



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	71230977
Nama Lengkap	MICHAEL HOSEA
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%)

Pada bagian ini, tuliskan kembali semua materi yang telah anda pelajari minggu ini. Sesuaikan penjelasan anda dengan urutan materi yang telah diberikan di saat praktikum. Penjelasan anda harus dilengkapi dengan contoh, gambar/ilustrasi, contoh program (source code) dan outputnya. Idealnya sekitar 5-6 halaman.

Dalam pengembangan aplikasi, seringkali kita perlu berinteraksi dengan file untuk membaca data dari file eksternal atau menyimpan data ke dalam file. Dalam Python, ada beberapa cara untuk melakukan operasi membaca dan menulis file. Berikut adalah penjelasan tentang cara membaca dan menulis file dalam Python:

Materi 1.1: Membuka File

Membaca file adalah langkah awal untuk mendapatkan data dari file eksternal ke dalam program Python. Untuk membaca file, kita perlu membuka file tersebut dengan fungsi `open()`. Fungsi ini memiliki dua parameter utama: nama file dan mode.

- Mode 'r' (Read): Mode ini digunakan untuk membuka file dalam mode baca.
- Mode 'rb' (Read Binary): Mode ini digunakan untuk membaca file biner.

```
file = open('file.txt', 'r')
```

Materi 1.2: Membaca File

Setelah membuka file, kita dapat membaca isi file dengan berbagai cara:

- Metode `read()`: Membaca seluruh konten file sebagai satu string.
- Metode `readline()`: Membaca satu baris teks pada setiap pemanggilan.
- Looping through file: Menggunakan loop `for` untuk membaca baris demi baris.

```
# Membaca seluruh isi file
content = file.read()
```

```
# Membaca satu baris
line = file.readline()
```

```
# Loop membaca baris per baris
for line in file:
    print(line)
```

Materi 1.3: Menutup File

Setelah selesai membaca, penting untuk menutup file menggunakan metode `close()` agar sumber daya sistem dibebaskan.

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

Materi 2.1: Menulis ke File untuk menulis

Selain membaca, kita juga sering perlu menulis data ke dalam file. Untuk menulis ke file, kita perlu membuka file dalam mode tertentu:

- Mode 'w' (Write): Membuka file untuk menulis. Jika file tidak ada, akan dibuat. Jika sudah ada, akan dihapus dan diganti dengan file baru.
- Mode 'a' (Append): Membuka file untuk menambahkan konten di akhir file.

```
# Contoh membuka file untuk menulis
file = open('file.txt', 'w')
atau
file = open('log.txt', 'a')
```

Materi 2.2: Menulis ke File

Setelah membuka file untuk menulis, kita dapat menulis ke file dengan menggunakan metode write().

```
# Menulis ke file
file.write("Hello, World!")
```

Materi 2.3: Menutup File Setelah Menulis

Setelah selesai menulis, kita juga harus menutup file untuk memastikan data telah ditulis dengan benar.

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

Materi 3 : Membaca dan Menulis dalam Mode yang Berbeda

Terkadang, kita juga bisa membaca dan menulis dalam mode yang berbeda pada satu file.

```
# Membuka file untuk membaca dan menulis
file = open('file.txt', 'r+')

# Membaca isi file
content = file.read()

# Menambahkan teks ke akhir file
file.write("\nAdding more content!")
```

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

Mode 'r+' adalah mode baca dan tulis pada file dalam Python. Dalam mode ini, kita dapat membaca (read) dan menulis (write) ke file yang sama.

Arti Mode 'r+':

- Read (Baca): Memungkinkan untuk membaca konten yang sudah ada dari file.
- Write (Tulis): Memungkinkan untuk menulis konten baru ke file yang sudah ada.
- Truncate (Pemotongan): File tidak akan dipotong (truncated) saat dibuka, jadi posisi awal untuk menulis adalah di awal file.

```
# Membuka file dalam mode r+
file = open('data.txt', 'r+')

# Membaca isi file
content = file.read()
print("Isi File Sebelum:", content)

# Menulis data baru ke file
file.write("\nData baru ditambahkan!")

# Kembali ke awal file untuk memperbarui
file.seek(0)

# Membaca isi file setelah menulis
updated_content = file.read()
print("Isi File Setelah:", updated_content)

# Menutup file
file.close()
```

Dalam contoh ini, kita membuka file `data.txt` dalam mode 'r+' (baca dan tulis), membaca isinya, menambahkan data baru di akhir file, kembali ke awal file untuk memperbarui isinya, kemudian menutup file setelah selesai. Mode 'r+' berguna saat kita ingin membaca dan menulis ke file yang sama tanpa harus membukanya dua kali (sekaligus efisien). Namun, perlu diperhatikan bahwa jika kita menulis data lebih pendek dari data yang sudah ada, data yang baru akan menempa bagian yang sudah ada sebelumnya. Jadi hati-hati dalam penggunaannya.

