Dictionary

In [ ]:

```
mahasiswa = {
    "nama": "Indah Puspitasari",
    "kelas": "TI 22H",
    "jurusan": "Teknik Informatika"
}
```

Pengertian Dictionary

Dictionary adalah tipe data pada python yang berfungsi untuk menyimpan kumpulan data/nilai dengan pendekatan “key-value”.

Dictionary sendiri memiliki dua buah komponen inti:

1. Yang pertama adalah `key` , ia merupakan nama atribut suatu item pada dictionary.
2. Yang kedua adalah `value` , ia adalah nilai yang disimpan pada suatu atribut.

Pada contoh diatas, variabel `mahasiswa` adalah sebuah `Dictionary` yang memiliki `key` . Setiap key diisi dengan sebuah nilai atau value. Berikut adalah kumpulan `key` diikuti dengan nilainya dari variabel `mahasiswa` :

1. `nama` , nilainya adalah `Indah Puspitasari`
2. `kelas` , nilainya adalah `TI22H`
3. `jurusan` , nilainya adalah `Teknik Informatika`

Sifat Dictionary

Dictionary items memiliki 3 sifat, yaitu:

1. `Unordered` - tidak berurutan
2. `Changeable` - bisa diubah
3. `Unique` - alias tidak bisa menerima dua keys yang sama

`Unordered` artinya ia tidak berurutan, sehingga key/atribut yang pertama kali kita definisikan, tidak berarti dia akan benar-benar menjadi yang “pertama” dibandingkan dengan key yang lainnya. Juga, unordered berarti tidak bisa diakses menggunakan index (integer) sebagaimana halnya list.

Sedangkan `changeable` artinya kita bisa kita mengubah value yang telah kita masukkan ke dalam sebuah dictionary. Hal ini berbeda dengan tipe data set mau pun tuple yang mana keduanya bersifat immutable alias tidak bisa diubah.

Dan yang terakhir, dictionary tidak bisa memiliki lebih dari satu key yang sama karena ia bersifat `unique` . Sehingga jika ada dua buah key yang sama, key yang didefinisikan terakhir akan menimpa nilai dari key yang didefinisikan lebih awal.

Perhatikan contoh berikut:

In [5]:

```
artikel = {
  "judul": "Menu Masakan Enak",
  "judul": "Menu Masakan Enak Tradisional"
}

print(artikel.get("judul"))
```

```
Menu Masakan Enak Tradisional
```

Kode program di atas akan menghasilkan output:

```
Menu Masakan Enak Tradisional
```

Kita bisa perhatikan bahwa pada dictionary `artikel` di atas, terdapat dua buah item dengan nama `“judul”` , akan tetapi ketika kita tampilkan hasilnya hanya nilai yang terakhir kita masukkan.

Selain 3 sifat yang telah disebutkan, item pada dictionary juga bisa menerima berbagai macam tipe data, mulai dari tipe data asli mau pun tipe data terusan seperti objek.

Yang artinya, dictionary juga bisa memiliki value berupa dictionary juga.

Contoh:

In [6]:

```
mahasiswa = {
    "nama": "Supardi G",
    "alamat": {
        "kota": "Cianjur",
        "provinsi": "Jawa Barat"
    }
}
```

Cara Membuat Dictionary

Berikutnya adalah bagaimana cara membuat dictionary pada python.

Untuk membuatnya, terdapat 2 cara:

1. Yang pertama dengan tanda kurung kurawal {}.
2. Dan yang kedua bisa menggunakan fungsi atau konstruktor dict().

Perhatikan contoh berikut:

In [ ]:

```
# cara pertama
buku = {
  "judul": "Daun Yang Jatuh Tidak Pernah Membenci Angin",
  "penulis": "Tere Liye"
}

# cara kedua
buku = dict(
  judul="Daun Yang Jatuh Tidak Pernah Membenci Angin",
  penulis="Tere Liye"
)
```

Cara Mengakses Item Pada Dictionary

Kita bisa mengakses item pada dictionary dengan dua cara:

- dengan menggunakan kurung siku ([])
- atau dengan menggunakan fungsi get()

Perhatikan contoh berikut:

In [7]:

```
pertemuan_hari_ini = {
  "judul": "Belajar Dictionary Pada Python 3",
  "tanggal": "01 Februari 2021",
  "kategori": [
    "Python", "Python Dasar"
  ],
  "page_views": 10,
  "published": True,
  "share_count": {
    "facebook": 0,
    "twitter": 2
  }
}

# Kita bisa mengaksesnya dengan:

print('Judul:', pertemuan_hari_ini.get('judul'))
# atau
print('Tanggal:', pertemuan_hari_ini['tanggal'])

# bisa menggunakan fungsi berantai untuk dictionary bertingkat
print('Facebook share:', pertemuan_hari_ini.get('share_count').get('facebook'))
# bisa juga dengan kurung siku dua-duanya
print('Twitter share:', pertemuan_hari_ini['share_count']['twitter'])
```

```
Judul: Belajar Dictionary Pada Python 3
Tanggal: 01 Februari 2021
Facebook share: 0
Twitter share: 2
```

Hanya saja, di antara keunggulan menggunakan fungsi `get()` adalah:

> Kita bisa mengatur nilai default jika key pada dictionary yang kita panggil tidak ditemukan. Hal itu akan mencegah sis tem untuk menampilkan error.

