


<p>Nama: Aini Rihhadatul Aisy</p> <p>NIM: 064102400024</p>	 <p>Praktikum Algoritma & Pemrograman</p>	<p>MODUL 9</p> <p>Nama Dosen: Binti solihah, S.T, M.KOM</p>
<p>Hari/Tanggal: Jumat, 22 November 2024</p>		<p>Nama Asisten Labratorium:</p> <ol style="list-style-type: none"> 1. Yustianas Rombon - 064002300015 2. Vira Aditya Kurniawan - 065002300012

File Handling

1. Teori Singkat

File Handling

File Handling merupakan sebuah istilah penanganan file dalam Bahasa Pemrograman. Dalam konsep file handling sendiri kita dapat membuka, menutup, membaca, menulis, menambahkan, serta mengcopy file. Python sendiri dalam memperlakukan beberapa file berbeda, entah itu dalam bentuk biner ataupun sebagai teks. Dalam mengimplementasikannya pada Bahasa pemrograman python sendiri, pemanggilanya, sintaksnya adalah: `file = open('nama file', 'mode')`. Ada tiga jenis mode, yang disediakan Python dan bagaimana file dapat dibuka:

"r", untuk membaca.

"w", untuk menulis.

"a", untuk menambahkan.

"r+", untuk membaca dan menulis

Dan jangan lupa jika kita telah selesai melakukan sesuatu pada file tersebut kita harus memanggil method untuk menutup file tersebut. Pemanggilan methodnya sendiri adalah dengan `file.close()`

Berikut Contoh programnya



```

# Write File
file = open("Test.txt", "w")
file.write("Nadiya Amanda Rizkania")
file.close()

# Read File
file = open("Test.txt", "r")
text = file.read()
print(text)
file.close()

# Add Text
file = open("Test.txt", "a")
file.write(" - 20 Tahun - Jakarta")
file.close()

# Read & Write File
file = open("Test.txt", "r+")
text = file.read()
print(text)
file.close()

```

2. Alat dan Bahan

Hardware : Laptop/PC



Software : Spyder (Anaconda Python)

3. Elemen Kompetensi

a. Latihan pertama

Buatlah sebuah Text File dengan nama Biodata.txt menggunakan implelementasi File Handling dengan output seperti dibawah ini yang diinputkan oleh user:

Nama: Nama Kalian

Umur: Umur Kalian

Alamat: Alamat Kalian

Email: Email Kalian

Dosen Wali: Dosen Wali Kalian

Masukkan kedua metode tulis dan metode baca kedalam fungsi agar program lebih terstruktur.

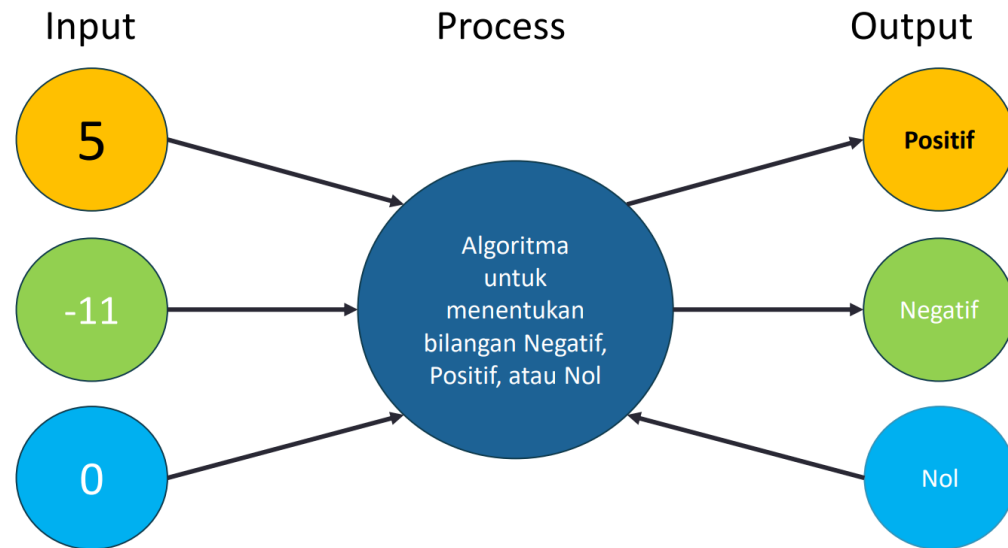
IPO (Input Process Output)

