**Pengertian Ekspresi**

Salah satunya adalah materi ekspresi kali ini. Data yang Anda simpan pada suatu variabel umumnya akan dioperasikan untuk menghasilkan suatu nilai sesuai keinginan. Masih ingat ekspresi pada matematika? Ekspresi pada matematika adalah kombinasi dari simbol-simbol matematika, seperti angka, variabel, operasi matematika, dan sebagainya. Contohnya seperti pada gambar berikut.



Pada gambar di atas, "4x-7" merupakan ekspresi, sedangkan "4x", "7", dan "5" merupakan suku bilangan. Apabila kita mengingat kembali, operasi pada matematika ataupun pemrograman merupakan suatu proses yang dilakukan untuk mendapatkan hasil nilai tertentu.

Lalu, apa itu ekspresi dalam pemrograman? Ekspresi pada pemrograman merupakan kombinasi dari satu atau lebih variabel, konstanta, operator, dan fungsi yang bermakna untuk menghasilkan suatu nilai dalam suatu tipe tertentu.

Struktur umum ekspresi yang sering dijumpai adalah <operan1> <operator> <operan2>. Namun, perlu Anda ketahui bahwa struktur umum tersebut merupakan struktur ekspresi biner (jenis ekspresi menggunakan dua operan). Mari bedah struktur tersebut lebih dalam.

1. Operan dapat berupa nilai, variabel, konstanta, atau ekspresi lain.
2. Operator merupakan suatu fungsi standar yang disediakan dalam bahasa pemrograman untuk melakukan beberapa hal dasar, seperti perhitungan aritmetika, logika, dan relasional. Contohnya adalah penjumlahan (+), pengurangan (-), perkalian (\*), modulus (%), dan sebagainya.

Jenis-Jenis Ekspresi

Pada dasarnya, jenis-jenis ekspresi dibagi menjadi dua. Pertama adalah menurut jumlah operan (arity) dari operator dan kedua adalah menurut tipe data yang dihasilkan.

**Ekspresi Menurut Arity dari Operator**

Biner : (x + y), (x - y), (x \* y), (x / y), (x \*\* y), (x < y), (x <= y), (x > y), (x >= y), (x % y), (x == y), (x != y).

Uner : (x += 1), (x-=1), (not x).

Ekspresi biner merupakan jenis yang memiliki dua operan. Operatornya meliputi penjumlahan (+), pengurangan (-), perkalian (\*), pembagian (/), perpangkatan (\*\*), lebih kecil dari (<), lebih kecil dari sama dengan (<=), lebih besar dari (>), lebih besar dari sama dengan (>=), modulus (%), sama dengan (==), dan tidak sama dengan (!=).

Namun, ekspresi uner adalah jenis ekspresi yang memiliki bentuk dasar operasi dengan satu operan. Contohnya adalah *increment* (x+=1), *decrement* (x-=1), dan negasi (not x).

Anda sudah melihat penerapan ekspresi biner pada submodul sebelumnya dan berikut adalah penerapan ekspresi uner.

Mari kita bedah satu persatu kode di atas.

1. Variabel a bernilai True, jika kita memberikan ekspresi "not a" yang artinya "not True", hasil yang didapat adalah False.
2. Baik *increment*maupun *decrement*, keduanya adalah pola yang sama untuk menambahkan dan mengurangi suatu variabel dengan jumlah tetap.
   1. a += 1 memiliki tujuan yang sama dengan struktur a = a + 1. Jika diasumsikan variabel a bernilai 1, seolah-olah kita melakukan operasi penjumlahan "1+1". Inilah alasan ia disebut dengan *increment* yang artinya kenaikan.
   2. *Decrement* dapat dijelaskan sebagai a -=1, memiliki tujuan yang sama dengan struktur a = a - 1. Jika diasumsikan variabel a bernilai 1, seolah-olah kita melakukan operasi pengurangan "1-1".

Penjelasan lebih detail mengenai operator akan kita bahas pada materi selanjutnya. Saat ini mari kita lanjut membahas ekspresi menurut tipe data yang dihasilkan.

**Ekspresi Menurut Tipe Data yang Dihasilkan**

Sebagaimana judulnya, jenis-jenis ini adalah ekspresi yang dikelompokkan berdasarkan tipe data yang dihasilkan. Mari kita bedah satu per satu.