Perhatikan contoh berikut, perintah yang pertama akan menimbulkan error\ sedangkan perintah yang kedua tidak:

In [9]:

```
share_count = pertemuan_hari_ini.get('share_count')

# ini error
print('Instagram share:', share_count['instagram'])
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Input In [9], in <cell line: 4>()
      1 share_count = pertemuan_hari_ini.get('share_count')
      3 # ini error
----> 4 print('Instagram share:', share_count['instagram'])

KeyError: 'instagram'
```

In [10]:

```
share_count = pertemuan_hari_ini.get('share_count')

# sedangkan ini tidak error
print('Instagram share:', share_count.get('instagram', 0))
```

```
Instagram share: 0
```

Perulangan Untuk Dictionary

Kita juga bisa menampilkan semua isi dari sebuah dictionary dengan memanfaatkan perulangan.

Perhatikan contoh berikut:

In [11]:

```
buku = {
  'judul': 'Hafalan Sholat Delisa',
  'penulis': 'Tere Liye'
}

for key in buku:
  print(key, '->', buku[key])
```

```
judul -> Hafalan Sholat Delisa
penulis -> Tere Liye
```

Atau kita juga bisa memanggil fungsi dictionary.items() untuk perulangan yang lebih simpel:

In [12]:

```
buku = {
  'judul': 'Hafalan Sholat Delisa',
  'penulis': 'Tere Liye'
}

for nama_atribut, nilai in buku.items():
  print(nama_atribut, '->', nilai)
```

```
judul -> Hafalan Sholat Delisa
penulis -> Tere Liye
```

```
## Mengubah Nilai Item

Seperti yang telah kita singgung di atas bahwasanya item pada dictionary bersifat changeable alias bisa diubah.

Untuk mengubah nilai item pada suatu dictionary, caranya simpel seperti mengubah variabel pada umumnya.

Contoh:
```

In [16]:

```
mahasiswa = {
  'nama': 'Lendis Fabri',
  'asal': 'Indonesia'
}

# mengubah data
print('Nama awal:', mahasiswa.get('nama'))
mahasiswa['nama'] = 'Andi Mukhlish'
print('Setelah diubah:', mahasiswa.get('nama'))
```

```
Nama awal: Lendis Fabri
Setelah diubah: Andi Mukhlish
```

Menambahkan item

Untuk menambahkan key dan item baru, caranya seperti mengedit item.

Jadi:

1. Kalau key yang kita definisikan `ternyata sudah ada` , sistem akan `me-replace item` yang lama dengan yang baru, alias meng-edit
2. Tapi jika key yang kita definisikan `ternyata tidak ada` , maka sistem akan `menambahkannya` sebagai item baru.

Contoh:

In [17]:

```
mahasiswa = {
  'nama': 'Lendis Fabri',
  'asal': 'Indonesia',
}

# output None
print('Hobi:', mahasiswa.get('hobi'))
# tambah data
mahasiswa['hobi'] = 'Memancing'
# print ulang
print('Hobi dari {} adalah {}'.format(
  mahasiswa.get('nama'),
  mahasiswa.get('hobi')
))
```

```
Hobi: None
Hobi dari Lendis Fabri adalah Memancing
```

Menghapus Item

```
Untuk menghapus item, terdapat dua cara:

1. Menggunakan statement del <dict[key]>.
2. Atau menggunakan fungsi dictionary.pop()
```

Perhatikan contoh berikut:

In [21]:

```
mahasiswa = {
  'nama': 'Wahid Abdullah',
  'usia': 18,
  'asal': 'Indonesia'
}

del mahasiswa['nama']
mahasiswa.pop('usia')
mahasiswa.pop('asal')
```

Out[21]:

```
'Indonesia'
```

Apa bedanya memakai cara pertama dan cara kedua? Bedanya, kalau menggunakan fungsi `pop()` , kita bisa mendapatkan nilai kembalian dari data yang telah dihapus.

Contoh:

In [22]:

```
pesan_singkat = {
  "isi": "ISI PESAN INI HANYA BISA DIBACA SEKALI SAJA!! 😱"
}

isi_pesan = pesan_singkat.pop('isi')

# akses langsung dari dictionary
# ouput: None
print('isi pesan:', pesan_singkat.get('isi'))

# akses dari hasil kembalian yang telah disimpan
print('isi pesan:', isi_pesan)
```

```
isi pesan: None
isi pesan: ISI PESAN INI HANYA BISA DIBACA SEKALI SAJA!! 😱
```

Operator Keanggotaan

Kita bisa memanfaatkan operator keanggotaan untuk tipe data dictionary pada python.

Perhatikan contoh berikut:

In [24]:

```
siswa = {
  'nama': 'Renza Ilhami'
}

print('Apakah variabel siswa memiliki key nama?')
print('nama' in siswa)

print('\nApakah variabel siswa TIDAK memiliki key usia?')
print('usia' not in siswa)
```

```
Apakah variabel siswa memiliki key nama?
True

Apakah variabel siswa TIDAK memiliki key usia?
True
```

Panjang atau Banyak Key Pada Dictionary

Terakhir tapi bukan yang paling akhir, kita bisa juga menghitung jumlah key yang terdapat pada sebuah dictionary dengan fungsi len().

Contoh:

In [25]:

```
sekolah = {
  'nama': 'Sekolah Dasar Negeri Surabaya 1',
  'jenjang': 'Sekolah Dasar',
  'akreditasi': 'A'
}

print(
  "Jumlah atribut variabel sekolah adalah:",
  len(sekolah)
)
```

```
Jumlah atribut variabel sekolah adalah: 3
```