# DAFATR ISI

[DAFATR ISI 1](#_Toc375854)

[1. Sejarah Python 2](#_Toc375855)

[2. Mengapa Harus Python? 2](#_Toc375856)

[3. Siapa Saja Pengguna Python 4](#_Toc375857)

[4. Memulai Program Python 5](#_Toc375858)

[5. Menulis Bahasa Python 6](#_Toc375859)

[6. Kata Kunci Dan Pengenal 7](#_Toc375860)

[7. Pengenal (Identifier) Python 8](#_Toc375861)

[6. Sintaks Dasar Python 9](#_Toc375862)

[7. Operator Pada Python 12](#_Toc375863)

# Sejarah Python

Python diciptakan oleh Guido van Rossum pertama kali di Scitchting Mathematisch Centrum (CWI) di Belanda pada awal tahun 1990-an. Bahasa python terinspirasi dari bahasa pemrograman ABC. Sampai sekarang, Guido masih menjadi penulis utama untuk python, meskipun bersifat open source sehingga ribuan orang juga berkontribusi dalam mengembangkannya. Di tahun 1995, Guido melanjutkan pembuatan python di Corporation for National Research Initiative (CNRI) di Virginia Amerika, dimana dia merilis beberapa versi dari python.

Pada Mei 2000, Guido dan tim Python pindah ke BeOpen.com dan membentuk tim BeOpen PythonLabs. Di bulan Oktober pada tahun yang sama, tim python pindah ke Digital Creation (sekarang menjadi Perusahaan Zope). Pada tahun 2001, dibentuklah Organisasi Python yaitu Python Software Foundation (PSF). PSF merupakan organisasi nirlaba yang dibuat khusus untuk semua hal yang berkaitan dengan hak intelektual Python. Perusahaan Zope menjadi anggota sponsor dari PSF.

Semua versi python yang dirilis bersifat open source. Dalam sejarahnya, hampir semua rilis python menggunakan lisensi GFL-compatible. Berikut adalah versi mayor dan minor python berikut tanggal rilisnya. Nama python sendiri tidak berasal dari nama ular yang kita kenal. Guido adalah penggemar grup komedi Inggris bernama Monty Python. Ia kemudian menamai bahasa ciptaannya dengan nama Python.

# Mengapa Harus Python?

Mengapa harus Python? Bukankah masih banyak bahasa pemrograman lain di luar sana? Apa kelebihan Python? Pertanyaan – pertanyaan tersebut sering menjadi pertanyaan yang muncul sebelum seseorang mempelajari Python. Berikut adalah beberapa di antara kelebihan Python:

Python adalah bahasa pemrograman yang populer. Per September 2018, Python berada di urutan ke 3 [bahasa program yang paling populer di dunia](https://www.tiobe.com/tiobe-index/).

1. Python relatif lebih mudah dipelajari dan digunakan dibandingkan bahasa pemrograman lain. Sintaksnya sederhana, mudah dibaca dan diingat karena filosofi python sendiri menekankan pada aspek kemudahan dibaca (*readibility*). Kode python mudah ditulis dan mudah dibaca, sehingga lebih mudah diperbaiki kalau ada kesalahan, dan juga mudah untuk dipelihara.
2. Selain lebih mudah dibaca, python juga lebih efisien dibandingkan bahasa lain seperti C, C++, maupun Java. Untuk melakukan sesuatu dengan 5 baris kode pada bahasa lain, bisa jadi di python hanya diperlukan 1 baris kode. Hal ini menyebabkan pembuatan program dalam Python menjadi lebih ringkas dan lebih cepat dibandingkan bahasa lain.
3. Python merupakan bahasa multifungsi. Dengan python Anda bisa melakukan berbagai hal mulai dari memproses teks, membuat website, membuat program jaringan, robotika, data mining, sampai dengan kecerdasan buatan. Dengan python Anda bisa membuat aplikasi berbasis desktop maupun berbasis smartphone.
4. Python kaya akan dukungan *library* (pustaka) standar. Tersedia banyak sekali modul-modul dan ekstensi program yang sudah siap Anda pakai untuk membuat program sesuai kebutuhan Anda. Komunitas python adalah komunitas yang sangat aktif mengembangkan python sehingga menjadi bahasa yang sangat handal.
5. Python bisa berinteraksi dengan bahasa lain. Kode python bisa memanggil oleh bahasa C, C++, dan sebaliknya juga bisa dipanggil dari bahasa lain.
6. Tapi, itu hanya kelebihannya. Terus, apa kekurangannya? Python adalah bahasa [interpreter](https://id.wikipedia.org/wiki/Penerjemah_(komputasi)). Kekurangan python dibanding bahasa lain yang menggunakan [kompiler](https://id.wikipedia.org/wiki/Kompilator) adalah ‘sedikit’ lebih lambat pada saat dijalankan bila dibandingkan bahasa C maupun C++. Tapi hal inipun sangat bersifat relatif. Tergantung dari besar ukuran program yang dibuat.

