



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	<71230998>
Nama Lengkap	<MERLYANA HELENA PATRICIA>
Minggu ke / Materi	09/Membaca dan Menulis File

SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.

SAYA SANGGUP MENERIMA SANKSI JIKA MELAKUKAN KEGIATAN PLAGIASI, TERMASUK SANKSI TIDAK LULUS MATA KULIAH INI.

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
YOGYAKARTA

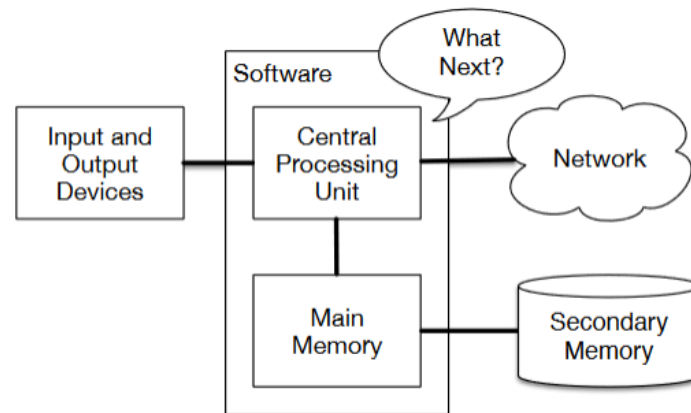
2024

BAGIAN 1: MATERI MINGGU INI (40%)

MATERI

Pengantar file

Pada praktikum ini, kita belajar tentang pentingnya pengelolaan data dalam sebuah program komputer. Kita mengetahui bahwa program yang dijalankan menggunakan memori primer, di mana semua data disimpan secara sementara. Namun, ketika program selesai dijalankan, semua data tersebut akan hilang. Oleh karena itu, untuk menjaga agar data tidak hilang dan dapat diakses kembali di kemudian hari, kita menggunakan penyimpanan sekunder, yaitu file. File merupakan tempat menyimpan secara permanen di dalam secondary memory yang berisi kumpulan informasi yang saling berelasi. Setiap file memiliki atribut seperti nama, ukuran, lokasi penyimpanan, dan lainnya, yang memungkinkan kita untuk mengorganisir dan mengakses data dengan lebih efisien.

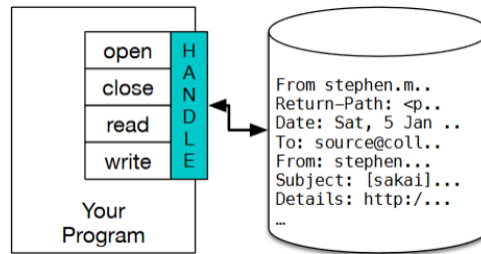


Gambar 8.1: Secondary Memory di Komputer

Pengaksesan file

Berikut ini adalah langkah-langkah pengaksesan file dengan python melalui vc code:

1. Menyiapkan file dan path yang akan diakses
2. Open file dengan menggunakan perintah `handle = open(nama file yang ingin diakses)`
3. Lakukan sesuatu dengan file yang telah diakses tersebut, seperti ditampilkan (read) isinya atau diubah / ditulisi (write)
4. Close file



Gambar 8.3: Ilustrasi Handle File

```
1 handle = open('mbox-short.txt')
2 print(handle)
```

Maka output yang dihasilkan:

```
PS C:\Users\Lenovo\file> & C:/msys64/mingw64/bin/python.exe c:/Users/Lenovo/file/latihan9.py
<_io.TextIOWrapper name='mbox-short.txt' mode='r' encoding='cp1252'>
```

Jadi, hasil dari pengaksesan file ini adalah tampilan nama file, mode aksesnya (contohnya: r untuk read), dan informasi encoding yang digunakan (biasanya unicode UTF-8) dari modul io pada Python. Jika nama file yang diberikan tidak ditemukan atau tidak ada, maka akan muncul error.

Contoh: Ketika kita membuka file "data.txt" untuk dibaca (`open("data.txt")`), kita akan mendapatkan handle untuk membaca file tersebut. Jika file tersebut tidak ada di dalam folder, maka akan muncul error.

```
PS C:\Users\Lenovo\file> & C:/msys64/mingw64/bin/python.exe c:/Users/Lenovo/file/latihan9.py
Traceback (most recent call last):
  File "c:/Users/Lenovo/file/latihan9.py", line 1, in <module>
    handle = open('data.txt')
             ^^^^^^^^^^^^^^^
FileNotFoundError: [Errno 2] No such file or directory: 'data.txt'
```

Manipulasi File

Untuk memanipulasi file, langkah pertama yang harus dilakukan adalah membaca file tersebut. Berikut adalah cara membaca file pada Python:

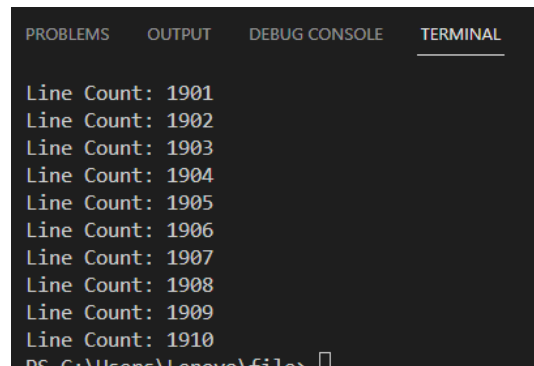
1. Siapkan file yang akan di manipulasi
2. Open file dengan cara yang sama ketika mengakses file yang sudah disiapkan

3. Loop file secara baris-perbaris untuk menghindari hang / crash ketika file yang digunakan berkapasitas besar.
4. Close file

```
1 handle = open('mbox-short.txt')
2 count = 0
3 for line in handle:
4     count = count + 1
5     print('Line Count:', count)
```

Untuk membuka file teks dengan nama 'mbox-short.txt' dan menghitung jumlah baris dalam file tersebut. Pertama, kita membuka file menggunakan fungsi `open()` dan menyimpannya dalam variabel `handle`. Selanjutnya, variabel `count` diinisialisasi dengan nilai 0 untuk menyimpan jumlah baris. Kemudian, dilakukan looping untuk setiap baris dalam file menggunakan perintah `for line in handle:`. Pada setiap iterasi, nilai `count` ditambah 1, sehingga menghitung jumlah baris yang telah dilewati. Setelah looping selesai, hasil jumlah baris dicetak menggunakan perintah `print('Line Count:', count)`. Akhirnya, file ditutup menggunakan perintah `handle.close()` untuk memastikan bahwa sumber daya file telah dilepaskan. Ini adalah langkah yang penting untuk mencegah kebocoran sumber daya dan memastikan kerja yang efisien dalam pemrosesan file.

Outputnya :



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Line Count: 1901
Line Count: 1902
Line Count: 1903
Line Count: 1904
Line Count: 1905
Line Count: 1906
Line Count: 1907
Line Count: 1908
Line Count: 1909
Line Count: 1910
PS C:\Users\Lenovo\files>
```

Untuk menampilkan ukuran file teks dalam bytes, kita dapat menggunakan fungsi `len()` untuk menghitung jumlah byte dalam string yang merupakan isi dari file teks. Berikut adalah contoh cara melakukannya:

```
1 handle = open('mbox-short.txt')
2 hasil = handle.read()
3 print("Ukuran: " + len(hasil) + "bytes")
4 print("Huruf dari belakang sendiri mundur 16 huruf adalah: " + hasil[-16::1])
```

Untuk membuka file 'mbox-short.txt', membaca isinya, dan melakukan dua operasi berbeda. Pertama, kode menghitung dan mencetak ukuran file dalam bytes dengan menggunakan fungsi `len()` untuk menghitung panjang string yang berisi seluruh teks dari file tersebut. Setelah itu, ukuran tersebut dikonversi menjadi string dan digabungkan dengan string " bytes" sebelum dicetak. Kedua, kode mencetak 16 karakter terakhir dari isi file tersebut dengan menggunakan slicing `hasil[-16:]`. Ini berguna untuk melihat bagian akhir dari teks file. Namun, penting untuk diingat bahwa setelah selesai membaca file, kita harus menutupnya menggunakan `handle.close()` untuk memastikan sumber daya file telah dilepaskan.

```
1  handle = open('mbox-short.txt')
2  count = 1
3  for line in handle:
4      if line.startswith("Date:") and count <= 10:
5          count += 1
6          print(line)
```

Outputnya:

```
PS C:\Users\Lenovo\file> & C:/msys64/mingw64/bin/python.exe c:/Users/Lenovo/file/latihan
Date: Sat, 5 Jan 2008 09:12:18 -0500

Date: 2008-01-05 09:12:07 -0500 (Sat, 05 Jan 2008)

Date: Fri, 4 Jan 2008 18:08:57 -0500

Date: 2008-01-04 18:08:50 -0500 (Fri, 04 Jan 2008)

Date: Fri, 4 Jan 2008 16:09:02 -0500

Date: 2008-01-04 16:09:01 -0500 (Fri, 04 Jan 2008)

Date: Fri, 4 Jan 2008 15:44:40 -0500

Date: 2008-01-04 15:44:39 -0500 (Fri, 04 Jan 2008)

