TEXT FILE DALAM PYTHON

1. Penjelasan pengertian Text File

Dalam Python, bekerja dengan file teks adalah suatu hal umum dan dilakukan dengan menggunakan fungsi-fungsi bawaan seperti **open(), read(), write(),** dan **close()**. Berikut adalah contoh penggunaan file teks di Python beserta penjelasan masing-masing langkah:

1) Membuka file / open()
   Untuk membuka file, Anda menggunakan fungsi open() dengan menyediakan nama file dan mode (misalnya, 'r' untuk membaca atau read, 'w' untuk menulisatau write, 'a' untuk menambahkan atau append, dll.).
   Contohnya:

```
# Membuka file untuk membaca ('r')
file_path = "contoh.txt"
file = open(file_path, 'r')
```

2) Membaca file / read()
   Anda dapat menggunakan metode read() untuk membaca seluruh isi file atau readline() untuk membaca satu baris pada satu waktu.
   Contohnya:

```
# Membaca isi file
konten = file.read()
print(konten)
```

3) Menulis atau menambahkan file / write()
   Jika Anda membuka file dengan mode 'w' (menulis) atau 'a' (menambahkan), Anda dapat menggunakan metode write() untuk menulis ke file.
   Contohnya:

```
# Membuka file untuk menulis ('w')
file_path = "contoh.txt"
file = open(file_path, 'w')

# Menulis ke file
file.write("Ini adalah contoh penambahan kalimat")
file.close()
```

4) Menutup file /close
   Penting untuk selalu menutup file setelah selesai menggunakan file tersebut. Ini dapat dilakukan dengan memanggil metode close() pada objek file jika tidak akan berakibat fatal bagi code.
   Contohnya:

```
# Menutup file setelah selesai
file.close()
```

5) Contoh lengkap

```
#Membuka dan Menambahkan kalimat dengan write()

file_path = "contoh.txt"
file = open(file_path,"w") # w untuk write

file.write("ini adalah file external.")
file.close()

#membaca file untuk membaca
file = open(file_path,"r")
konten = file.read()
print(konten)

#menutup
file.close()
```

2. Penjelasan perbedaan method read() dan open() dan contoh syntaxnya.

a. Read() method
read() adalah metode yang digunakan pada objek file untuk membaca kontennya. Ketika read() dipanggil pada objek file, itu membaca seluruh isi file atau sejumlah byte tertentu (jika argumen jumlah byte diberikan).
Contohnya:

```
#membaca file read()
file_path = “contoh.txt”
file = open(file_path,”r”) r untuk read()

#membaca seluruh isi file
konten = file.read()
print(konten)
```

b. open() method
open() adalah fungsi yang digunakan untuk membuka file. Fungsi ini mengembalikan objek file yang dapat digunakan untuk melakukan operasi berbeda pada file tersebut, seperti membaca, menulis, atau menutupnya. Sebenarnya tidak ada perbedaan yang signifikan antara keduanya
Contohnya:

```
#membuka file untuk membaca “r”
file_path = “contoh.txt”
file = open(file_path,”r”)

#lakukan perintah operasi pada file
file.close()
```

Secara ringkas:
‘Open()’ digunakan untuk membuka file dan mengembalikan objek file.
‘read()’ adalah metode yang dipanggil pada objek file untuk membaca kontennya.

Dalam prakteknya, biasanya kita menggunakan open() untuk membuka file, dan kemudian kita menggunakan metode seperti read(), write(), atau metode lainnya yang sesuai dengan kebutuhan kita pada objek file yang dikembalikan oleh open()

3. perbedaan mode 'w' dan mode 'a' pada method open()

1) Mode ‘w’ write
Mode 'w' digunakan untuk membuka file dalam mode penulisan. Jika file sudah ada, kontennya akan dihapus, dan file tersebut akan dianggap kosong. Jika file tidak ada, file baru akan dibuat.
Contohnya:

```
# Membuka file untuk menulis ('w')
file_path = "contoh.txt"
file = open(file_path, 'w')

# Menulis ke file
file.write("Ini adalah contoh kalimat yang ditulis ke
file.")

# Menutup file setelah selesai menulis
file.close()
```

2) Mode ‘a’ append

Mode 'a' digunakan untuk membuka file dalam mode penambahan. Jika file sudah ada, data baru akan ditambahkan di akhir file tanpa menghapus isi yang sudah ada. Jika file tidak ada, file baru akan dibuat.

Contohnya:

```
# Membuka file untuk menambahkan ('a')
file_path = "contoh.txt"
file = open(file_path, 'a')


# Menambahkan data ke file
file.write("\nMenambahkan kalimat pada baris kedua.")


# Menutup file setelah selesai menambahkan
file.close()
```

Secara ringkas:

Mode ‘w’ write: digunakan untuk menulis file dan menghapusnya jika sudah ada.

Mode ‘a’ append: digunakan untuk menambahkan data pada akhir file tersebut tanpa menghapusnya`

Exception

1. Penjelasan tentang Exception

   Dalam Python, exception merujuk pada situasi atau kondisi yang dapat terjadi selama eksekusi program dan berpotensi menghentikan jalannya program. Pengelolaan exception memungkinkan pengguna untuk secara sistematis menanggapi kesalahan atau situasi khusus yang mungkin timbul selama eksekusi.

   Konsep utama terkait exception di Python melibatkan pernyataan **try, except**, dan **finally**:

   - ‘try’: Digunakan untuk melindungi blok kode yang mungkin memunculkan exception.
   - ‘except’: Blok ini akan dieksekusi jika exception terjadi di dalam blok try. Jenis exception dapat disebutkan untuk menangani kasus tertentu.
   - ‘finally’: Blok ini berisi kode yang akan dieksekusi tanpa peduli apakah exception terjadi atau tidak.

2. Jenis-jenis untuk menangani Exception Error dan contoh syntaxnya

   Contohnya:
   - Menggunakan blok ‘try’ dan ‘except’

     ```
     try:
          #kode yang menimbulkan exception
          a = 10 / 0
     except ZeroDivisionError: #PEMBAGIAN TIDAK BISA DI BAGI 0
          print(“tidak bisa di bagi 0”)
     ```

   - Block ‘finally’ pada ‘try’-‘except’

     ```
     try:
          #kode yang menimbulkan exception
          a = 10 / 0
     except ZeroDivisionError:
          print(“tidak bisa di bagi 0”)
     finally:
          # Blok ini dijalankan selalu, baik ada exception atau
          tidak
          print("Eksekusi blok finally.")
     ```

nama : Muhammad Arham Arsyad

NIM : 231031042
kelas : SI23-B