Untuk program besar yang membutuhkan kecepatan pemrosesan tinggi mungkin Python kalah cepat dari bahasa C, tapi untuk hal selain itu Python lebih mudah dan lebih baik dari bahasa lain. Selain itu, kode sumber sekarang sudah dioptimasi menggunakan bahasa C, sehingga kecepatannya juga sudah sangat mendekati kecepatan bahasa C. Spesifikasi komputer juga sekarang ini sudah semakin tinggi sehingga bisa memproses program dengan cepat, sehingga sering kali ini tidak menjadi hal penting dan bisa diabaikan.

# Siapa Saja Pengguna Python

Kalau Anda adalah pemula yang ingin belajar Python, atau baru menggunakan Python, bisa jadi Anda bertanya siapa saja yang menggunakan Python. Daftar berikut pasti membuat Anda mengetahui bahwa Python adalah sebuah bahasa yang populer dan layak dipelajari.

1. Google adalah perusahaan besar yang menggunakan banyak kode Python di dalam mesin pencarinya. Dan mesin pencari google adalah yang paling terkenal di dunia.
2. Youtube, situs video terbesar dan terpopuler di dunia, sebagian besar kodenya ditulis dalam bahasa Python.
3. Facebook, media sosial terbesar di dunia, menggunakan Tornado, sebuah framework Python untuk menampilkan timeline.
4. Instagram, siapa yang tidak kenal. Instagram menggunakan Django, framework python sebagai *mesin* pengolah sisi server dari aplikasinya.
5. Pinterest, banyak menggunakan python untuk membangun aplikasinya.
6. Dropbox, barangkali Anda adalah salah seorang pengguna layanan ini. Dropbox menggunakan python baik di sisi server maupun di sisi pengguna layanannya.
7. Quora, salah satu situs tanya jawab terbesar di dunia, dibangun menggunakan Python.
8. NASA, badan antariksa Amerika ini menggunakan Python untuk bidang sainsnya.
9. NSA, badan mata – mata Amerika banyak menggunakan Python untuk analisa kriptografi dan intelijen.
10. Industrial Light & Magic, Pixar, banyak menggunakan Python dalam animasi movie.
11. Blender, Maya, software pembuat animasi 3D terkenal, menggunakan Python sebagai salah satu bahasa skrip pemrogramannya.
12. Raspberry Pi, komputer mini yang banyak digunakan sebagai mikrokontroller, menggunakan Python sebagai bahasa utamanya.
13. ESRI, produsen terkenal pembuat software pemetaan GIS banyak menggunakan Python di produknya.

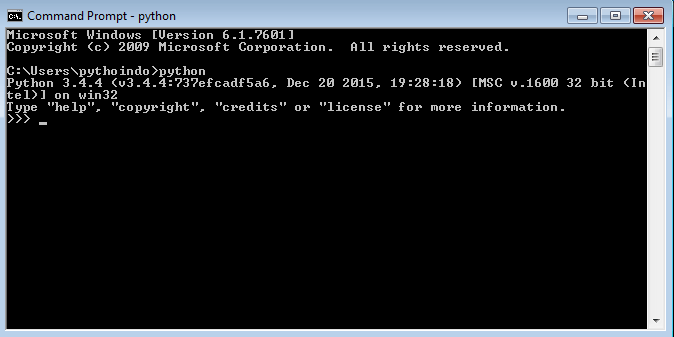
Selain yang di atas, masih banyak lagi aplikasi dari perusahaan besar maupun individu pengguna Python yang tidak disebutkan di sini. Anda baca di <https://siftery.com/python> maupun di

# Memulai Program Python

Python merupakan bahasa interpreter yang artinya kode python akan dieksekusi baris perbaris. Bila ada kesalahan program di tengah eksekusi, maka program akan berhenti, kecuali menggunakan penanganan eksepsi. Ada dua cara menjalankan interpreter python ini untuk selanjutnya kita hanya akan menyebut python saja. Yang pertama adalah dengan menggunakan program **command prompt** (cmd) di windows. Yang kedua menggunakan program IDLE yang merupakan bawaan python sendiri. Di sini kita akan membahas keduanya.