Date: Fri, 4 Jan 2008 15:01:38 -0500

Date: 2008-01-04 15:01:37 -0500 (Fri, 04 Jan 2008)
```

Program di atas bertujuan untuk membuka file 'mbox-short.txt' dan menampilkan hanya 10 baris yang dimulai dengan "Date:". Pada awalnya, file dibuka dan handle-nya disimpan dalam variabel `handle`. Variabel `count` diinisialisasi dengan nilai 1, yang akan digunakan untuk menghitung jumlah baris yang sesuai dengan kriteria. Selama proses looping, setiap baris dari file diperiksa menggunakan fungsi `startswith()` apakah dimulai dengan "Date:" dan apakah jumlah baris yang sudah ditemukan masih kurang dari atau sama dengan 10. Jika kedua kondisi tersebut terpenuhi, maka baris tersebut dicetak dan nilai `count` ditambah 1. Dengan demikian, program ini akan mencetak 10 baris pertama yang sesuai dengan kriteria "Date:" dari file 'mbox-short.txt'.

Penyimpanan file

Penyimpanan file dalam Python mirip dengan membuka file, namun dengan metode yang berbeda. Untuk menulis ke file, kita menggunakan metode 'w' (write) saat membuka file dengan 'open()'.

Contohnya dengan cara 'handle=open('output.txt','w')'. Setelah membuka file, kita bisa menuliskan teks langsung ke dalamnya dengan 'write(<teks>)'. Terakhir, jangan lupa untuk menutup file setelah selesai menulis.

```
1 handle = open('output.txt','w')
2 tulisan = "teks ini akan dituliskan ke file\n"
3 handle.write(tulisan)
4 handle.close()
```

BAGIAN 2: LATIHAN MANDIRI (60%)

Link Github : <https://github.com/memel25/71230998-Pertemuan9.git>

SOAL 8.1

```
def banding(file1, file2):
    with open(file1, "r") as file1:
        ls1 = [line.rstrip() for line in file1]

    with open(file2, "r") as file2:
        ls2 = [line.rstrip() for line in file2]

    gabung = zip(ls1, ls2)
    no = 1
    for l1, l2 in gabung:
        if l1 != l2:
            print(f"Perbedaan di baris {no}:")
            print(f"File 1: {l1}")
            print(f"File 2: {l2}")
        no += 1

file1 = input("Masukkan nama file pertama : ")
file2 = input("Masukkan nama file kedua : ")
banding(file1, file2)
```

Masukkan nama file pertama :

Fungsi `banding` membuka dua file teks yang diberikan, membaca setiap baris dari masing-masing file, dan membandingkannya satu per satu. Jika ada perbedaan antara baris-baris tersebut, program akan mencetak nomor baris dan isi baris dari kedua file. Setiap baris yang dibaca dari file dihapus karakter newline di ujungnya dengan menggunakan `rstrip()`. Variabel `gabung` adalah hasil dari fungsi `zip()` yang menggabungkan kedua list baris dari file. Selanjutnya, program melakukan iterasi melalui gabungan baris dari kedua file menggunakan loop `for`, dengan menyertakan nomor baris. Jika ada perbedaan antara baris dari kedua file, program akan mencetak perbedaan tersebut.[SOAL 8.2](#)

```
def kuis(file):
    with open(file, "r") as f:
        ls = [line.rstrip() for line in f]

    for l in ls:
        soal = l.split("||")
        print(f"Pertanyaan: {soal[0]}")
        kunci = soal[1].lower().strip()
        jawaban = input("Jawab: ").lower().strip()

        if kunci == jawaban:
            print("Jawaban benar!")
        else:
            print("Jawaban salah!")

file = input("Sebutkan file soal : ")
kuis(file)
```

Fungsi `kuis()` membuka file yang diberikan (`file`) dan membacanya baris per baris. Setiap barisnya disimpan dalam list `ls`. Kemudian, program melakukan loop melalui setiap baris dalam list `ls`. Setiap baris dipisahkan menjadi dua bagian, pertanyaan dan jawaban, yang dipisahkan oleh `||`. Pertanyaan dicetak ke layar. Jawaban dari pengguna diminta melalui `input()`, kemudian diubah menjadi huruf kecil dan dihapus spasi di sekitarnya menggunakan `.lower().strip()`. Setelah itu, jawaban pengguna dibandingkan dengan jawaban yang benar. Jika sama, maka program mencetak "Jawaban benar!", jika tidak, mencetak "Jawaban salah!". Baris terakhir meminta pengguna untuk memasukkan nama file yang berisi pertanyaan dan jawaban, kemudian menjalankan fungsi `kuis()` dengan file yang diberikan. Jadi, secara keseluruhan, program ini membaca file teks dengan format pertanyaan dan jawaban dipisahkan oleh `||`, menampilkan pertanyaan dari file, meminta pengguna untuk menjawab, dan memberikan umpan balik apakah jawaban itu benar atau salah.