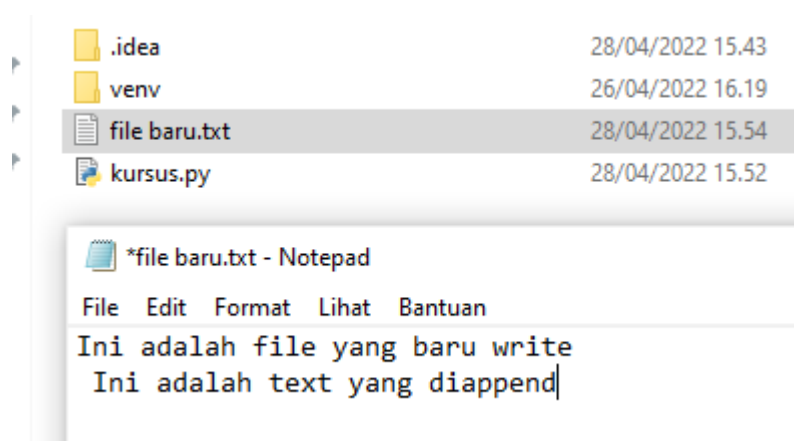
```
file = open("file_baru.txt", "a")
file.write("\n Ini adalah text yang diappend")
file.close()
```

Maka hasilnya dapat dilihat pada file yang sebelumnya dibuat.

Dari file seperti ini: ↓



Menjadi seperti ini: ↓



**Tandai Selesai**