

LAPORAN PRAKTIKUM

PEMROGRAMAN KOMPUTER (PYTHON)



Disusun Oleh:
Rafli Pratama
H1101241008

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS TANJUNGPURA PONTIANAK
2024

Praktikum 8

Penanganan File

1.1. Dasar Teori

1.1.1. File Descriptor

File descriptor adalah sebuah integer unik yang diberikan oleh sistem operasi setiap kali file dibuka. File descriptor merepresentasikan sebuah referensi ke file atau perangkat lain (misalnya, *socket* atau *pipe*) dan digunakan oleh sistem operasi untuk mengidentifikasi dan mengelola file tersebut. Dalam Python, file descriptor dapat diakses dengan properti `fileno()` dari objek file. Contoh penggunaan:



```
index.py
index.py > ...
1 fileSaya = open('pertamaFix.txt', 'r')
2 print(fileSaya.fileno()) #Output nya adalah 3
```

1.1.2 Buffering

Buffering adalah mekanisme untuk mengelola aliran data antara memori dan disk dengan cara menyimpan data sementara di dalam *buffer* (memori sementara). Tujuan buffering adalah untuk meningkatkan efisiensi I/O karena akses ke memori lebih cepat dibandingkan akses langsung ke disk.

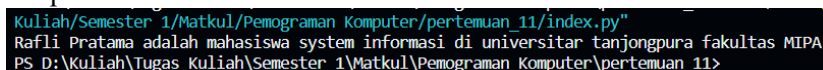
Buffering dapat diatur dengan parameter ketiga dari fungsi `open()`. Nilai defaultnya -1 (otomatis) yang menentukan tingkat buffering tergantung pada file dan sistem operasi. Terdapat tiga mode buffering utama:

- Unbuffered (0): Data ditulis atau dibaca langsung dari disk, tanpa penyimpanan di buffer.
- Line-buffered (1): Data di-buffer satu baris per satu waktu (umumnya untuk file teks).
- Fully buffered (>1 atau -1): Data di-buffer dengan ukuran blok tertentu (defaultnya sistem).



```
index.py
index.py > ...
1 fileSaya = open('pertamaFix.txt', 'r', buffering=1024)
2 print(fileSaya.read())
3
4
```

Output :



```
Kuliah/Semester 1/Matkul/Pemograman Komputer/pertemuan_11/index.py"
Rafli Pratama adalah mahasiswa system informasi di universitas tanjongpura fakultas MIPA
PS D:\Kuliah\Tugas Kuliah\Semester 1\Matkul\Pemograman Komputer\pertemuan_11>
```

1.1.3 Mode File

File mode menentukan cara membuka dan mengakses file. Python mendukung beberapa mode yang digunakan sebagai parameter kedua dari fungsi `open()`:

- "r": *Read* - membuka file hanya untuk membaca.

- "w": *Write* - membuka file untuk menulis, menghapus konten jika file sudah ada.
- "a": *Append* - membuka file untuk menulis di akhir konten tanpa menghapus konten yang ada.
- "b": *Binary* - membuka file dalam mode biner, biasanya dikombinasikan dengan mode lain seperti "rb" atau "wb".
- "+": membuka file untuk *updating* (membaca dan menulis).

Contoh :

```
fileSaya = open('pertamaFix.txt','r')
print(fileSaya)
✓ 0.0s
<_io.TextIOWrapper name='pertamaFix.txt' mode='r' encoding='cp1252'>

fileSaya = open('pertamaFix.txt',mode='w')
print(fileSaya)
✓ 0.0s
<_io.TextIOWrapper name='pertamaFix.txt' mode='w' encoding='cp1252'>
```

1.1.4 Membuka File

Fungsi open() digunakan untuk membuka file dalam mode tertentu. Fungsi ini memiliki dua parameter utama: nama file dan mode akses file.

Berikut berbagai mode yang bisa digunakan dalam open():

- **'r' (Read)**: Membuka file hanya untuk dibaca. File harus sudah ada, dan pointer berada di awal file.
- **'w' (Write)**: Membuka file hanya untuk menulis. Jika file sudah ada, isinya akan dihapus; jika tidak, file baru akan dibuat.
- **'a' (Append)**: Membuka file untuk menulis di akhir file tanpa menghapus isinya. Jika file belum ada, maka akan dibuat.
- **'b' (Binary)**: Digunakan bersamaan dengan mode lain ('rb', 'wb', dll.) untuk membuka file dalam mode biner (bukan teks), yang biasanya digunakan untuk file non-teks seperti gambar atau video.

