Modul Praktikum Python: Bekerja dengan File

Pendahuluan

Pada praktikum ini, mahasiswa akan mempelajari cara bekerja dengan berbagai jenis file menggunakan Python, termasuk file teks, CSV, dan JSON. Mahasiswa juga akan mengerjakan beberapa studi kasus untuk memahami penerapan Python dalam pengelolaan file secara praktis.

Tujuan Pembelajaran

1. Mahasiswa mampu membaca dan menulis file teks.
2. Mahasiswa mampu mengelola file CSV.
3. Mahasiswa memahami cara memproses file JSON.
4. Mahasiswa dapat menyelesaikan studi kasus berbasis file secara mandiri.

Dasar Teori

Python menyediakan fungsi bawaan `open()` untuk bekerja dengan file.

Mode yang sering digunakan

- `"r"` → membaca file.
- `"w"` → menulis dan menimpa file.
- `"a"` → menambahkan data ke akhir file.
- `"rb"`, `"wb"` → mode biner.

Struktur umum penggunaan file

```
with open("nama_file.txt", "r") as f:
    isi = f.read()
```

Praktikum & Studi Kasus

Kasus 1 — Membaca & Menulis File Teks

Deskripsi

Buat program yang membaca file `input.txt`, kemudian menghitung jumlah kata dan menyimpannya ke `output.txt`.

Langkah Penyelesaian

1. Baca isi file.
2. Hitung jumlah kata.
3. Tulis hasil ke file baru.

Contoh Penyelesaian

```
with open("input.txt", "r") as f:
    teks = f.read()

jumlah_kata = len(teks.split())

with open("output.txt", "w") as f:
    f.write(f"Jumlah kata: {jumlah_kata}")
```

Kasus 2 — Membaca & Menulis File CSV

Deskripsi

Anda diberi file CSV berisi daftar nilai mahasiswa: nama dan nilai. Buat program yang membaca file tersebut kemudian membuat file baru yang hanya berisi mahasiswa dengan nilai di atas 75.

Contoh CSV

```
Nama,Nilai
Budi,80
Siti,70
Rina,90
```

Contoh Penyelesaian

```
import csv

filtered = []

with open("nilai.csv", "r") as f:
    reader = csv.DictReader(f)
```

```
    for row in reader:
        if int(row["Nilai"]) > 75:
            filtered.append(row)

with open("lulus.csv", "w", newline="") as f:
    writer = csv.DictWriter(f, fieldnames=["Nama", "Nilai"])
    writer.writeheader()
    writer.writerows(filtered)
```

Kasus 3 — Mengolah File JSON

Deskripsi

Terdapat file `user.json` berisi data pengguna. Buat program untuk menambah field baru: `status: "active"` untuk setiap pengguna.

Contoh JSON

```
[
    {"nama": "Andi", "umur": 25},
    {"nama": "Budi", "umur": 30}
]
```

Contoh Penyelesaian

```
import json

with open("user.json", "r") as f:
    data = json.load(f)

for user in data:
    user["status"] = "active"

with open("user_updated.json", "w") as f:
    json.dump(data, f, indent=4)
```

Kasus 4 — Log Analyzer Sederhana

Deskripsi

Anda memiliki file log aplikasi dengan format:

```
[INFO] Aplikasi dimulai
[ERROR] Koneksi gagal
[INFO] Pengguna login
[ERROR] Timeout server
```

Buat program untuk menghitung berapa banyak error yang muncul dan simpan hasilnya ke `summary.txt`.

Contoh Penyelesaian

```
error_count = 0

with open("app.log", "r") as f:
    for line in f:
        if "ERROR" in line:
            error_count += 1

with open("summary.txt", "w") as f:
    f.write(f"Jumlah ERROR: {error_count}")
```

Kasus 5 — Mencari Teks dalam File Teks

Deskripsi

Buat program untuk mencari kemunculan suatu kata pada file `dokumen.txt` dan menampilkan baris mana saja yang mengandung kata tersebut.

Contoh Penyelesaian

```
kata_cari = "python"

with open("dokumen.txt", "r") as f:
    for nomor, baris in enumerate(f, start=1):
        if kata_cari.lower() in baris.lower():
            print(f"Baris {nomor}: {baris.strip()}")
```

Tugas Mandiri

1. Buat program yang mengekstrak semua email dari sebuah file teks.
2. Buat program yang menghitung total nilai dari file CSV berisi transaksi.
3. Buat aplikasi kecil yang menyimpan data pengguna ke file JSON setiap kali program dijalankan.

Penutup

Dengan menguasai dasar pengelolaan file, mahasiswa dapat melakukan berbagai proses otomatisasi data, pencatatan, dan analisis sederhana menggunakan Python. Pada praktikum lanjutan, teknik ini dapat dikembangkan untuk bekerja dengan file Excel, database, dan data berskala besar.