1. Menjalankan python dari command promp (cmd)

Buka command promp atau cmd di windows dan ketikan python, lalu enter



1. Menjalankan python dengan IDLE

IDLE merupakan program editor bawaan Python. IDLE bisa digunakan dalam untuk menjalankan python dalam mode interaktif. Arti interaktif di sini adalah ketika Anda mengetikkan perintah, maka hasilnya akan langsung ditampilkan. Untuk menjalankan IDLE, lakukan langkah – langkah berikut:

# Menulis Bahasa Python

Seringkali, program “Hello World!” digunakan untuk mengenalkan suatu bahasa pemrograman ke pemula. Program “Hello World!” adalah sebuah program sederhana yang menampilkan kalimat “Hello World!” di monitor. Akan tetapi, Python adalah bahasa yang paling mudah dipelajari, dan membuat program “Hello World!” hanya sesederhana menuliskan perintah print("Hello World!"). Oleh karena itu, kita akan lebih memilih untuk membuat program penjumlahan dua bilangan.

*Contoh :   
# Menambahkan dua bilangan  
bil1 = 15  
bil2 = 25  
jumlah = bil1+bil2  
print(jumlah)*

**Penjelasan Program**Python adalah bahasa pemrograman yang menggunakan interpreter. Pada interpreter program akan dieksekusi baris perbaris. Bila ada error maka program akan terhenti, kecuali dengan menggunakan metode penanganan eksepsi.Pada program di atas, baris 1 adalah komentar program. Interpreter tidak memproses komentar. Komentar hanya untuk penjelasan kode agar dipahami oleh manusia yang membacanya.

Pada baris 2 kita menciptakan sebuah objek bilangan yaitu 15. Kita membuat variabel bil1 menunjuk ke objek 15. Dengan kata lain, objek 15 ditugaskan ke variabel bil1. Penjelasan untuk baris 3 sama dengan penjelasan untuk baris ke 2. Selanjutnya pada baris ke 4, objek yang ditunjuk oleh bil1 yaitu 15 dan yang ditunjuk oleh bil2 yaitu 25 dijumlahkan. Hasilnya ditugaskan ke variabel jumlah. Di baris terakhir, kita menggunakan fungsi bawaan (builtin) python, yaitu print() untuk menampilkan variabel jumlah ke monitor. Demikian program sederhana menggunakan python. Untuk selanjutnya, Anda akan belajar dasar Python dan program yang lebih kompleks.

# Kata Kunci Dan Pengenal

Kata kunci adalah kata – kata yang sudah ditetapkan dan dipakai Python untuk mendefinisikan sintaks dan struktur bahasa python sendiri. Kata kunci tidak bisa digunakan sebagai nama variabel, fungsi, maupun nama untuk pengenal lainnya. Keyword bersifat case sensitive. Artinya huruf besar berbeda dengan huruf kecil. Sejauh ini, ada 33 kata kunci di Python. Semua kata kunci menggunakan huruf kecil kecuali True, False, dan None. Daftar kata kunci Python adalah seperti berikut:

|  |  |  |  |
| --- | --- | --- | --- |
| True | def | if | raise |
| False | del | import | return |
| None | elif | in | try |
| and | else | is | while |
| as | except | lambda | with |
| assert | finally | nonlocal | yield |
| break | for | not |  |
| class | from | or |  |
| continue | global | pass |  |

# Pengenal (Identifier) Python

Pengenal adalah nama yang diberikan untuk menamai hal di python seperti variabel, fungsi, kelas, dan lain sebagainya. Fungsinya adalah untuk membedakan antara satu entitas dengan yang lainnya.

**Aturan Penulisan Pengenal**

1. Pengenal bisa terdiri dari kombinasi huruf kecil (a-z) atau huruf besar (A-Z), angka ( 0-9 ), dan underscore ( \_ ). namaKaryawan, bilangan\_2, dan jenis\_kenderaan\_umum adalah nama variabel yang valid.
2. Pengenal tidak bisa dimulai menggunakan angka. 1nama adalah invalid sedangkan nama1 adalah valid.
3. Kata kunci tidak bisa dipakai sebagai nama pengenal.
4. Karakter spesial seperti !, @, %, $, dan sebagainya tidak bisa digunakan sebagai pengenal.
5. Python bersifat case sensitive. Huruf besar dan kecil dianggap berbeda. Jadi Variabel, variabel, dan VARIABEL, adalah 3 buah nama yang berbeda.

