KONSEP PEMROGRAMAN

[Python 3.6.2]

Abstract

Data Structure Lanjut (Dictionary)

Puji Winar Cahyo

github.com/pwcahyo

A. Tujuan Pembelajaran.

Mahasiswa diharapkan dapat :

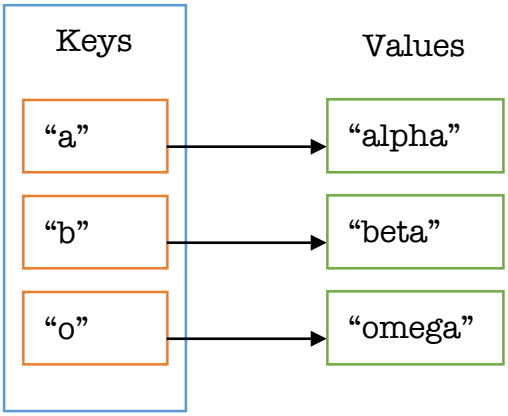
1) Memahami konsep dictionary
2) Dapat melakukan implementasi dictionary

B. Pengantar

Bahasa pemrograman python memiliki efisien key/value hash table yang disebut "dictionary". Dictionary tersebut memiliki key dan value yang saling berpasangan. Sehingga memudahkan pemrogram untuk melakukan kustomisasi dalam penamaan key pada setiap nilai value yang dipunyai.

Dictionary merupakan object yang bertipe mutable, sehingga isi data dari dictionary tersebut dapat dirubah dengan mudah. Tidak hanya itu, didalam dictionary data bisa terdiri dari immutable object maupun mutable object. Berikut contoh dictionary python :

```
dict = {"a": "alpha", "b": "beta", "o": "omega"}
```



Keterangan :

- key "a" memiliki value "alpha"
- key "b" memiliki value "beta"
- key "o" memiliki value "omega"

C. Implementasi

Dictionary merupakan data structure yang tersedia dalam bahasa pemrograman python. Data didalam dictionary diapit menggunakan kurung kurawal {…}, data didalamnya berisikan key dan value yang saling berpasangan `{key1:value1,key2:value2}`. Key dan Value dapat berbentuk immutable maupun mutable object; seperti string maupun nnumber. Beberapa method yang dapat digunakan adalah sebagai berikut.

a) Mendefinisikan dictionary

- ✓ Mendefinisikan dictionary secara langsung dengan data didalamnya.

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
```

- ✓ Mendefinisikan dictionary kosong kemudian menambahkan data didalamnya

```
>>> dict = {}
>>> dict[“a”] = “alpha”
>>> dict[“b”] = “beta”
>>> dict[“o”] = “omega”
```

- ✓ Menggunakan constructor
  - o keyword argument

```
>>> dictionary = dict(a=“alpha”, b=“beta”, o=“omega”)
>>> dictionary
{“a”:“alpha”, “b”:“beta”, “o”:“omega”}
```

  - o iterable tuple

```
>>> dictionary = dict([(“a”,“alpha”), (“b”,“beta”), (“o”,“omega”)])
>>> dictionary
{“a”:“alpha”, “b”:“beta”, “o”:“omega”}
```

  - o iterable list

```
>>> dictionary = dict(zip([“a”,“b”,“o”], [“alpha”,“beta”,“omega”]))
>>> dictionary
{“a”:“alpha”, “b”:“beta”, “o”:“omega”}
```

b) mencetak data list

nilai pada dictionary dapat dicetak menggunakan pemanggilan key data pada dictionary. berikut mencetak nilai “beta” yang berada pada key “b”.

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> print(dict[“b”])
```

mencetak isi dictionary menggunakan perulangan tanpa index

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> for data in dict:
>>>     print(data)
```

mencetak isi dictionary menggunakan perulangan dengan index

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> for index, data in dict.items():
>>>     print(“index ke %s adalah %s”%(index,data))
```

mencetak key data dari dictionary menggunakan perulangan

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> for index in dict.keys():
>>>     print(“index ke %s”%(index))
```

mencetak nilai data dari dictionary menggunakan perulangan

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> for data in dict.values():
>>>     print(“data %s”%(data))
```

b. len(dict)

mengetahui jumlah panjang atau banyaknya isi dictionary.

