COMPUTER
PROGRAMING 2

KONVERSI
TYPE DATA PYTHON

MODUL 4

2023

DAFTAR ISI

JUDUL

Pada pembahasan awal tentang tipe data dan variabel yang telah berlalu, kita mengetahui bahwa ada beberapa tipe data pada python. Baik tipe data yang bersifat asli **(primitif)** mau pun tipe data yang tergolong canggih karena ia merupakan susunan dari tipe data asli lainnya.

Dari berbagai macam tipe data tersebut, kita bisa langsung membuatnya tanpa perlu repot-repot mendefinsikan terlebih dahulu tipe datanya apa.

Misal:

```
nama = 'Lendis Fabri'
```

Variabel nama dalam kode program di atas, otomatis memiliki tipe data string ketika kita mengisinya dengan data string, dan otomatis akan menjadi tipe data integer jika kita mengisinya dengan tipe data integer, dan begitu seterusnya. Konsep pendekatan seperti ini dinamakan weak typing.

A. Memeriksa Jenis Tipe Data

Selain itu, kita bisa memeriksa tipe data dari suatu variabel dengan fungsi type() bawaan python.

Contoh:

```
nama = 'Budi'
usia = 20
lajang = True

print(type(nama))
print(type(usia))
print(type(lajang))
```

Output:

```
<class 'str'>
<class 'int'>
<class 'bool'>
```

B. Apa itu Konversi Tipe Data?

Konversi tipe data adalah teknik mengubah nilai yang awalnya dari tipe data a, menjadi tipe data b. Terdapat 2 cara dalam mengkonversi tipe data:

- Konversi secara implisit (otomatis)
- Konversi secara eksplisit (manual)

1. Alasan Kenapa Data Butuh Dikonversi?

Jawabannya adalah:

Tidak semua data itu valid dan tidak semua data itu bisa kita oleh sesuai kebutuhan.

Yang pertama, tidak semua data itu valid. Apa maksudnya?

Contoh, kita meminta user untuk menginputkan data usia. User menuliskan angka usianya. Akan tetapi, python justru menganggap data tersebut sebagai string, bukan integer.

Perhatikan contoh berikut:

```
usia = input('Masukkan usia anda: ')

print(type(usia))
```

Jika kita masukkan 20, maka outputnya akan seperti ini:

```
Masukkan usia anda: 20
<class 'str'>
```

Nah? Sedangkan jika kita berusaha mengoperasikan (misal perkalian) string dan integer, atau string dan string, maka (secara umum) kita akan mendapatkan error.

Misal:

```
panjang = input('Masukkan panjang: ')
lebar = input('Masukkan lebar: ')

print('Luas =', panjang * lebar)
```

Error yang kita dapat:

```
Masukkan panjang: 10
Masukkan lebar: 5

Exception has occurred: TypeError
can't multiply sequence by non-int of type 'str'
```

Karena tidak semua data itu tipe datanya valid, yang mana mengakibatkan tidak bisa diolahnya data tersebut, oleh karena itu mengkonversi tipe data sesuai kebutuhan merupakan sesuatu yang penting.

2. Konversi Tipe Data Secara Implisit

Ada dua model konversi tipe data. Yang pertama adalah model implisit dan yang kedua adalah eksplisit.

Model implisit adalah proses pengkonversian tipe data yang terjadi secara otomatis dibalik layar, tanpa perlu kita instruksi secara langsung.

Misal, hasil dari operasi pembagian antar dua bilangan akan otomatis dikonversi menjadi tipe data float.

Perhatikan contoh berikut:

```
a = 10 / 2
print(type(a))

b = 1 / 2
print(type(b))
```

Output:

```
<class 'float'>
<class 'float'>
```

Hal ini juga berlaku untuk penjumlahan atau perkalian atau pengurangan jika salah satu operan-nya adalah bilangan pecahan, seperti contoh berikut:

```
c = 10 + 5
print(type(c))
```

```
d = 10 + 5.0
print(type(d))
```

```
e = 10.5 - 3
print(type(e))
```

Output:

```
<class 'int'>
<class 'float'>
<class 'float'>
```

Perkalian String

Demikian pula dengan perkalian antara string dan integer, ia akan menghasilkan sebuah string:

```
kata = 'Jeruk ' * 5

print(kata)
print(type(kata))
```

Output:

```
Jeruk Jeruk Jeruk Jeruk Jeruk
<class 'str'>
```

3. Konversi Tipe Data Secara Eksplisit

Kita tahu bahwa tidak semua operasi dari tipe data yang berbeda bisa dikonversi dengan benar oleh python. Beberapa di antaranya justru menghasilkan error jika tidak kita konversi manual. Proses konversi manual ini lah yang dinamakan sebagai tipe data casting secara eksplisit.

Dalam python, kita bisa mengkonversi tipe data secara eksplisit dengan memanggil fungsi konstruktor dari masing-masing tipe data.

Untuk tipe data primitif, kita bisa memanggil fungsi berikut:

- int() - untuk konversi data ke integer
- float() - untuk konversi data ke float
- bool() - untuk konversi data ke bool
- str() - untuk konversi data string (sebagian orang menyebutkan bahwa string adalah tipe data primitif pada python)

Dan untuk tipe data lain yang tergolong tipe data canggih (atau buatan), kita juga bisa memanggil masing-masing konstruktornya untuk melakukan tipe data casting, semisal:

- list() - untuk list
- set() - untuk set
- tuple() - untuk tuple

4. Konversi String ke Numerik

Mari kita mencoba mengkonversi string ke numerik dengan memperbaiki kode program di bawah ini:

```
panjang = input('Masukkan panjang: ')
lebar = input('Masukkan lebar: ')

print('Luas =', panjang * lebar)
```

Di awal, kita telah mencoba mengeksekusi program di atas akan tetapi interpreter mengembalikan sebuah error. Itu terjadi karena kita tidak bisa mengalikan string dengan string secara langsung.