1. Fungsi read, readline, dan ridelines

- **read()**: Membaca keseluruhan file atau sejumlah karakter tertentu. Jika size tidak diatur, read() akan membaca semua konten.

```
fileSaya = open('pertamaFix.txt','r')
print(fileSaya.read())
✓ 0.0s
Rafli Pratama
Timothy Walukow
Athallah Rizky

fileSaya = open('pertamaFix.txt','r')
print(fileSaya.read(27))
✓ 0.0s
Rafli Pratama
Timothy Waluk
```

- **readline()**: Membaca satu baris dalam file. Setiap panggilan readline() mengembalikan baris berikutnya.

```
fileSaya = open('pertamaFix.txt','r')
print(fileSaya.readline())
print(fileSaya.readline())
```

✓ 0.0s

Rafli Pratama

Timothy Walukow

- **readlines():** Membaca semua baris dalam file dan mengembalikannya sebagai list di mana setiap elemen adalah satu baris.

```
fileSaya = open('pertamaFix.txt','r')
print(fileSaya.readlines())
fileSaya.close()
```

✓ 0.0s

['Rafli Pratama\n', 'Timothy Walukow\n', 'Athallah Rizky']

2. Fungsi Write dan Writelines

- **write():** Menulis teks ke file. Jika file sudah ada dalam mode "w", semua konten lama akan dihapus. Untuk menambahkan teks tanpa menghapus, gunakan mode "a".

```
fileSaya = open('pertamaFix.txt', mode='w')
fileSaya.write('Rafli Pratama\n')
fileSaya.write('Timothy Walukow')
fileSaya.close()
```

```
fileSaya = open('pertamaFix.txt', mode='r')
content = fileSaya.read()
print(content)
```

✓ 0.0s

Rafli Pratama

Timothy Walukow

- **writelines():** Menulis daftar string ke file. Setiap elemen dalam daftar ditulis tanpa karakter baris baru kecuali jika ditambahkan secara eksplisit.

```
fileSaya = open('pertamaFix.txt', mode='w')
fileSaya.writelines(['Rafli Pratama\n', 'Timothy Walukow'])
fileSaya.close()
```

```
fileSaya = open('pertamaFix.txt', mode='r')
content = fileSaya.read()
print(content)
```

✓ 0.0s

Rafli Pratama

Timothy Walukow

3. Append

- Mode "a" atau **append** dalam fungsi open() di Python digunakan untuk menulis data ke akhir file tanpa menghapus konten yang sudah ada. Mode ini dapat digunakan jika ingin menambahkan data baru tanpa memengaruhi data lama dalam file.

```
fileSaya = open('pertamaFix.txt', mode='a')
fileSaya.write('\nAthallah Rizky')
fileSaya.close()

fileSaya = open('pertamaFix.txt', mode='r')
content = fileSaya.read()
print(content)
```

✓ 0.0s

Rafli Pratama
Timothy Walukow
Athallah Rizky

4. Fungsi Close()

Fungsi `close()` di Python digunakan untuk menutup file yang telah dibuka, baik untuk membaca, menulis, atau menambah data, sehingga memastikan semua data tersimpan dengan benar serta melepaskan sumber daya yang digunakan oleh file tersebut. Menutup file sangat penting karena membebaskan memori, mencegah kebocoran sumber daya, dan memastikan bahwa data yang masih berada di *buffer* disimpan secara permanen ke file. Jika `close()` tidak dipanggil, hal ini bisa menyebabkan error atau bahkan kehilangan data yang belum tersimpan.

```
fileSaya = open('pertamaFix.txt', mode='a')
fileSaya.write('\nAthallah Rizky')
fileSaya.close()

fileSaya = open('pertamaFix.txt', mode='r')
content = fileSaya.read()
print(content)
fileSaya.close()
```

✓ 0.0s

Rafli Pratama
Timothy Walukow
Athallah Rizky

Karena fungsi `close` kita dapat melanjutkan program dengan mengubah mode `listSaya` yang sebelumnya `w` menjadi `a` tanpa mengalami error, lain hal jika kita tidak menggunakan fungsi `close`, program akan error atau bahkan mengalami bug dan resiko kehilangan data apabila data yang di buka banyak

1.1.5 With

`with` adalah pernyataan yang digunakan untuk menangani file atau sumber daya lain dengan cara yang lebih aman dan efisien. Saat bekerja dengan file, `with` membantu membuka dan menutup file secara otomatis setelah blok kode selesai dijalankan, sehingga kita tidak perlu memanggil `close()` secara manual. Jika terjadi error atau pengecualian di dalam blok `with`, Python tetap akan memastikan bahwa file ditutup dengan benar. Menggunakan `with` adalah praktik terbaik saat bekerja dengan file, karena mencegah potensi kebocoran sumber daya dan membuat kode lebih bersih serta lebih mudah dibaca.