1. Ekspresi aritmetika: jenis ekspresi yang memiliki **operan bertipe numerik**dan menghasilkan **numerik**.
2. Ekspresi relasional: jenis ekspresi yang memiliki **operan bertipe numerik**dan menghasilkan nilai **boolean/logika**.
3. Ekspresi logika: jenis ekspresi yang memiliki **operan bertipe boolean/logika** dan menghasilkan nilai **boolean/logika**.

print(2+2)

print(3<10)

print(True or False)

Output:

4

True

True

Pada kode di atas, kita melakukan operasi dengan melibatkan ekspresi aritmetika, ekspresi relasional, dan ekspresi logika. Pada ekspresi aritmetika, "2+2" menghasilkan nilai "4". Pada ekspresi relasional, "3<10" menghasilkan nilai True; tentu kita tahu bahwa tiga termasuk bilangan yang kurang dari sepuluh. Pada ekspresi logika "True or False" menghasilkan nilai True, ini merupakan gerbang logika pada dunia pemrograman. Anda akan mempelajari lebih detail pada materi operator logika.

**Jenis-Jenis Operator**

Selain ekspresi dengan beragam jenis, operator pun memiliki berbagai jenis yang dikelompokkan menjadi operator aritmetika, operator relational, operator logika, dan operator *assignment*.

**Operator Aritmetika**

Jenis pertama adalah operator untuk melakukan operasi aritmetika. Perhatikan tabel di bawah ini untuk memahami contoh penerapan operator aritmetika. **Asumsikan x bernilai 11 dan y bernilai 5.**

* Penjumlahan (+) Menambahkan nilai dari kedua operan.
* Pengurangan (-) Mengurangi nilai dari kedua operan.
* Perkalian (\*) Mengalikan nilai dari kedua operan.
* Pembagian Bulat (//) Membagi nilai dari kedua operan. Jika operan adalah integer, hasil operasi adalah bilangan bulat.
* Pembagian Riil (/) Membagi nilai dari kedua operan. Jika operan adalah float, hasil operasi adalah bilangan riil.
* Modulo (%) Sisa hasil pembagian nilai dari dua operan.
* Pangkat (\*\*) Memangkatkan nilai dari dua operan.

Semua operator aritmetika di atas adalah jenis untuk melakukan operasi aritmetika yang sering dijumpai. Mari lihat penerapannya pada Python.

**Operator Relasional**

Operator relasional merupakan operator perbandingan antara dua operan yang berupa integer, float, string, ataupun boolean. Hasil akhir dari operator ini adalah nilai bertipe boolean. Perhatikan tabel di bawah untuk memahami contoh penerapan operator relasional. **Asumsikan kedua variabel bertipe numerik atau float dengan x bernilai 5 dan y bernilai 10.**

* Sama dengan (==) Menghasilkan True, jika kedua operan bernilai sama.
* Tidak Sama dengan (!=) Menghasilkan True, jika kedua operan tidak bernilai sama.
* Lebih Besar dari (>) Menghasilkan True, jika operan kiri lebih besar dari operan kanan.
* Kurang dari (<) Menghasilkan True, jika operan kanan lebih besar dari operan kiri.
* Lebih Besar dari Sama dengan (>=) Menghasilkan True, jika operan kiri lebih besar atau sama dengan operan kanan.
* Kurang dari Sama dengan (<=) Menghasilkan True, jika operan kanan lebih besar atau sama dengan operan kiri.

Pada dasarnya, ini adalah operator yang membandingkan dua nilai dengan hasil akhir bertipe boolean. Mari lihat penerapannya pada Python.

Pada kode di atas, Anda telah menerapkan seluruh operasi menggunakan operator relasional. Jika kita lihat lebih detail, seluruh output tersebut memiliki hasil yang sama dengan tabel sebelumnya. Sebagaimana telah dijelaskan sebelumnya, operator relasional dapat menggunakan operan bertipe integer, string, ataupun float. Kode di atas telah menggunakan operan bertipe integer, Anda juga bisa mengubahnya dengan operan bertipe float.