Selain hal tersebut, ada beberapa hal lagi yang perlu diketahui terkait nama pengenal. Nama pengenal sebaiknya menggunakan nama yang deskriptif. Walaupun menuliskan nama variabel panjang, p = 10 adalah benar, tapi menuliskannya menjadi panjang = 10 adalah jauh lebih baik agar nantinya Anda mengerti apa yang dimaksud di kode program Anda.

Kemudian untuk nama variabel yang panjang, ada 2 cara penulisan. Pertama dengan menggunakan metode underscore dan yang kedua dengan metode camelCase. Misalnya untuk menamai sebuah variabel diameter persegi panjang, dengan metode underscore bisa ditulis diameter\_persegi\_panjang. Sedangkan dengan metode camelCase ditulis diameterPersegiPanjang.

# Sintaks Dasar Python

Python merupakan bahasa pemrograman yang memiliki sintaks yang sederhana dan mudah dimengerti. Python memiliki filosofi bahwa kode program harus mudah dibaca. Filosofi python tertuang dalam PEP 20.

**Statement (Pernyataan) di Python**

Semua perintah yang bisa dieksekusi oleh Python disebut statement. Misalnya, a = 1 adalah sebuah statement penugasan. Selain statement penugasan ada statement lain seperti statement if, statement for, dan lain sebagainya.

**Statement Multibaris**

Di Python, akhir dari sebuah statement adalah karakter baris baru (newline). Kita dapat membuat sebuah statement terdiri dari beberapa baris dengan menggunakan tanda backslash ( \ ). Misalnya:

a = panjang1 + panjang2 + \

panjang3 + \

panjang4

Statement yang ada di dalam tanda kurung [ ], { }, dan ( ) tidak memerlukan tanda \. Contohnya:

nama\_bulan = ['Januari', 'Maret', 'Juni', 'September']

**Baris Dan Indentasi**

Python tidak menggunakan tanda { } untuk menandai blok / grup kode. Blok kode di python menggunakan tanda indentasi (spasi). Jumlah spasi untuk setiap baris yang ada dalam satu blok kode harus sama. Contoh yang benar adalah sebagai berikut:

if nilai <= 5:

print("Nilai merah")

print("Tidak lulus")

else:

print("Nilai biru")

print("Lulus")

 Bila indentasi dalam satu grup kode tidak sama, python akan menampilkan sintaks error.

if True:

print ("Jawab")

print ("Benar")

else:

print ("Jawab")

print ("Salah")

SyntaxError: unexpected indent

**Tanda Kutip di Python**

Python menggunakan tanda kutip tunggal (‘), ganda (“), maupun triple (”’ atau “””) untuk menandai string, sepanjang stringnya diawali oleh tanda kutip yang sama di awal dan akhir string. Tanda kutip tiga digunakan untuk string multibaris. Ketiga contoh berikut, semuanya adalah benar,

kata = 'kata'

kalimat = "Ini adalah kalimat'

paragraf = """Ini adalah paragraf. Paragraf

terdiri dari beberapa baris."""

**Komentar di Python**

Tanda pagar ( # ) digunakan untuk menandai komentar di python. Komentar tidak akan diproses oleh interpreter Python. Komentar hanya berguna untuk programmer untuk memudahkan memahami maksud dari kode.

# Komentar pertama

print("Hello World!") # Komentar kedua

Kode di atas akan menghasilan keluaran:

Hello World!

Python tidak memiliki fitur komentar multibaris. Kita harus mengomentari satu persatu baris seperti berikut:

# Ini komentar

# Ini juga adalah komentar

# Ini juga masih komentar

# Operator Pada Python

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.

Python memiliki sejumlah operator, yaitu:

* Operator Aritmatika
* Operator Perbandingan
* Operator Penugasan
* Operator Logika
* Operator Bitwise
* Operator Identitas
* Operator Keanggotaan

Mari kita tinjau satu persatu.

**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, mengurangkan 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 |

**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 |

**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 yang ada di sebelah kiri | c = a + b menugaskan a + b ke c |
| += | Menambahkan operand yang di kanan dengan operand yang ada di kiri dan hasilnya di tugaskan 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 di tugaskan 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 di tugaskan 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 di tugaskan 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 kanan dengan operand di kiri dan hasilnya di simpan di operand yang di kiri | c %= a sama dengan c = c % a |

**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 |

**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 | ~x = -11 (1111 0101) |
| ^ | Bitwise XOR | x ^ y = 14 (0000 1110) |
| >> | Bitwise right shift | x>> 2 = 2 (0000 0010) |
| << | Bitwise left shift | x<< 2 = 40 (0010 1000) |