Mengapa with sangat penting :

1. **Pengelolaan Sumber Daya yang Aman:** with memastikan bahwa sumber daya, seperti file, ditutup secara otomatis setelah digunakan. Ini mengurangi risiko kebocoran sumber daya yang dapat terjadi jika file tidak ditutup dengan benar.
2. **Penanganan Error:** Jika terjadi kesalahan atau pengecualian di dalam blok with, Python tetap akan menutup sumber daya tersebut. Ini membantu mencegah situasi di mana file tetap terbuka meskipun ada masalah.
3. **Kode yang Lebih Bersih dan Jelas:** Menggunakan with membuat kode lebih ringkas dan mudah dibaca. Hal ini menghilangkan kebutuhan untuk menulis kode penutupan terpisah, sehingga mengurangi kompleksitas.
4. **Meminimalkan Kesalahan Manusia:** Dengan otomatisasi proses penutupan, with membantu mengurangi kemungkinan kesalahan manusia, seperti lupa memanggil close() setelah selesai bekerja dengan file.

```
with open('pertamaFix.txt','r') as file:
    content = file.read()
    print(content)
    print(f'Apakah file sudah di close? : {file.closed}')
    print(f'Apakah file sudah di close? : {file.closed}')
```

✓ 0.0s

Rafli Pratama
Timothy Walukow
Athallah Rizky
Apakah file sudah di close? : False
Apakah file sudah di close? : True

1.2. Percobaan Praktikum

1.2.1. Soal 1

Siapkan sebuah file txt dan isikan teks ke dalamnya. Kemudian tampilkan isi teks menggunakan bahasa python.

The image shows a Python IDE with two windows. The top window, titled 'index.py', contains the following code:

```
1 d = open('Soal-1/data.txt', 'w')
2 d.write('Halo\n')
3 d.write('Teks ini akan muncul di file ini')
4 d.close
5
6 d= open('Soal-1/data.txt', 'r')
7 print(d.read())
```

Below the code editor is a terminal window with tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL'. The 'TERMINAL' tab is active, showing the command prompt 'PS D:\Kuliah\Tugas Kuliah\Semester 1\Matkul\Praktikum Pemograman' and the output of the program:

```
Halo
Teks ini akan muncul di file ini
```

The bottom window, titled 'index.py', shows the file 'data.txt' open in the editor. The content of the file is:

```
1 Halo
2 Teks ini akan muncul di file ini
```

Penjelasan :

Dalam program tersebut kita menggunakan fungsi `open()` yang merupakan salah satu fungsi yang di gunakan untuk penanganan file pada python, pada baris pertama kita membuka file pada folder Soal-1 dengan nama file data.txt dan kita membuka nya dengan mode 'w' atau write, sehingga kita dapat menambahkan konten ke dalam file tersebut, lalu kita mengclose itu menyimpan datanya dan kita buka kembali dengan mode 'r' atau read yang kita gunakan untuk mengprint konten yang ada di dalam file data.txt.

