Python Fundamental

Ch.9 : Input Output Stream

Instruktur : Ahmad Rio Adriansyah
Nurul Fikri Komputer

Penyimpanan pada Memori

- Bersifat volatile (mudah hilang)
- Coba :

  buka python

  ```
  $ python
  ```

  buat sebuah variabel dengan nilai tertentu mis :

  ```
  >>> a = 100
  ```

  tutup python dan buka kembali

  ```
  >>> exit()
  $ python
  ```

  panggil variabel tersebut

  ```
  >>> print(a)
  ```

Penyimpanan pada File

- Permanen (relatif cukup lama)
- Mudah dipindah

- Fungsi-fungsi I/O File Stream (aliran data) biasanya digunakan untuk menangani input/output ke file baik secara berurutan ataupun acak.

Fungsi Umum I/O File Stream

- Open()
- Close()
- Write()
- Writelines()
- Read()
- Readline()
- Readlines()
- Seek()

Open()

- Sebelum sebuah file digunakan untuk dibaca atau ditulis, butuh dimasukkan dulu ke dalam interpreter python sebagai sebuah objek
- Caranya dengan memanggil nama filenya dengan fungsi **open**().
- Variabel yang menyimpan fungsi open tersebut disebut sebagai **file handler**.

Open()

- Formatnya

  >>> f = open(nama_file , mode)

- Ada beberapa mode yang dapat dilakukan, diantaranya :
  - (‘w’) Write
  - (‘r’) Read
  - (‘r+’) Read and write
  - (‘wb’) Write binary
  - (‘rb’) Read binary
  - (‘a’) Append

Open()

Buka sebuah file di komputer masing masing (file text saja agar mudah)

```
>>> f = open(‘namafile’, ‘r’)
>>> print(f)
```

Close()

- Fungsi **close**() digunakan untuk menutup stream file
- Jangan lupa melakukan pemanggilan fungsi close() setelah file selesai digunakan.

- Format :

  >>> f.close()

Write()

- Setelah ada koneksi ke file tertentu dengan fungsi open() dalam mode ‘r+’, ‘w’, ‘wb’, atau ‘a’, kita bisa mulai menuliskan sesuatu ke dalam file.

- Format :

  >>> f.write(string)

Write()

Buat sebuah file baru

```
>>> f = open('latihan.txt', 'w')
>>> f.write('Ini tulisan pertama')
>>> f.write('Yang ini tulisan kedua')
>>> f.close()
```

Lalu buka filenya melalui teks editor

Write()

- Fungsi write() menuliskan file ke samping
- Jika ingin menuliskan per baris dapat menambahkan '**\n**' di akhir string inputnya

```
>>> f = open('latihan2.txt', 'w')
>>> f.write('ini jadi baris pertama\n')
>>> f.write('yang ini jadi baris kedua\n')
>>> f.close()
```

Escape Characters

- **\n** adalah karakter yang merepresentasikan enter
- **\t** adalah karakter yang merepresentasikan tabulasi

- Keduanya (dan beberapa karakter lainnya yang didahului oleh tanda \) disebut sebagai **escape character**.

Writelines()

- Untuk memudahkan jika yang ingin dituliskan ada banyak, kita dapat memasukkan string-string tersebut ke dalam list
- Untuk memasukkannya list tersebut ke dalam file dapat menggunakan fungsi writelines()

```
...
>>> a = [‘1’,’2’]
>>> f.writelines(a)
...
```

- Mode write atau write binary akan menumpuk file sebelumnya yang sudah ada.
- Coba buka kembali file **latihan.txt** atau **latihan2.txt** dengan mode write (‘w’)
- Lalu isikan sesuatu ke dalamnya
- Perhatikan perubahan tulisan yang telah anda input sebelumnya

Mode Append

- Mode append (‘a’) menangani hal tersebut
- Mode ini membuka file untuk ditulis, tetapi meletakkan kursornya di posisi akhir file

```
>>> f.open(‘latihan2.txt’, ‘a’)
...
```

Mode Baca Tulis

- Mode write dan append tidak readable. Kita bisa menuliskan, tetapi tidak bisa memanggil stream file untuk dibaca
- Untuk membaca dapat digunakan mode ‘r’ atau ‘rb’
- Mode baca tulis (‘r+’) juga dapat digunakan untuk membaca dan menulis dalam satu file handler

Read()

- Fungsi read() membaca sejumlah karakter dari suatu stream input

- Format :

  >>> f.read(size)

- Parameter size (tipe integer) pada fungsi read() berfungsi menunjukkan berapa banyak karakter yang dibaca. Defaultnya fungsi read() akan membaca seluruh isi file.

Readline() dan Readlines()

- Fungsi **readline**() membaca baris per baris (dibatasi oleh karakter \n)
- Bedakan dengan **readlines**() , yang akan membaca file dan memasukkan tiap barisnya ke dalam sebuah list

Seek()

- Kursor pembacaan file akan berpindah sesuai dengan perintah yang dijalankan
- Fungsi seek() dapat digunakan untuk mengatur posisi kursor ke posisi tertentu pada file
- Tidak semua file dapat diterapkan fungsi seek()

- Format :

  >>> f.seek(offset, whence)

Seek()

```
>>> f.seek(offset, whence)
```

- Argumen fungsi seek bukan keyword argument
- Offset (argumen pertama) menunjukkan posisi dalam integer
- Whence (argumen kedua) adalah opsional, menunjukkan status pembacaan posisi : dari awal file (0), dari posisi saat ini (1), atau dari akhir file (2)

Latihan

- Buat sebuah file berisi 25 baris bilangan. Masing masing baris adalah bilangan yang berbeda (acak, tidak perlu berurutan)
- Baca file tersebut, urutkan, lalu tuliskan kembali ke dalam file baru