Namun, berbeda halnya dengan operan bertipe string. Anda dapat melihat tabel di bawah untuk contoh penerapannya. **Asumsikan x bernilai "Dicoding" dan y bernilai "Indonesia".**

* Sama dengan (==) Menghasilkan True, jika kedua string memiliki nilai yang identik/sama persis.
* Tidak Sama dengan (!=) Menghasilkan True, jika kedua string memiliki nilai yang tidak sama.
* Lebih Besar dari (>) Menghasilkan True, jika **huruf pertama pada string pertama**memiliki nilai unicode dengan urutan yang lebih **tinggi** dibandingkan **huruf pertama pada string kedua.**
* Kurang dari (<) Menghasilkan True, jika **huruf pertama pada string pertama**memiliki nilai unicode dengan urutan yang lebih **rendah** dibandingkan **huruf pertama pada string kedua.**
* Lebih Besar dari Sama dengan (>=) Menghasilkan True, jika **huruf pertama pada string pertama**memiliki nilai unicode dengan urutan yang lebih **tinggi** atau sama dibandingkan **huruf pertama pada string kedua.**
* Kurang dari Sama dengan (<=) Menghasilkan True, jika **huruf pertama pada string pertama**memiliki nilai unicode dengan urutan yang lebih **rendah** atau sama dibandingkan **huruf pertama pada string kedua.**

**Catatan**: Nilai [unicode](https://docs.python.org/3/howto/unicode.html) adalah standar internasional yang menetapkan kode numerik untuk setiap karakter dari hampir semua sistem tulisan dan simbol-simbol yang digunakan oleh manusia.

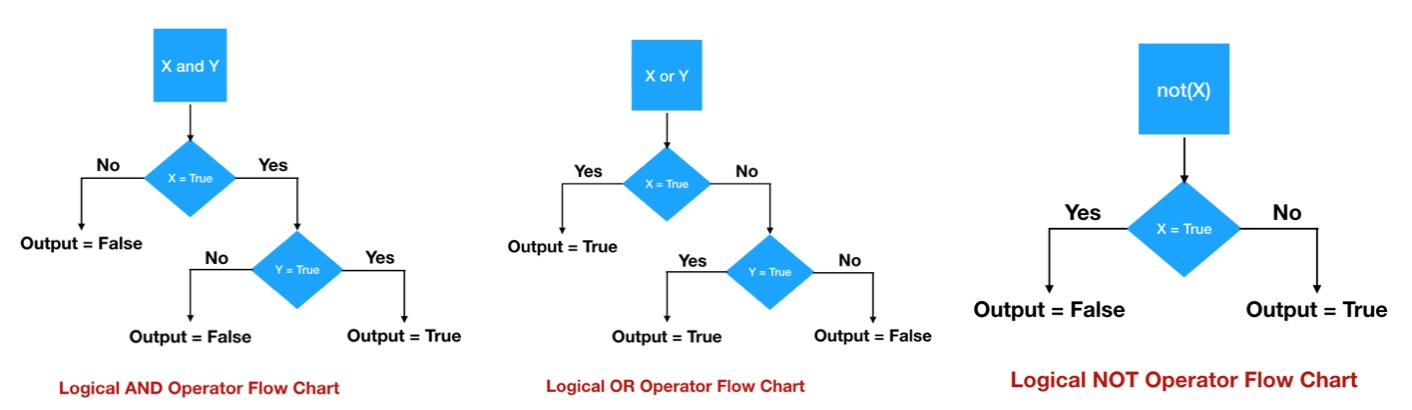
Perhatikan kode di atas, string "Dicoding" dan "Indonesia" tidak sama sehingga sintaks operator sama dengan "==" menghasilkan nilai False, sedangkan operator tidak sama dengan "!=" menghasilkan nilai True. Operator sisanya akan membandingkan huruf D pada string "Dicoding" dan huruf I pada string "Indonesia". Anda bisa mencoba mengganti huruf pertama tersebut untuk memahami lebih detail.

**Operator Logika**

Operator logika merupakan jenis operator untuk melakukan operasi logika dengan kedua operannya bertipe boolean. Hasil akhir dari operasi ini adalah nilai bertipe boolean. Perhatikan kode di bawah ini untuk memahami contoh penerapannya, **asumsikan bahwa p bernilai True dan q bernilai False**.