1.2.1. Soal 2

Buat program sederhana pendaftaran mahasiswa. Identifikasi field yang diperlukan. Setelah proses pendaftaran selesai masukkan data mahasiswa tersebut kedalam sebuah file txt.

```
index.ipynb \ • data2.txt test.txt data1.txt index.ipynb Tes
index.ipynb > def tulisdata():
+ Code + Markdown ▶ Run All ≡ Clear All Outputs ↺ Restart Variables Outline ...
def tulisdata():
    nama = input('Masukkan nama calon mahasiswa : ')
    umur = int(input('Umur Calon Mahasiswa'))
    asal = input('Asal daerah calon mahasiswa')
    asalSekolah = input('Masukkan asal sekolah calon Mahasiswa : ')
    data = f'Nama Mahasiswa : {nama}\nUmur : {umur}\nAsal daerah : {asal}\nasal sekolah : {asalSekolah}\n'
    f = open("dataBase/data2.txt", 'a')
    f.write(data)
    f.close()

def bacaData():
    f = open("dataBase/data2.txt", 'r')
    s = f.read()
    print(s)

while True :
    pilih = int(input('Anda ingin apa ? 1. Tambah Data 2. Baca data 3. Keluar'))
    if pilih == 1:
        tulisdata()
    elif pilih == 2:
        bacaData()
    elif pilih == 3:
        break
    else:
        print('Silahkan Pilih yang benar')

[2] ✓ 3.8s

... Nama Mahasiswa : Rafli Pratama
Umur : 18
Asal daerah : Pontianak
asal sekolah : SMAN 2 Pontianak

Nama Mahasiswa : Rafli Tapi Beda Orang
Umur : 18
Asal daerah : Sanggau
asal sekolah : SMAN 1 Sanggau
```

```
index.ipynb \ • data2.txt × test.txt
dataBase > data2.txt
1 Nama Mahasiswa : Rafli Pratama
2 Umur : 18
3 Asal daerah : Pontianak
4 asal sekolah : SMAN 2 Pontianak
5
6 Nama Mahasiswa : Rafli Tapi Beda Orang
7 Umur : 18
8 Asal daerah : Sanggau
9 asal sekolah : SMAN 1 Sanggau
10
```

Penjelasan :

Dalam program kita membuat sebuah fungsi tulisData yang didalamnya berisi variabel yang menyimpan inputan dari user berupa data nama, umur, asal daerah, dan asal sekolah. Lalu kita membuat variabel data yang menyimpan data data yang user inputkan lalu kita membuka file data2.txt di folder dataBase dengan mode 'w' atau write lalu kita menulis variabel data ke dalam file data2.txt

Dan di bawahnya kita membuat fungsi baca data yang membuka file data2.txt dengan mode 'r' atau read lalu kita menampilkan isinya dan di akhir kita membuat perulangan while yang masing masing kondisi menjalankan fungsi fungsi yang telah kita buat

1.2.1. Soal 3

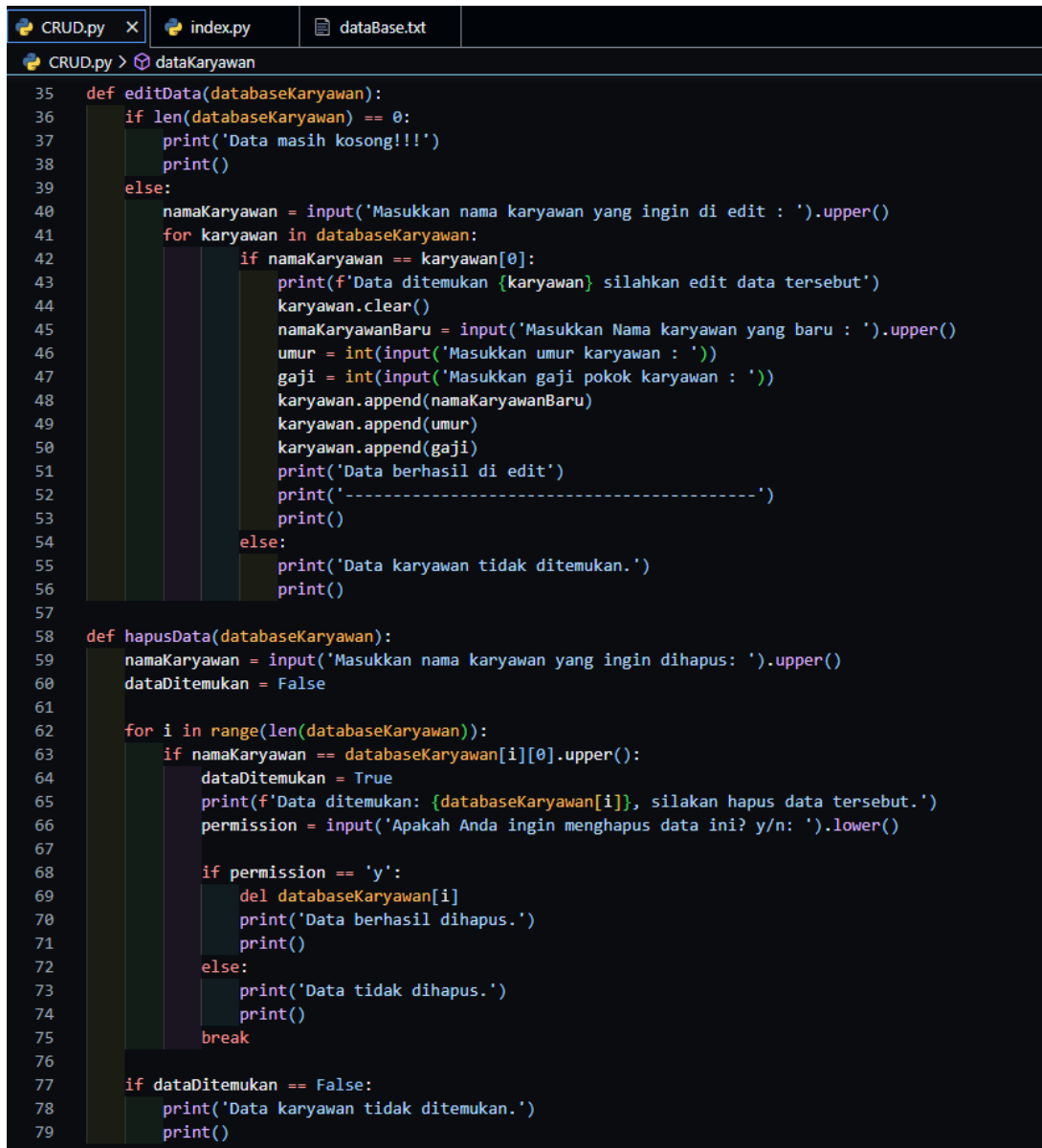
Gunakan file txt yang diberikan. Manipulasi data yang ada dan simpan dalam sebuah list. Kemudian buatlah program menu yang memiliki operasi dasar CRUD dari data yang sudah didapat.

