Data Collection

Oleh : Agus Priyanto, M.Kom

INSTITUT TEKNOLOGI TELKOM PURWOKERTO
Smart, Trustworthy, And Teamwork

Outline Materi

- Lists
- Dictionary
- Tuples
- Set

List

- List digunakan untuk membuat data berkelompok yang nilai setiap anggotanya dapat dirubah
- List dibuat dengan tanda *bracket* ([ ]) dan setiap anggotanya dipisahkan dengan tanda koma
- Setiap anggota list disebut juga sebagai elemen list

```
>>> x = [10, 20, 30]
>>> x
[10, 20, 30]
>>>
```

Setiap list merupakan objek (instance) dari kelas list

```
>>> type (x)
<class 'list'>
>>>
```

- x merupakan variabel type int yang memiliki tiap anggota, yaitu 10, 20, dan 30

- Masing-masing bernilai bulat dan dapat diakses menggunakan indeksnya mulai dari nilai 0

```
>>> x[0]
10
>>> x[1]
20
>>> x[2]
30
>>>
```

```
>>> for data in x :
        print (data)
10
20
30
>>>
```

Mengakses elemen list dengan blok perulangan for dan while

```
>>> while i < 3 :
        print (x[i])
        i += 1
10
20
30
>>>
```

Menambahkan elemen List

- Terdapat tiga cara untuk menambahkan list, yaitu dengan metode append (), insert (), maupun extend() dari kelas list

```
>>> x = [10, 20, 30]
>>> # menggunakan metode append ()
>>> x.append(40)
>>> x
[10, 20, 30, 40]
>>>
```

```
>>> # menggunakan metode insert()
>>> x.insert(2, 50)
>>> x
[10, 20, 50, 30, 40]
>>>
>>> # menggunakan metode extend()
>>> x.extend([66, 77, 88])
>>> x
[10, 20, 50, 30, 40, 66, 77, 88]
>>>
```

Menghapus elemen List

- Terdapat tiga cara untuk menghapus elemet list, yaitu dengan perintah del, metode remove(), pop() dari kelas list

```
>>> x = [10, 20, 30, 40, 50]
>>> # menggunakan perintah del
>>> del x[2]
>>> x
[10, 20, 40, 50]
>>>
```

```
>>> # menggunakan metode remove()
>>> x.remove(20)
>>> x
[10, 40, 50]
>>>
>>> # menggunakan metode pop()
>>> nilai = x.pop()
>>> nilai
50
>>> x
[10, 40]
>>>
```

Mengurutkan elemen List

- Terdapat dua cara untuk mengurutkan elemet list, yaitu dengan fungsi sorted, metode list.sort() dari kelas list

```
>>> x = [50, 10, 40, 20, 30]
>>> # menggunakan fungsi sorted ()
>>> y = sorted (x)
>>> y
[10, 20, 30, 40, 50]
>>> # menggunakan metode sort ()
>>> x.sort ()
>>> x
[10, 20, 30, 40, 50]
>>>
```

Membalik urutan elemen List

- Untuk membalik urutan elemet list, yaitu dengan fungsi reverse() dari kelas list

```
>>> x = [50, 10, 40, 20, 30]
>>> # menggunakan fungsi reverse()
>>> x.reverse ()
>>> x
[30, 20, 40, 10, 50]
>>>
```

- Selain menggunakan metode reverse (), kita juga dapat membuat fungsi tersendiri.

```
>>> def reverse (lst):
        temp = []
        for element in lst [::-1]:
                temp.append (element)
        return temp
>>> # contoh penggunaan
>>> a = [50, 10, 40, 20, 30]
>>> b = reverse (a)
>>> b
[30, 20, 40, 10, 50]
>>>
```

Mencari nilai minimal dan maksimal

- Untuk mencari nilai minimum pada suatu kelompok data, gunakan fungsi min()
- Untuk mencari nilai maksimum pada suatu kelompok data, gunakan fungsi mak()

```
>>> x = [300, 210, 500, 100, 150]
>>> # menggunakan fungsi min()
>>> min(x)
100
>>> # menggunakan fungsi max()
>>> max(x)
500
>>>
```

Dictionary

- Dictionary digunakan untuk membuat data berkelompok yang nilai setiap anggotanya dapat dirubah, tapi indeksnya dapat ditentukan sendiri

- Dictionary dibuat dengan tanda *kurawal*({ }) dan setiap anggotanya berupa pasangan kunci-nilai (key-value)

- Antara kunci dan nilai dipisahkan dengan colon (:)

```
>>> d = {1:100, 2:200, 3:300, 4:400, 5:500}
>>> d
{1:100, 2:200, 3:300, 4:400, 5:500}
>>>
```

Setiap dictionary merupakan objek (instance) dari kelas dict

```
>>> type (d)
<class 'dict'>
>>>
```

- d merupakan sebuah dictionary yang elemennya terindeks mulai dari 1 sampai dengan 5

- Pengaksesan elemen dictionary sama seperti list menggunakan tanda bracket ([])

```
>>> d = {1:100, 2:200, 3:300, 4:400, 5:500}
>>> d [1]
100
>>> d [2]
200
>>> d [3]
300
>>>
```

```
>>> d = {1:100, 2:200, 3:300, 4:400, 5:500}
>>> d.get(1)
100
>>> d.get(2)
200
>>>
```