* "AND" atau "&" Logika yang hanya menghasilkan True jika kedua operan bernilai True.
* "OR" atau "|" Logika yang menghasilkan True jika salah satu dari kedua operan bernilai True.
* NOT Logika yang bertujuan untuk membalikkan nilai logika dari operannya.

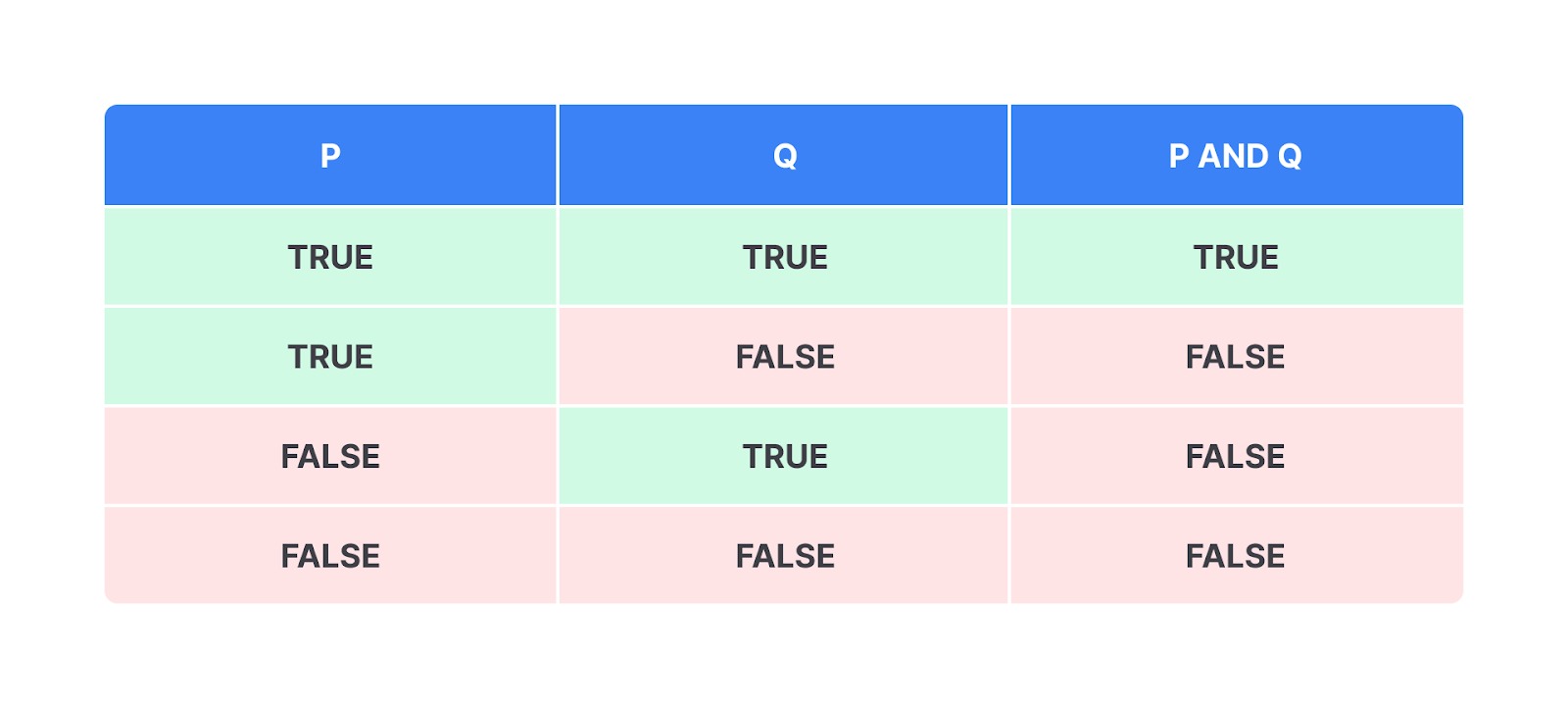
Jenis operator ini membandingkan dua operan bertipe boolean. Perhatikan gambar ini untuk memahaminya lebih dalam.