Yang harus kita lakukan adalah mengkonversinya terlebih dahulu, entah ke integer menggunakan fungsi int(), atau ke float menggunakan fungsi float().

Hasil akhirnya:

```
panjang = int(input('Masukkan panjang: '))
lebar = float(input('Masukkan lebar: '))

print('Luas =', panjang * lebar)
```

Contoh output:

```
Masukkan panjang: 10
Masukkan lebar: 5
Luas = 50.0
```

Kita juga bisa mengkonversi sebuah float ke integer atau sebaliknya:

```
print(int(5.6))
print(float(5))
```

Output:

```
5
5.0
```

5. Konversi Numerik ke String

Sebaliknya, pada saat-saat tertentu kita justru butuh untuk mengkonversi data numerik ke dalam string. Kita bisa dengan mudah menggunakan fungsi konstruktor str().

Perhatikan contoh berikut:

```
nama = 'Wudi'
tahun_lahir = 2000

print(nama + ' lahir di tahun ' + str(tahun_lahir))
```

Output:

```
Wudi lahir di tahun 2000
```

Kenapa harus menggunakan **str()?**

Karena proses penggabungan string hanya bisa dilakukan oleh 2 buah string, jika salah satunya numeric (integer / float) maka harus dikonversi terlebih dahulu.

NB: cara yang disarankan untuk menyisipkan variabel ke dalam string adalah menggunakan f-string:

```
print(f'{nama} lahir di tahun {tahun_lahir}')
```

6. Konversi ke Boolean

Kita bisa melakukan konversi ke boolean dengan menggunakan fungsi **bool().**

Data yang bernilai “putih” atau kosong akan dianggap dan dikonversi sebagai False, sebaliknya data yang bernilai “hitam” atau tidak kosong akan dianggap dan dikonversi menjadi True.

Di antara data yang dianggap False:

- None- kosong
- 0 - integer kosong
- 0.0 - float kosong
- "" - string kosong
- [] - list kosong
- () - tuple kosong
- {} - dictionary kosong

Selain dari itu, data-data yang tidak “putih” akan dianggap sebagai True.

Untuk lebih memperjelas lagi, silakan tulis dan jalankan kode program di bawah:

```
print(type(None), '->', bool(None))
print(type(0), '->', bool(0))
print(type(0.0), '->', bool(0.0))
print(type(""), '->', bool(""))
print(type([]), '->', bool([]))
print(type(()), '->', bool(()))
print(type({}), '->', bool({}))

print("-" * 20)

print(type(5), '->', bool(5))
print(type(-14.5), '->', bool(-14.5))
print(type("Aku"), '->', bool("Aku"))
print(type([1, 2, 3]), '->', bool([1, 2, 3]))
print(type(("a", "b", False)), '->', bool(("a", "b",
False)))
print(type({ 'nama': 'Lendis Fabri' }), '->', bool({ 'nama':
'Lendis Fabri' }))
```

Output dari kode program di atas adalah:

```
<class 'NoneType'> -> False
<class 'int'> -> False
<class 'float'> -> False
<class 'str'> -> False
<class 'list'> -> False
<class 'tuple'> -> False
<class 'dict'> -> False
--------------------
<class 'int'> -> True
<class 'float'> -> True
<class 'str'> -> True
<class 'list'> -> True
<class 'tuple'> -> True
<class 'dict'> -> True
```

7. Konversi Dari dan Ke List, Set dan Tuple

Untuk tipe data berjenis koleksi, kita juga bisa melakukan konversi satu sama lain dengan fungsi konstruktor dari masing-masing tipe data:

- fungsi list() untuk list
- fungsi set() untuk set
- fungsi tuple() untuk tuple

Perhatikan contoh-contoh berikut:

```
# list ke tuple dan ke set
listHuruf = ['a', 'b', 'c', 'c']

print(listHuruf)
print(tuple(listHuruf))
print(set(listHuruf))
```

Output:

```
['a', 'b', 'c', 'c']
('a', 'b', 'c', 'c')
{'b', 'c', 'a'}
```

```
# tuple ke list dan ke set
tplBuah = ('Mangga', 'Jambu')

print(tplBuah)
print(list(tplBuah))
print(set(tplBuah))
```

Output:

```
('Mangga', 'Jambu')
['Mangga', 'Jambu']
{'Mangga', 'Jambu'}
```

```
# set ke list dan ke tuple
setAngka = {1, 3, 5, 5}

print(setAngka)
print(list(setAngka))
print(tuple(setAngka))
```

Output:

```
{1, 3, 5}
[1, 3, 5]
(1, 3, 5)
```

C. Kesimpulan

Setelah melakukan berbagai macam percobaan, kita bisa simpulkan pertemuan kali ini menjadi beberapa poin penting. Di antaranya:

- Python melakukan deteksi otomatis terhadap tipe data suatu variabel.
- Selain itu, python juga melakukan konversi tipe data secara otomatis ketika melakukan operasi tertentu dari dua tipe data yang berbeda. Konversi ini disebut sebagai konversi implisit.
- Beberapa operasi tidak bisa dilakukan untuk beberapa tipe data yang berbeda, oleh karena perlu adanya konversi tipe data secara manual. Dan cara ini disebut sebagai konversi eksplisit.
- Kita bisa mengkonversi berbagai macam tipe data secara eksplisit, mulai dari tipe data yang asli mau pun tipe data kolektif.

=== To be Continue ===