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> len(dict)
>>> 5
```

c. membership

method membership akan melakukan pengecekan key atau nilai yang ada didalam dictionary.

✓ Berikut pengecekan key membership, apakah “b” termasuk dalam key dict.

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> member = “b” in dict.keys()
>>> print(member)
True
```

- ✓ pengecekan nilai pada dictionary

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> member = “beta” in dict.values()
>>> print(member)
True
```

d. not membership

not membership adalah kebalikan dari membership, yaitu melakukan pengecekan key atau nilai tidak termasuk dalam dictionary

- ✓ Berikut pengecekan key not membership, apakah “b” termasuk dalam key dict.

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> member = “b” not in dict.keys()
>>> print(member)
False
```

- ✓ Berikut pengecekan nilai not membership, apakah “beta” termasuk dalam nilai dict.

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> member = “beta” not in dict
>>> print(member)
False
```

e. Copy dictionary

Dictionary dapat dicopy menjadi dictionary baru yang mempunyai object dan alamat memori yang baru menggunakan dict.copy()

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> new_dict = dict.copy()
>>> id(dict)
4324921704
>>> print(dict)
{“a”:“alpha”, “b”:“beta”, “o”:“omega”}
>>> id(new_dict)
4324921846
>>> print(new_dict)
{“a”:“alpha”, “b”:“beta”, “o”:“omega”}
>>> new_dict[“o”] = “orion”
>>> print(new_dict)
{“a”:“alpha”, “b”:“beta”, “o”:“orion”}
```

f. Mengubah nilai element pada dictionary

Nilai pada dictionary dapat diubah dengan cara memanggil data melalui key yang dimiliki data tersebut kemudian replace nilai tersebut dengan nilai baru.

```
>>> dictionary = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> dictionary[“o”] = “octal”
>>> print(dictionary)
{“a”: “alpha”, “b”: “beta”, “o”: “octal”}
```

g. Reset nilai data dictionary

Nilai data pada dictionary dapat dilakukan reset atau disesuaikan. Seperti contoh dibawah, melakukan penciptaan dictionary baru dengan melakukan reset nilai dictionary sebelumnya.

```
>>> dictionary = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> new_dict = dict.fromkeys(dictionary, “tada”)
>>> print(new_dict)
{“a”: “tada”, “b”: “tada”,“o”: “tada”}
```

h. Mengambil element data pada dictionary

Data pada dictionary dapat diambil menggunakan dict.pop(element) seperti contoh berikut melakukan pengambilan element beta pada dictionary.

```
>>> dictionary = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> element_beta = dictionary.pop(b)
>>> print(element_beta)
“beta”
>>> dictionary
{“a”: “alpha”, “o”: “omega”}
```

i. Penggabungan dictionary

Dua buah atau lebih dictionary dapat digabungkan menjadi 1 dictionary dengan menggunakan method dict.update(other_dict).

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> new_dict = {“n”: “nano”, “t”: “tetha”}
>>> dict.update(new_dict)
>>> print(dict)
{“a”: “alpha”, “b”: “beta”, “o”: “omega”, “n”: “nano”, “t”: “tetha”}
```

j. Delete element dictionary

Element didalam dictionary dapat dihapus. Penghapusan tersebut langsung akan melakukan penghapusan key dan nilai dari data dictionary secara langsung, seperti contoh berikut melakukan penghapusan data “beta” pada key “b”.

```
>>> dict = {“a”: “alpha”, “b”: “beta”, “o”: “omega”}
>>> del dict[“b”]
>>> dict
{“a”: “alpha”, “o”: “omega”}
```

Latihan Dictionary, Kerjakan secara urut.

1. Ujian UTS Konsep Pemrograman dihari pertama menghasilkan nilai sebagai berikut
   - Mady = 8
   - Roger = 5
   - Paul = 5
   - Lucy = 7

   Tuliskan data berikut kedalam python dictionary

   (nama sebagai key, hasil ujian sebagai nilai).

2. Dihari kedua ada 3 orang yang mengikuti ujian susulan dan mendapatkan nilai sebagai berikut :
   - Ken = 8
   - Andrea = 7
   - Smith = 6

   Buat dictionary baru untuk menampung data hasil ujian susulan tersebut.

3. Gabungkan hasil ujian susulan pada hari kedua kedalam hasil ujian pada hari pertama.
4. Nilai Roger, Paul dan Smith kurang memuaskan. Untuk itu mereka mengikuti remidi untuk memperbaiki hasil ujian UTS. Setelah ikut remidi nilai mereka bertiga berubah menjadi 8. Ubah dictionary data Roger, Paul dan Smith menjadi 8.
5. Tampilkan semua nama dan nilai yang dihasilkan, dengan syarat sebagai berikut : (data yang ditampilkan adalah data mahasiswa yang mempunyai nilai lebih sama dengan 8) menggunakan perulangan for/while .