Operator AND hanya akan menghasilkan nilai True jika kedua operannya bernilai True. Operator OR akan menghasilkan nilai True jika salah satunya bernilai True. Operator NOT hanya akan membalikkan nilai logika; jika True, False yang akan dikembalikan, begitupun sebaliknya.

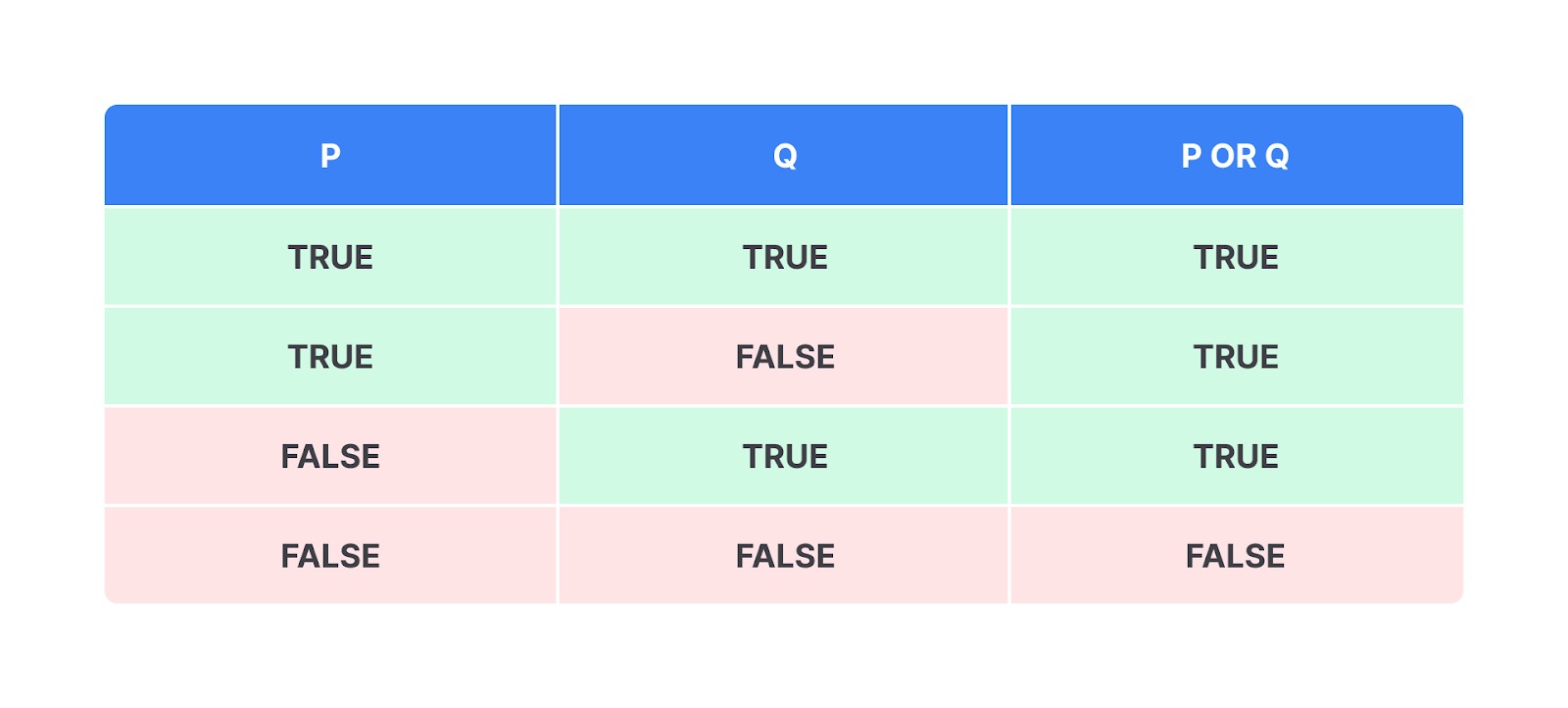
Mari kita bedah satu per satu jenis operator di atas.