Konsep Dasar Input, Process, dan Output (IPO)

- Konsep input, process, dan output adalah prinsip dasar dalam pemrograman dan pengembangan algoritma.
- Setiap algoritma melibatkan tiga tahap utama: mengambil data masukan (input), melakukan operasi atau pengolahan data (process), dan menghasilkan hasil akhir (output).
- Konsep ini menggambarkan bagaimana algoritma beroperasi untuk memproses informasi.



Gambaran IPO (Menentukan Bilangan)



Notasi Algoritma Flowchart


1. Flowchart adalah representasi visual atau diagram alir yang digunakan untuk menggambarkan langkahlangkah dan urutan proses suatu algoritma atau program.
2. Flowchart menyajikan langkah-langkah dalam bentuk simbol-simbol grafis yang saling terhubung, membantu dalam memvisualisasikan bagaimana informasi mengalir dan bagaimana proses dilakukan.
3. Dalam kaitannya dengan notasi deskriptif, notasi algoritma yang menggunakan flowchart dapat lebih cepat dibaca dan dilihat alur dan hubungannya.


Simbol-simbol pada Flowchart


1. Setiap elemen flowchart dihubungkan oleh garis aliran bertanda panah
2. Garis aliran dimulai dari atas symbol dan keluar dari bagian bawah, kecuali symbol keputusan yang alirannya keluar dari bawah atau samping
3. Aliran bergerak dari atas ke bawah
4. Proses awal dan akhir menggunakan symbol terminal.




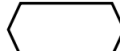
... Simbol-simbol pada Flowchart


 **Terminator** yang menandakan *Start* (awal) atau *End* (akhir) program.


 **Flow line** yang digunakan untuk menunjukkan arah aliran pada program.

 **Process** menunjukkan proses yang dilakukan pada masukan.

 **Input atau output** untuk menunjukkan masukan dan keluaran.

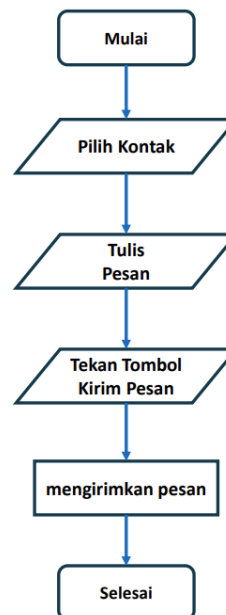
 **Preparation** digunakan untuk membuat deklarasi nilai awal.