**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 |

**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 |

# Percabangan

Percabangan adalah cara yang digunakan untuk mengambil keputusan apabila di dalam program dihadapkan pada kondisi tertentu. Jumlah kondisinya bisa satu, dua atau lebih. Percabangan mengevaluasi kondisi atau ekspresi yang hasilnya benar atau salah .  Kondisi atau ekspresi tersebut disebut ekspresi boolean. Hasil dari pengecekan kondisi adalah True atau False. Bila benar (True), maka pernyataan yang ada di dalam blok kondisi tersebut akan dieksekusi. Bila salah (False), maka blok pernyataan lain yang dieksekusi.

Di Python ada 3 jenis pernyataan yang digunakan untuk percabangan, yaitu sebagai berikut:

|  |  |  |
| --- | --- | --- |
| **No** | **Pernyataan** | **Deskripsi** |
| 1 | if | Pernyataan **if** terdiri dari ekspresi *boolean*diikuti oleh satu baris atau lebih pernyataan. |
| 2 | if…else | Bila pernyataan **if** benar, maka blok pernyataan **if** dieksekusi. Bila salah, maka blok pernyataan **else** yang dieksekusi. |
| 3 | if…elif…else | Disebut juga if bercabang. Bila ada kemungkinan beberapa kondisi bisa benar maka digunakan pernyataan **if…elif** atau **if…elif…else** |

Percabangan

Percabangan adalah cara yang digunakan untuk mengambil keputusan apabila di dalam program dihadapkan pada kondisi tertentu. Jumlah kondisinya bisa satu, dua atau lebih.

Percabangan mengevaluasi kondisi atau ekspresi yang hasilnya benar atau salah .  Kondisi atau ekspresi tersebut disebut ekspresi boolean. Hasil dari pengecekan kondisi adalah True atau False. Bila benar (True), maka pernyataan yang ada di dalam blok kondisi tersebut akan dieksekusi. Bila salah (False), maka blok pernyataan lain yang dieksekusi.

Di Python ada 3 jenis pernyataan yang digunakan untuk percabangan, yaitu sebagai berikut:

|  |  |  |
| --- | --- | --- |
| **No** | **Pernyataan** | **Deskripsi** |
| 1 | if | Pernyataan **if** terdiri dari ekspresi booleandiikuti oleh satu baris atau lebih pernyataan. |
| 2 | if…else | Bila pernyataan **if** benar, maka blok pernyataan **if** dieksekusi. Bila salah, maka blok pernyataan **else** yang dieksekusi. |
| 3 | if…elif…else | Disebut juga if bercabang. Bila ada kemungkinan beberapa kondisi bisa benar maka digunakan pernyataan **if…elif** atau **if…elif…else** |

Berikut adalah penjelasan masing-masing pernyataan.

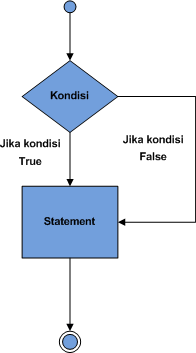
**Pernyataan if**

Pernyataan if menguji satu buah kondisi. Bila hasilnya benar maka pernyataan di dalam blok if tersebut dieksekusi. Bila salah, maka pernyataan tidak dieksekusi. Sintaksnya adalah seperti berikut:

if tes kondisi:

blok pernyataan if

Gambar diagram alir untuk pernyataan if adalah seperti berikut:



*# Bila bilangan positif, tampilkan pesan*

*angka = 5*

*if angka > 0:*

*print(angka, "adalah bilangan positif.")*

*angka = -1*

*# yang berikut akan bernilai False sehingga tidak dieksekusi*

*if angka > 0:*

*print(angka, "adalah bilangan positif.")*

**Pernyataan if…else**

Pernyataan if…else menguji 2 kondisi. Kondisi pertama kalau benar, dan kondisi kedua kalau salah. Sintaksnya adalah seperti berikut:

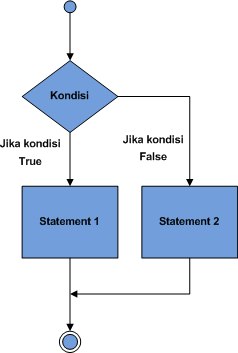
if tes kondisi:

blok pernyataan if

else:

blok pernyataan else

Diagram alir untuk pernyataan if…else adalah seperti berikut:



# Program menguji apakah sebuah bilangan positif atau negative  
# dan menampilkan pesan ke monitor  
bilangan = 5  
# coba juga mengubah bilangan menjadi bilangan = -1  
# dan perhatikan hasilnya  
if bilangan >= 0:  
 print("Positif atau Nol")  
else:  
 print("Bilangan negatif")

**Pernyataan if…elif…else…**

Pernyataan if…elif…else digunakan untuk menguji lebih dari 2 kondisi. Bila kondisi pada if benar, maka pernyataan di dalamnya yang dieksekusi. Bila salah, maka masuk ke pengujian kondisi elif. Terakhir bila tidak ada if atau elif yang benar, maka yang dijalankan adalah yang di blok else. Sintaksnya adalah seperti berikut:

if tes kondisi:

blok pernyataan if

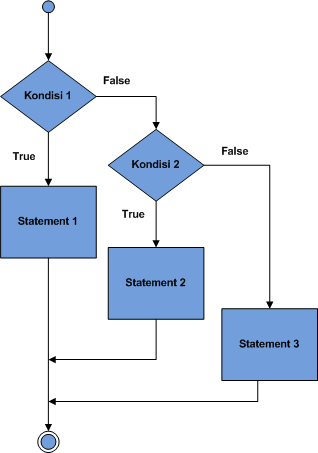
elif tes kondisi:

blok pernyataan elif

else:

blok pernyataan else

Diagram alir if…else…if adalah sebagai berikut:



# Di sini kita menguji apakah sebuah bilangan  
# adalah bilangan positif, nol, atau negative  
# dan menampilkan hasilnya ke layar  
bilangan = 5.5  
# Coba juga mengganti bilangan jadi  
# bilangan = 0  
# bilangan = -5.5  
if bilangan > 0:  
 print("Bilangan positif")  
elif bilangan == 0:  
 print("Nol")  
else:  
 print("Bilangan negatif")

Contoh if bersarang ( if didalam if)

gaji = 10000000

berkeluarga = True

punya\_rumah = True

if gaji > 3000000:

print ("Gaji sudah diatas UMR")

if berkeluarga:

print ("Wajib ikutan asuransi dan menabung untuk pensiun")

else:

print ("Tidak perlu ikutan asuransi")

if punya\_rumah:

print ("wajib bayar pajak rumah")

else:

print ("tidak wajib bayar pajak rumah")

else:

print ("Gaji belum UMR")

contoh ke dua

**huruf = raw\_input("Masukkan sebuah huruf: ")**

**if (huruf >= 'A'):**

**if (huruf <= 'Z'):**

**print "Ini adalah Huruf Besar"**

**elif (huruf >= 'a'):**

**if (huruf <= 'z'):**

**print "Ini adalah huruf kecil"**

**else:**

**print "Huruf > z"**

**else:**

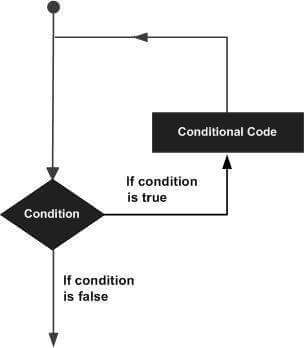
**print "Huruf > z tapi < a"**

**else:**

**print "Huruf < A"**

# Perulangan

Secara umum, Python mengeksekusi program baris perbaris. Mulai dari baris satu, dua, dan seterusnya. Ada kalanya, kita perlu mengeksekusi satu baris atau satu blok kode program beberapa kali. Hal ini disebut dengan perulangan atau biasa disebut looping atau iterasi. Untuk lebih jelasnya perhatikan gambar berikut:



Pada gambar bisa dilihat bahwa perulangan juga memerlukan tes kondisi. Bila hasil tes kondisi True, maka blok kode kembali dieksekusi. Tapi jika False, maka keluar dari perulangan

Di python, perulangan bisa dilakukan dengan dua cara atau metode, yaitu:

1. Menggunakan for
2. Menggunakan while

**Perulangan Dengan Menggunakan For**

Perulangan dengan menggunakan for memiliki sintaks seperti berikut:

for var in sequence:

body of for

var adalah variabel yang digunakan untuk penampung sementara nilai dari *sequence* pada saat terjadi perulangan. Sequence adalah tipe data berurut seperti string, list, dan tuple.

Perulangan terjadi sampai looping mencapai elemen atau anggota terakhir dari sequence. Bila loop sudah sampai ke elemen terakhir dari sequence, maka program akan keluar dari looping.