**Operator AND**



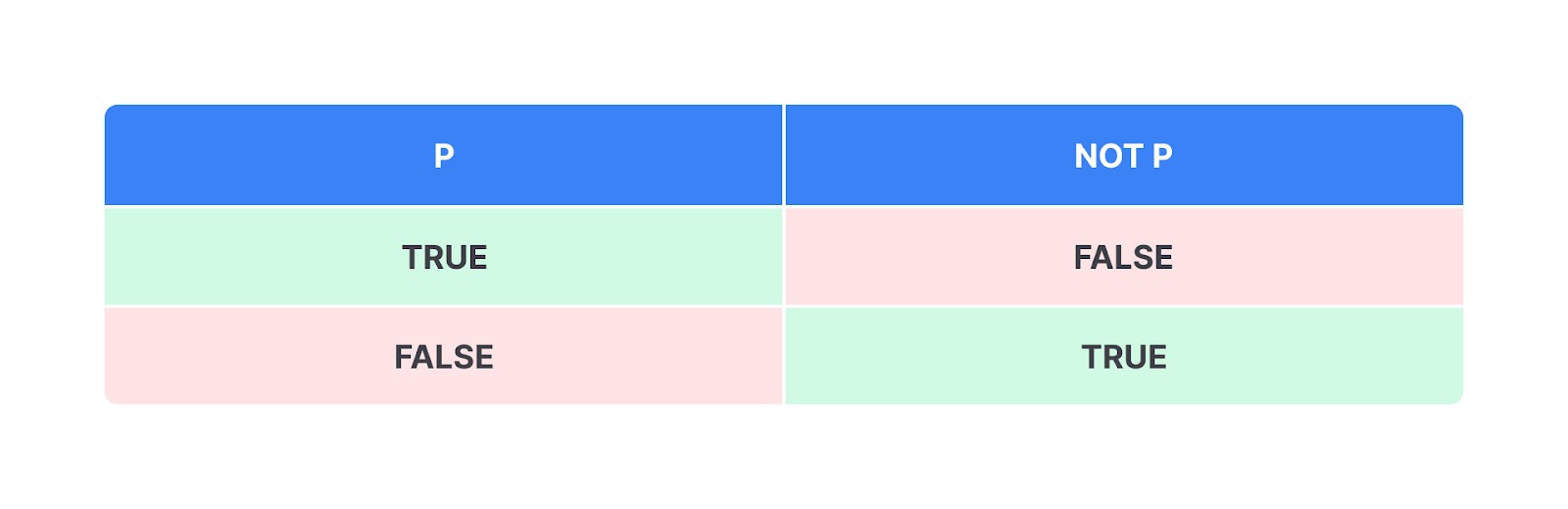
Pada tabel di atas, operator AND hanya akan menghasilkan nilai TRUE jika kedua operan, yakni P dan Q bernilai TRUE. Sisanya akan menghasilkan nilai FALSE. Mari lihat penerapannya pada Python.

**Operator OR**



Pada tabel di atas, operator OR akan menghasilkan nilai TRUE jika salah satu atau kedua operan (yakni P atau Q) bernilai True. Mari lihat penerapannya pada Python.

**Operator NOT**



Pada tabel di atas, operator NOT akan membalikkan nilai boolean dari operan aslinya atau disebut sebagai negasi. Mari lihat penerapannya pada Python.

**Operator Assignment**

Operator selanjutnya adalah assignment. Operator ini bertujuan untuk melakukan proses assignment atau pemberian nilai pada suatu variabel dengan nilai tetap. Perhatikan tabel di bawah ini untuk memahami contoh penerapan operator assignment. **Asumsikan x bernilai 11 dan y bernilai 5.**

* += Menyederhanakan operasi x = x + y.
* -= Menyederhanakan operasi x = x - y.
* \*= Menyederhanakan operasi x = x \* y.
* /= Menyederhanakan operasi x = x / y.
* %= Menyederhanakan operasi x = x % y.

Sederhananya, seluruh operator di atas setara dengan x = x <operator> y. Biasanya, Anda akan sering menjumpai operator assignment ini pada perulangan (Anda akan mempelajarinya nanti). Perulangan pada pemrograman berarti Anda membuat sebuah program yang akan terus berulang hingga berakhir karena suatu kondisi. Salah satu caranya agar dapat terus berulang adalah dengan menambahkan operator assignment.