File index :

```
CRUD.py index.py x dataBase.txt
index.py > ...
1 import CRUD
2 databaseKaryawan = []
3
4 print('Selamat datang di program CRUD Karyawan')
5 while True :
6     print('Silahkan pilih menu operasi yang anda inginkan : ')
7     menu = int(input('1. Membuat data karyawan\n2. Menampilkan database karyawan\n3. Mengupdate data karyawan\n'))
8     if menu == 1:
9         data = int(input('Berapa banyak karyawan yang ingin di daftarkan? '))
10        for i in range(data):
11            isiDb = CRUD.dataKaryawan()
12            databaseKaryawan.append(isiDb)
13            CRUD.updateDataBase(databaseKaryawan)
14        elif menu == 2:
15            CRUD.cetakDataKaryawan(databaseKaryawan)
16        elif menu == 3:
17            print('Untuk mengupdate data karyawan, silahkan masukkan nama karyawan!!!')
18            CRUD.editData(databaseKaryawan)
19            CRUD.updateDataBase(databaseKaryawan)
20        elif menu == 4:
21            print('Untuk menghapus data karyawan, silahkan masukkan nama karyawan !!!')
22            CRUD.hapusData(databaseKaryawan)
23            CRUD.updateDataBase(databaseKaryawan)
24        elif menu == 5:
25            print('Anda memilih keluar !!! Terima kasih')
26            CRUD.updateDataBase(databaseKaryawan)
27            break
28
29
CRUD.py x index.py dataBase.txt
CRUD.py > dataKaryawan
1 def dataKaryawan():
2     listKaryawan = []
3     def daftarKaryawan():
4         namaKaryawan = input('Masukkan Nama Karyawan : ').upper()
5         umur = int(input('Masukkan Umur Karyawan : '))
6         gaji = int(input('Masukkan gaji pokok karyawan : '))
7         return namaKaryawan, umur, gaji
8
9     def simpanDataKaryawan(namaKaryawan, umur, gaji):
10        listKaryawan.append(namaKaryawan)
11        listKaryawan.append(umur)
12        listKaryawan.append(gaji)
13        print('Data Karyawan berhasil di simpan')
14        print('-----')
15
16        namaKaryawan, umur, gaji= daftarKaryawan()
17        simpanDataKaryawan(namaKaryawan, umur, gaji)
18        return listKaryawan
19
20
21 def cetakDataKaryawan(databaseKaryawan):
22     jmldata = 3
23     print()
24     print('Berikut adalah data Karyawan: ')
25     print('-----')
26     print(f'|\tNama Karyawan\t|\tUmur Karyawan\t|\tGaji Karyawan\t|')
27     print('-----')
28     for i in range (len(databaseKaryawan)):
29         print(f'|\t {databaseKaryawan[i][jmldata-2]} \t|\t {databaseKaryawan[i][jmldata-2]} \t|\t {databaseKaryawan[i][jmldata-1]} \t|')
30         print('-----')
31     print()
32
33
```