# Program untuk menemukan jumlah bilangan dalam satu list

# List number

numbers = [7, 5, 9, 8, 4, 2, 6, 4, 1]

# variablel untuk menyimpan jumlah

sum = 0

# iterasi

for each in numbers:

sum = sum + each

# Output: Jumlah semuanya: 46

print("Jumlah semuanya:", sum)

**fungsi range**

Fungsi **range()** dapat digunakan untuk menghasilkan deret bilangan. range(10) akan menghasilkan bilangan dari 0 sampai dengan 9 (10 bilangan). Kita juga bisa menentukan batas bawah, batas atas, dan interval dengan format range(batas bawah, batas atas, interval).Bila interval dikosongkan, maka nilai default 1 yang akan digunakan. Fungsi range tidak menyimpan semua nilai dalam memori secara langsung. Ia hanya akan mengingat batas bawah, batas atas, dan interval dan *membangkitkan*hasilnya satu persatu hanya bila dipanggil. Untuk membuat fungsi ini langsung menampilkan semua item, kita bisa menggunakan fungsi list(). Untuk jelasnya perhatikan contoh berikut:

# Output: range(0,10)

print(range(10))

# Output: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

print(list(range(10))

# Output: [2, 3, 4, 5, 6, 7]

print(list(range(2,8))

# Output: [2, 5, 8, 11, 14, 17]

print(list(range(2, 20, 3))

Kita bisa menggunakan fungsi range() dalam perulangan menggunakan for untuk iterasi bilangan berurut. Hal ini dengan cara mengkombinasikan fungsi range() dengan fungsi **len()**. Fungsi len() berfungsi untuk mendapatkan panjang atau jumlah elemen suatu data sekuensial atau berurut.

# Program untuk iterasi list menggunakan pengindeksan  
 mapel = ['matematika', 'fisika', 'kimia']  
# iterasi list menggunakan indeks  
 for i in range(len(mapel)):  
 print("Saya suka", mapel[i])

**Perulangan Menggunakan while**

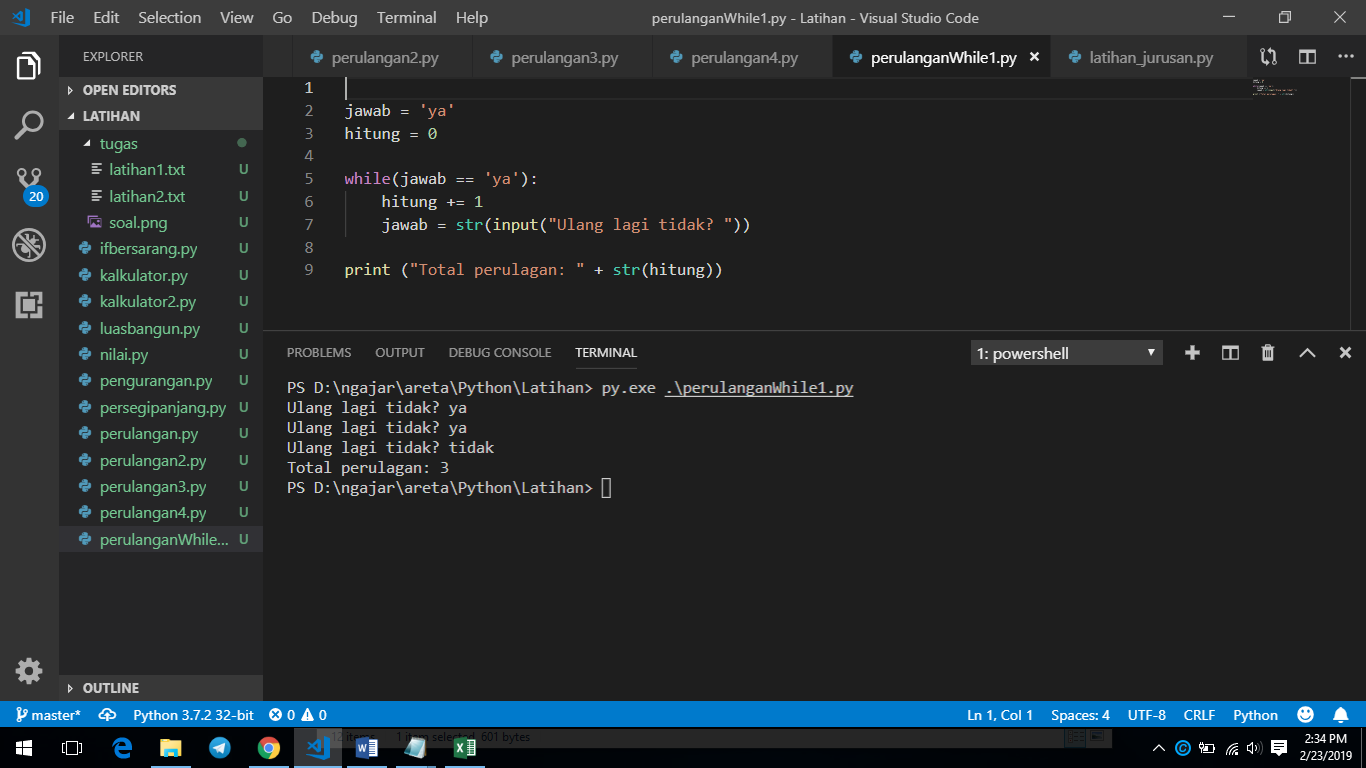
Perulangan menggunakan while akan menjalankan blok pernyataan terus menerus selama kondisi bernilai benar.