`file.seek()` adalah sebuah metode yang digunakan untuk mengatur posisi kursor atau pointer baca/tulis di dalam file yang sedang dibuka. Dengan menggunakan `seek()`, kita dapat menggerakkan posisi ini ke lokasi tertentu dalam file.

Format Umum:

```
file.seek(offset, whence)
```

Parameter:

- `offset`: Jumlah byte untuk menggeser posisi kursor. Jika `whence` adalah 0 (default), maka ini adalah posisi dari awal file. Jika `whence` adalah 1, maka ini adalah posisi dari posisi sekarang. Jika `whence` adalah 2, maka ini adalah posisi dari akhir file.
- `whence`: Parameter opsional yang menentukan titik referensi untuk `offset`. Nilainya bisa:

0: Mulai dari awal file (default).

1: Mulai dari posisi sekarang.

2: Mulai dari akhir file.

Materi 4 : Menangani Error dengan try-except

Ketika bekerja dengan file, penting juga untuk menangani kemungkinan error, seperti file tidak ditemukan.

```
# Contoh menulis dengan penanganan error
try:
    file = open('file.txt', 'w')
    file.write("Hello, World!")
    file.close()
except FileNotFoundError:
    print("File tidak ditemukan!")
```

BAGIAN 2: LATIHAN MANDIRI (60%)

Pada bagian ini anda menuliskan jawaban dari soal-soal Latihan Mandiri yang ada di modul praktikum. Jawaban anda harus disertai dengan source code, penjelasan dan screenshot output.

SOAL 1

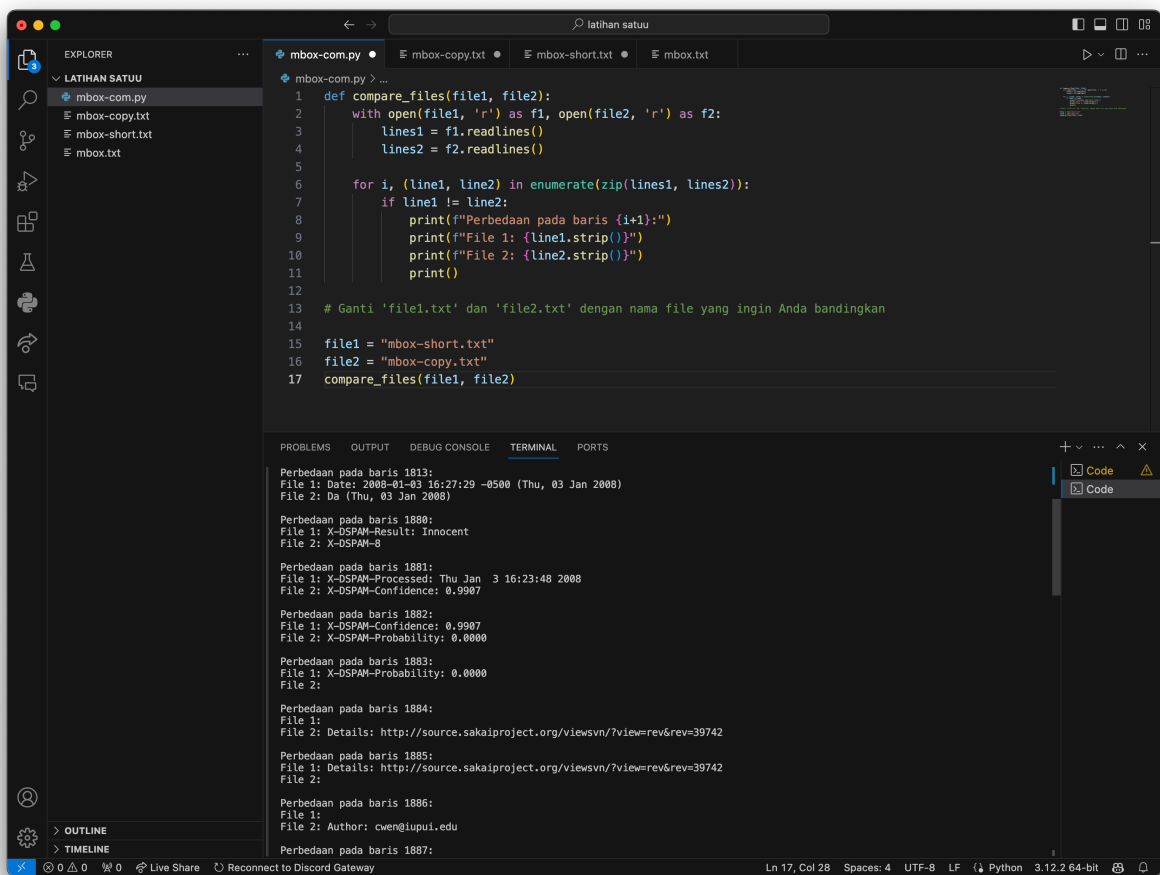
Buatlah sebuah program yang dapat membandingkan 2 buah file teks dan kemudian menampilkan perbedaan antar kedua teks per barisnya jika ada perbedaan!