- Indeks dari dictionary tidak harus bertipe bilangan bulat, tapi juga bisa berupa string

```
>>> d = {'satu':100, 'dua':200, 'tiga':300}
>>> d ['satu']
100
>>> d ['dua']
200
>>> d ['tiga']
300
>>>
```

- Nilai elemen dictionary bisa berasal dari tipe apa saja dan tidak sejenis, misalnya seperti berikut

```
>>> pegawai = {
        'nip': "P001",
        'nama' : "Adi",
        'gaji' : 850000
        }
>>>
>>> pegawai ['nip']
'P001'
>>> pegawai ['nama']
'Adi'
>>> pegawai ['gaji']
850000
>>>
```

Menambahkan elemen Dictionary

- Untuk menambahkan elemen kedalam dictionary dengan menyertakan kunci dan nilai yang akan dimasukan

```
>>> # dictionary kosong
>>> d = {}
>>> # menambahkan elemen
>>> d['satu']=100
>>> d['dua']=200
>>> d['tiga']=300
>>> d
{'tiga':300, 'dua':200, 'satu':100}
>>>
```

Mengubah elemen Dictionary

- Untuk mengubah elemen, sebutkan kunci dan nilai dari elemen yang akan diubah

```
>>> d = {'satu':100, 'dua':200, 'tiga':300}
>>> d
{'tiga':300, 'dua':200, 'satu':100}
>>> # mengubah elemen
>>> d ['dua'] = 900
>>> d
{'tiga':300, 'dua':900, 'satu':100}
>>>
```

```
>>> # perintah dibawah ini akan menambahkan elemen baru
>>> # bukan mengubah elemen kedua
>>> d['Dua'} = 700
>>> d
{'tiga':300, 'Dua':700, 'dua':200, 'satu':100}
>>> len (d)
4
>>>
```

Menghapus elemen Dictionary

- Untuk menghapus elemet dictionary, yaitu dengan perintah del, dengan menyebutkan kunci elemen yang akan dihapus

```
>>> d = {'satu':100, 'dua':200, 'tiga':300}
>>> #menghapus elemen pertama
>>> del d['satu']
>>> d ['dua'] = 900
>>> d
{'tiga':300, 'dua':900}
>>>
```

```
>>> d = {'satu':100, 'dua':200, 'tiga':300,'empat':400}
>>> d
{'empat':400, 'tiga':300, 'dua':200, 'satu':100,}
>>> # daftar kunci yang akan dihapus
>>> keys = ['satu', 'tiga']
>>>
>>> #menghapus tiga elemen sekaligus
>>> for key in keys :
       del d[key]
>>> d
{'empat':400,'dua':200}
>>>
```

Tuple

- Tuple digunakan untuk membuat data berkelompok yang nilai setiap anggotanya tidak dapat dirubah

- Tuple dibuat dengan tanda *kurung*()

```
>>> t = (100, 200, 300, 400, 500)
>>> type (t)
<class 'tuple'>
>>> t
(100, 200, 300, 400, 500)
>>>
```

- Cara mengakses elemen Tuple sama seperti mengakses elemen list, yaitu dengan menggunakan indeks yang diawali dari 0

```
>>> t
(100, 200, 300, 400, 500)
>>> t[0]
100
>>> t[1]
200
>>>
```

Menghitung banyaknya elemen Tuple

- Untuk menghitung banyaknya elemen tuple menggunakan metode count()

```
>>> t =(10, 20, 10, 30, 10, 40, 20)
>>>
>>> t.count(10)
3
>>> t.count(20)
2
>>>
```

Set (Himpunan)

- Set digunakan untuk membuat data berkelompok yang tidak memiliki duplikasi data
- Data berasal dari list, dictionary, tuple, maupun string

```
>>> a = [10, 20, 30]
>>> s = set(a)
>>> s
[10, 20, 30]
>>> type (s)
<class 'set'>
>>>
```

- Set digunakan untuk menghasilkan suatu himpunan atau kelompok data yang setiap elemennya bersifat unik

- Himpunan tersebut merupakan objek dari kelas set

- Jika data sumber memiliki beberapa elemen yang sama maka elemen tersebut secara otomatis akan dihilangkan

```
>>> a = [10, 20, 10, 10, 30, 40, 20]
>>> a
[10, 20, 10, 10, 30, 40, 20]
>>> s = set(a)
>>> s
[40,10,20,30]
>>>
```

Menambahkan elemen Set

- Untuk menambahkan elemen kedalam set menggunakan metode add() atau update()

```
>>> s = set ([10, 20, 30]
>>> s
{10, 20, 30}
>>>
>>> s.add(40)
>>> s
{40, 10, 20, 30}
>>>
>>> s.update ([50, 60, 70])
>>> s
{70, 40, 10, 50, 20, 60, 30}
>>>
```

Menghapus elemen Set

- Untuk menghapus satu atau semua elemet set, yaitu dengan metode discard() dan clear()

```
>>> s = set ([10, 20, 30]
>>> s
{10, 20, 30}
>>>
>>> s.discard(20)
>>> s
{10, 30}
>>>
>>> s.clear()
>>> s
Set()
>>>
```

Mengubah elemen Set

- Objek set tidak dapat diindeks, sehingga cukup menyulitkan mengubah nilainya

```
>>> s = set ([10, 20, 30]
>>> s
{10, 20, 30}
>>>
>>> s.discard(20)
>>> s.add(50)
>>>
>>> s
{10, 50, 30}
>>>
```


Backspace
Thank You
Page Up
Home
Page Down
Insert
Shift