Adapun sintaks dari perulangan menggunakan while adalah:

while expression:

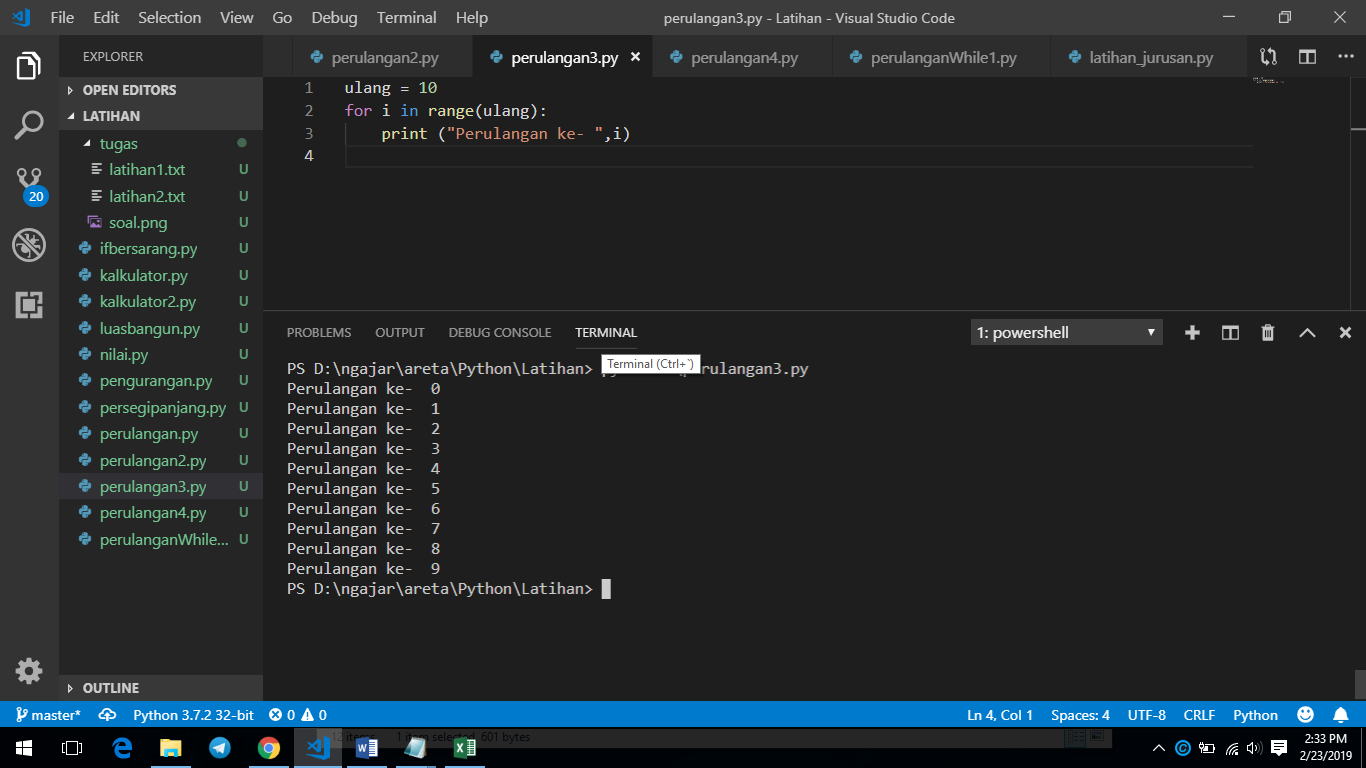
statement (s)

Contoh 1



contoh 2

**Perualngan Denga For**



**Contoh kedua**