 **On Page Connector** digunakan untuk menghubungkan antar *flowchart*

 **Decision** menunjukkan keputusan atau kondisi untuk memilih keputusan.

Contoh sederhana
Penggunaan *flowchart*
untuk menunjukan algoritma

Kasus/Aliran:
Mengirim pesan WhatsApp



IPO



INPUT

- nama
- umur
- alamat
- email
- Dosen wali

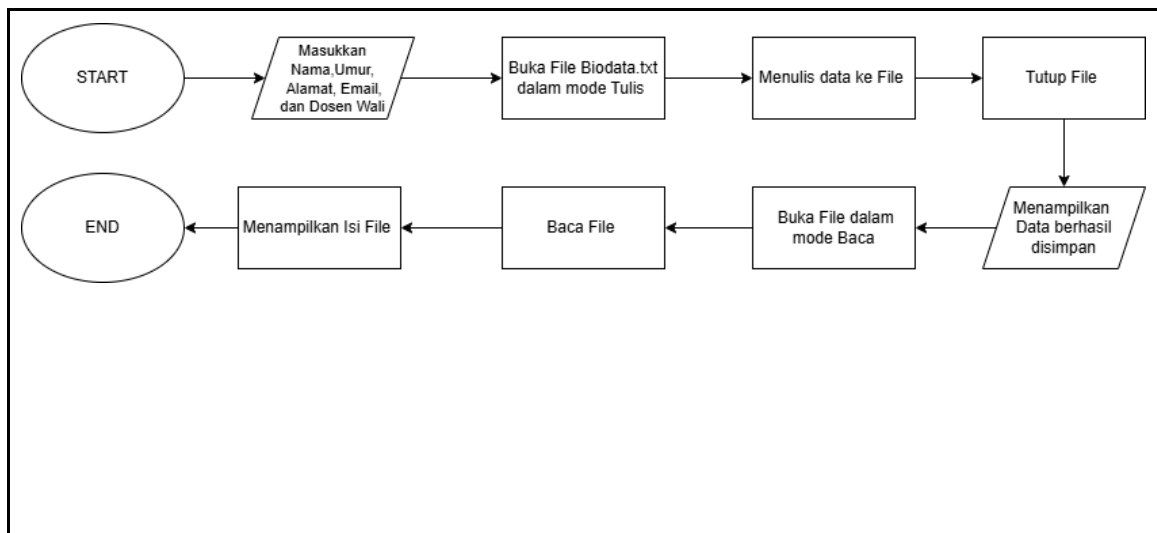
Proses

- Membuka Biodata.txt menggunakan mode “w” / tulis
- Menulis data data yang diinput
- menutup file
- Membuka Biodata.txt menggunakan mode “r” / baca
- Membaca file
- menutup file

Output

- Menampilkan Pesan “Data berhasil disimpan”
- Menampilkan isi dari file Biodata.txt

Flowchart



Source Code



```

▶ nama = input("Nama: ")
umur = input("Umur: ")
alamat = input("Alamat: ")
email = input("Email: ")
dosen_wali = input("Dosen Wali: ")

file = open("Biodata.txt", "w")
file.write(f>Nama: {nama}\n")
file.write(f">Umur: {umur}\n")
file.write(f">Alamat: {alamat}\n")
file.write(f">Email: {email}\n")
file.write(f">Dosen Wali: {dosen_wali}\n")
file.close()
print("Data berhasil disimpan.")

file = open("Biodata.txt", "r")
data = file.read()
print(data)
file.close()

```

Output




```

➡ Nama: Aini
  Umur: 18
  Alamat: Bogor
  Email: ii@gmail.com
  Dosen Wali: bu syandra
  Data berhasil disimpan.
  Nama: Aini
  Umur: 18
  Alamat: Bogor
  Email: ii@gmail.com
  Dosen Wali: bu syandra
  
```

b. Latihan Kedua

Buatlah sebuah program dimana program tersebut dapat membuat file, membaca file dan menambahkan text ke dalam file yang dimana, nama file didapat dari hasil inputan user dan juga data yang ditambahkan kedalam file didapat dari inputan user, implementasikan program kedalam fungsi dan juga implementasikan percabangan serta perulangan pada program seperti yang diajarkan pada materi sebelumnya dimana jika pilihan menu close tidak dipilih (diinputkan oleh user) maka program akan terus berjalan.

IPO



Input:

- Nama file.
- Pilihan menu (1. Buat, 2. Baca, 3. Tambah, 4. Keluar).
- Menambahkan text kedalam file

Proses:

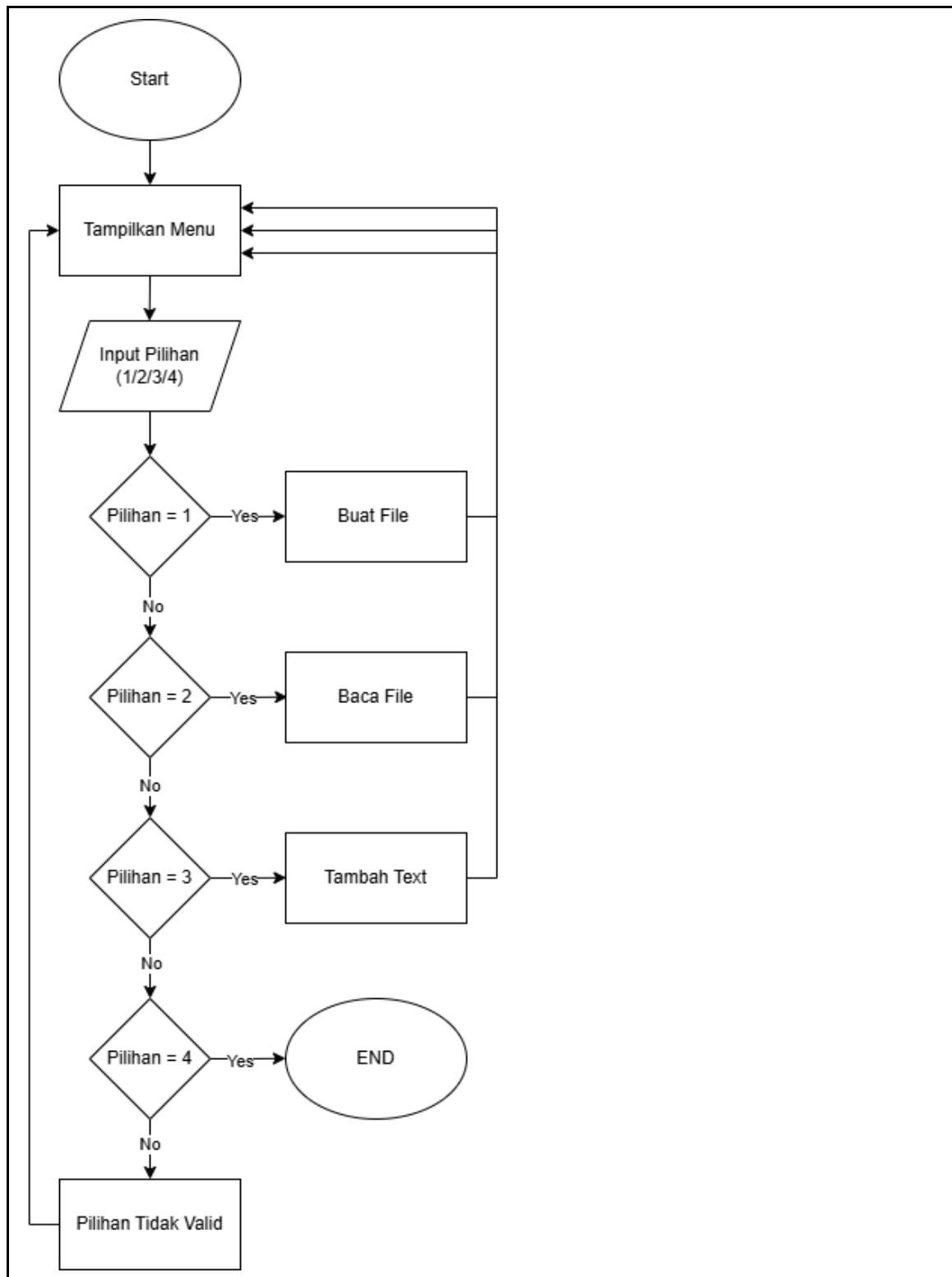
- Buat File: Membuat file kosong dengan nama yang diberikan.
- Baca File: Memeriksa keberadaan file, lalu membaca isinya jika ada.
- Tambah Teks: Menambahkan teks ke file jika file ada.

Output:

- Konfirmasi file yang berhasil dibuat
- Isi file yang dibaca atau pesan file tidak ditemukan.
- Konfirmasi penambahan teks.

Flowchart





Source Code

```
def file_ada(nama_file):
    """Memeriksa apakah file ada dengan mencoba membukanya."""
    try:
        open(nama_file).close()
        return True
    except:
        return False
def buat_file(nama_file):
    open(nama_file, 'w').close()
    print(f"File '{nama_file}' berhasil dibuat.")
def baca_file(nama_file):
    if file_ada(nama_file):
        with open(nama_file, 'r') as file:
            isi = file.read()
            print(f"Isi file '{nama_file}':\n{isi or '(Kosong)'}")
    else:
        print(f"File '{nama_file}' tidak ditemukan.")

while True:
    print("\nMenu: 1.Buat File 2.Baca File 3.Tambah Text 4.Keluar")
    p = input("Pilih (1/2/3/4): ")
    if p == '4':
        print("Program selesai."); break
    nama = input("Nama file: ")
    if p == '1': buat_file(nama)
    elif p == '2': baca_file(nama)
    elif p == '3': tambah_teks(nama, input("Teks: "))
    else: print("Pilihan tidak valid.")
```