- Source Code dan Output

```
def compare_files(file1, file2):  
    with open(file1, 'r') as f1, open(file2, 'r') as f2:  
        lines1 = f1.readlines()  
        lines2 = f2.readlines()  
  
    for i, (line1, line2) in enumerate(zip(lines1, lines2)):  
        if line1 != line2:  
            print(f"Perbedaan pada baris {i+1}:")  
            print(f"File 1: {line1.strip()}")  
            print(f"File 2: {line2.strip()}")  
            print()
```

Ganti 'file1.txt' dan 'file2.txt' dengan nama file yang ingin Anda bandingkan

```
file1 = "mbox-short.txt"  
file2 = "mbox.txt"  
compare_files(file1, file2)
```



```
1 def compare_files(file1, file2):
2     with open(file1, 'r') as f1, open(file2, 'r') as f2:
3         lines1 = f1.readlines()
4         lines2 = f2.readlines()
5
6     for i, (line1, line2) in enumerate(zip(lines1, lines2)):
7         if line1 != line2:
8             print(f"Perbedaan pada baris {i+1}:")
9             print(f"File 1: {line1.strip()}")
10            print(f"File 2: {line2.strip()}")
11            print()
12
13 # Ganti 'file1.txt' dan 'file2.txt' dengan nama file yang ingin Anda bandingkan
14
15 file1 = "mbox-short.txt"
16 file2 = "mbox-copy.txt"
17 compare_files(file1, file2)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Perbedaan pada baris 1813:
File 1: Date: 2008-01-03 16:27:29 -0500 (Thu, 03 Jan 2008)
File 2: Da (Thu, 03 Jan 2008)

Perbedaan pada baris 1880:
File 1: X-SPAM-Result: Innocent
File 2: X-SPAM-8

Perbedaan pada baris 1881:
File 1: X-SPAM-Processed: Thu Jan 3 16:23:48 2008
File 2: X-SPAM-Confidence: 0.9907

Perbedaan pada baris 1882:
File 1: X-SPAM-Confidence: 0.9907
File 2: X-SPAM-Probability: 0.0000

Perbedaan pada baris 1883:
File 1: X-SPAM-Probability: 0.0000
File 2:

Perbedaan pada baris 1884:
File 1:
File 2: Details: <http://source.sakaiproject.org/viewsvn/?view=rev&rev=39742>

Perbedaan pada baris 1885:
File 1: Details: <http://source.sakaiproject.org/viewsvn/?view=rev&rev=39742>
File 2:

Perbedaan pada baris 1886:
File 1:
File 2: Author: cwen@iupui.edu

Perbedaan pada baris 1887:

Ln 17, Col 28 Spaces: 4 UTF-8 LF Python 3.12.2 64-bit

- Penjelasan

Kode ini adalah sebuah fungsi Python yang dibuat untuk membandingkan dua file teks baris per baris. Fungsi ini bernama `compare_files` dan menerima dua parameter, yaitu `file1` dan `file2`, yang merupakan nama file yang ingin dibandingkan.

Langkah-langkah Kode:

1. Fungsi `compare_files` dibuat dengan dua parameter: `file1` dan `file2`.
2. Fungsi menggunakan `with` statement untuk membuka kedua file secara bersamaan dengan mode `"r"` (baca) menggunakan `open()`.
3. Menggunakan `readlines()`, fungsi ini membaca seluruh konten dari kedua file dan menyimpan setiap baris dalam list: `lines1` untuk `file1` dan `lines2` untuk `file2`.
4. Kemudian, fungsi melakukan iterasi melalui kedua list `lines1` dan `lines2` secara bersamaan menggunakan `enumerate` dan `zip`.
 - `enumerate` memberikan indeks (`i`) dan nilai (`line1`, `line2`) untuk setiap pasangan baris dalam `lines1` dan `lines2`.
 - `zip` menggabungkan baris-baris dari kedua file sehingga bisa dibandingkan.

Pengecekan Perbedaan:

1. Dalam loop, setiap pasangan baris dari `file1` (`line1`) dan `file2` (`line2`) dibandingkan.

2. Jika terdapat perbedaan pada baris tersebut (`line1 != line2`), maka pesan perbedaan ditampilkan:
 - "`Perbedaan pada baris {i+1}:`" mencetak nomor baris dimana perbedaan ditemukan (diindeks dari 1 bukan 0).
 - "`File 1: {line1.strip()}`" mencetak baris dari file1 tanpa spasi ekstra di awal atau akhir.
 - "`File 2: {line2.strip()}`" mencetak baris dari file2 tanpa spasi ekstra di awal atau akhir.
 - `print()` digunakan untuk memberikan baris kosong setelah setiap perbedaan.

SOAL 2

Buatlah sebuah program untuk menampilkan soal sederhana yang diambil dari file teks `soal.txt` yang memiliki format sebagai berikut:

```
1+1 = || 2
Bendera Indonesia? || Merah Putih
Kota gudeg adalah: || Yogyakarta
Komponen PC untuk penyimpanan file adalah... || harddisk
50 * 20 = || 1000
```

Dari soal tersebut tampilkan sbb:

```
nama file1: soal.txt
1+1 =
Jawab: 2
Jawaban benar!
Bendera Indonesia?
Jawab: merah putih
Jawaban benar!
Kota gudeg adalah:
Jawab: yogya
Jawaban salah!
Komponen PC untuk penyimpanan file adalah...
Jawab: HARDDISK
Jawaban benar!
```

- Source Code dan Output




```
def compare_files(file1, file2):
    with open(file1, 'r') as f1, open(file2, 'r') as f2:
        lines1 = f1.readlines()
        lines2 = f2.readlines()

    for i, (line1, line2) in enumerate(zip(lines1, lines2)):
        if line1 != line2:
            print(f"Perbedaan pada baris {i+1}:")
            print(f"File 1: {line1.strip()}")
            print(f"File 2: {line2.strip()}")
            print()

# Ganti 'file1.txt' dan 'file2.txt' dengan nama file yang ingin Anda bandingkan

file1 = "mbox-short.txt"
file2 = "mbox-copy.txt"
compare_files(file1, file2)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Perbedaan pada baris 1813:
File 1: Date: 2008-01-03 15:27:29 -0500 (Thu, 03 Jan 2008)
File 2: Da (Thu, 03 Jan 2008)

Perbedaan pada baris 1880:
File 1: X-SPAM-Result: Innocent
File 2: X-SPAM-8

Perbedaan pada baris 1881:
File 1: X-SPAM-Processed: Thu Jan 3 16:23:48 2008
File 2: X-SPAM-Confidence: 0.9907

Perbedaan pada baris 1882:
File 1: X-SPAM-Confidence: 0.9907
File 2: X-SPAM-Probability: 0.0000

Perbedaan pada baris 1883:
File 1: X-SPAM-Probability: 0.0000
File 2:

Perbedaan pada baris 1884:
File 1:
File 2: Details: <http://source.sakaiproject.org/viewsvn/?view=rev&rev=39742>

Perbedaan pada baris 1885:
File 1: Details: <http://source.sakaiproject.org/viewsvn/?view=rev&rev=39742>
File 2:

Perbedaan pada baris 1886:
File 1:
File 2: Author: cwen@iupui.edu

Perbedaan pada baris 1887:

- Penjelasan

Fungsi `tampilkan_soal` ini adalah untuk membantu dalam menampilkan soal dan memeriksa jawaban yang diberikan oleh pengguna. Fungsi ini membaca setiap baris dari file teks yang berisi pasangan soal dan jawaban, kemudian menampilkan soal kepada pengguna, meminta jawaban, dan memeriksa apakah jawaban yang diberikan benar atau salah.

Langkah-langkah Kode:

1. Fungsi `tampilkan_soal` dibuat dengan satu parameter: `nama_file`, yang merupakan nama file teks yang berisi pasangan soal dan jawaban.
2. Dalam blok `with`, file `nama_file` dibuka dalam mode "r" (baca) menggunakan `open()`.
3. `readlines()` digunakan untuk membaca seluruh konten dari file tersebut, dan setiap barisnya disimpan dalam list `soal_jawaban`.
4. Setiap baris dalam `soal_jawaban` diiterasi menggunakan loop `for`.
 - Baris tersebut kemudian di-split berdasarkan string " | " menjadi dua bagian: soal dan jawaban.
 - soal akan dicetak ke layar sebagai pertanyaan kepada pengguna.
 - Pengguna diminta untuk memasukkan jawaban dengan `input()`.
 - Jawaban pengguna (`jawaban_user`) dibandingkan dengan jawaban yang benar (`jawaban`) tanpa memperhatikan huruf besar/kecil (lowercase).

- Jika jawaban pengguna sama dengan jawaban yang benar, maka pesan "Jawaban benar!" akan dicetak.
- Jika jawaban pengguna tidak sama dengan jawaban yang benar, maka pesan "Jawaban salah!" akan dicetak.