Output

```
Menu: 1.Buat File 2.Baca File 3.Tambah Text 4.Keluar
Pilih (1/2/3/4): 1
Nama file: halo.txt
File 'halo.txt' berhasil dibuat.
```

```
Menu: 1.Buat File 2.Baca File 3.Tambah Text 4.Keluar
Pilih (1/2/3/4): 2
Nama file: halo.txt
Isi file 'halo.txt':
(Kosong)
```

```
Menu: 1.Buat File 2.Baca File 3.Tambah Text 4.Keluar
Pilih (1/2/3/4): 3
Nama file: halo.txt
Teks: haii namaku haloo
Teks ditambahkan ke 'halo.txt'.
```

```
Menu: 1.Buat File 2.Baca File 3.Tambah Text 4.Keluar
Pilih (1/2/3/4): 2
Nama file: halo.txt
Isi file 'halo.txt':
haii namaku haloo
```

```
Menu: 1.Buat File 2.Baca File 3.Tambah Text 4.Keluar
Pilih (1/2/3/4): 4
Program selesai.
```

4. File Praktikum

Github Repository:

5. Soal Latihan

Soal:

1. Apa saja kegunaan file handling dalam sebuah bahasa pemrograman dan kenapa file handling diperlukan dalam sebuah Bahasa pemrograman?



2. Deskripsikan serta narasikan jalannya alur source code program yang sebelumnya telah kalian buat pada Elemen Kompetensi Latihan Kedua!

Jawaban:

1. file handling adalah elemen penting dalam pengembangan perangkat lunak untuk memastikan aplikasi dapat menangani data secara efektif, menyimpan data secara permanen, dan berinteraksi dengan berbagai jenis file. Dan File handling memungkinkan aplikasi untuk berinteraksi dengan data secara efisien dan menyimpannya untuk penggunaan jangka panjang.
2. Program yang dibuat memungkinkan untuk:
Membuat file baru.
Membaca isi file.
Menambahkan teks ke file yang sudah ada.

Alur Program:

Fungsi file_ada(nama_file): Memeriksa apakah file ada dengan mencoba membukanya.

Fungsi buat_file(nama_file): Membuat file baru jika belum ada.

Fungsi baca_file(nama_file): Membaca dan menampilkan isi file jika ada.

Fungsi tambah_teks(nama_file, teks): Menambahkan teks ke file jika ada.

Menu Utama:

Program menawarkan menu dengan pilihan:

- 1: Buat file baru.
- 2: Baca isi file.
- 3: Tambah teks ke file.
- 4: Keluar dari program.

Program memeriksa apakah file ada sebelum melakukan operasi baca atau tambah. Jika file tidak ada, pengguna diberi pesan kesalahan. Program akan mengulang menu hingga pengguna memilih untuk keluar.

6. Kesimpulan

- a. Dalam pengerjaan program dengan bahasa pemrograman Python, kita harus benar-benar teliti dalam menginputkan suatu fungsi untuk menampilkan suatu keluaran pada layar dengan sesuai.
- b. Kita dapat mengetahui pentingnya mengelola file dengan berbagai mode seperti 'r', 'w', 'a', dan 'r+'. Fungs-fungsi seperti open(), close(), write(), read(), dan append() membantu kita menangani dan menyimpan data dengan efisien. Latihan ini juga meningkatkan pemahaman kita tentang alur program dan struktur file handling dalam aplikasi.

7. Cek List (✓)



No	Elemen Kompetensi	Penyelesaian	
		Selesai	Tidak Selesai
1.	Latihan Pertama	✓	
2.	Latihan Kedua	✓	

8. Formulir Umpan Balik

No	Elemen Kompetensi	Waktu Pengerjaan	Kriteria
1.	Latihan Pertama	20 Menit	Menarik
2.	Latihan Kedua	30 Menit	Menarik

Keterangan:

1. Menarik
2. Baik
3. Cukup
4. Kurang