Penjelasan :

Pada program ini akan mengimport modul yang telah di buat,lalu membuat list kosong untuk menyimpan semua data karyawan, lalu akan masuk ke dalam perulangan while True, jika user menginputkan 1, maka program akan meminta user memasukkan jumlah karyawan yang ingin di data dan membuat perulangan sesuai jumlah tersebut dan di dalam perulangan itu akan di jalankan fungsi dataKaryawan yang ada di modul Jika user menginputkan 2, maka fungsi [ada modul CRUD yaitu cetakDataKaryawan akan di jalankan, jika 3 maka program pada fungsi editData akan dijalankan, jika 4 maka fungsi hapusData akan di jalankan dan jika memilih 5 maka program akan keluar dari perulangan, didalam setiap kondisi akan dijalan fungsi updateDataBase yang menuliskan data di dalam dataBaseKaryawan ke dalam file dataBase.txt



```
35 def editData(databaseKaryawan):
36     if len(databaseKaryawan) == 0:
37         print('Data masih kosong!!!')
38         print()
39     else:
40         namaKaryawan = input('Masukkan nama karyawan yang ingin di edit : ').upper()
41         for karyawan in databaseKaryawan:
42             if namaKaryawan == karyawan[0]:
43                 print(f'Data ditemukan {karyawan} silahkan edit data tersebut')
44                 karyawan.clear()
45                 namaKaryawanBaru = input('Masukkan Nama karyawan yang baru : ').upper()
46                 umur = int(input('Masukkan umur karyawan : '))
47                 gaji = int(input('Masukkan gaji pokok karyawan : '))
48                 karyawan.append(namaKaryawanBaru)
49                 karyawan.append(umur)
50                 karyawan.append(gaji)
51                 print('Data berhasil di edit')
52                 print('-----')
53                 print()
54             else:
55                 print('Data karyawan tidak ditemukan.')
56                 print()
57
58 def hapusData(databaseKaryawan):
59     namaKaryawan = input('Masukkan nama karyawan yang ingin dihapus: ').upper()
60     dataDitemukan = False
61
62     for i in range(len(databaseKaryawan)):
63         if namaKaryawan == databaseKaryawan[i][0].upper():
64             dataDitemukan = True
65             print(f'Data ditemukan: {databaseKaryawan[i]}, silakan hapus data tersebut.')
66             permission = input('Apakah Anda ingin menghapus data ini? y/n: ').lower()
67
68             if permission == 'y':
69                 del databaseKaryawan[i]
70                 print('Data berhasil dihapus.')
71                 print()
72             else:
73                 print('Data tidak dihapus.')
74                 print()
75             break
76
77     if dataDitemukan == False:
78         print('Data karyawan tidak ditemukan.')
79         print()
80
```

```

81 def updateDataBase(databaseKaryawan):
82     with open('dataBase.txt', 'w') as database:
83         pass
84     for data in databaseKaryawan:
85         nama = data[0]
86         umur = data[1]
87         gaji = data[2]
88         with open('dataBase.txt', 'a') as database:
89             database.write(f'\nNama Karyawan : {nama}')
90             database.write(f'\nUmur Karyawan : {umur}')
91             database.write(f'\nGaji Pokok Karyawan : {gaji}')
92             database.write('\n')

```

Output :

```

Silahkan pilih menu operasi yang anda inginkan :
1. Membuat data karyawan
2. Menampilkan database karyawan
3. Mengupdate data karyawan
4. Menghapus data karyawan
5. Keluar
Pilihah anda : 1
Berapa banyak karyawan yang ingin di daftarkan? 2
Masukkan Nama Karyawan : Rafli
Masukkan Umur Karyawan : 20
Masukkan gaji pokok karyawan : 6000000
Data Karyawan berhasil di simpan
-----
Masukkan Nama Karyawan : Nabila
Masukkan Umur Karyawan : 19
Masukkan gaji pokok karyawan : 4000000
Data Karyawan berhasil di simpan
-----

```

```

Silahkan pilih menu operasi yang anda inginkan :
1. Membuat data karyawan
2. Menampilkan database karyawan
3. Mengupdate data karyawan
4. Menghapus data karyawan
5. Keluar
Pilihah anda : 2

```

Berikut adalah data Karyawan:

Nama Karyawan	Umur Karyawan	Gaji Karyawan
RAFLI	20	6000000
NABILA	19	4000000

```

Silahkan pilih menu operasi yang anda inginkan :
Data ditemukan ['RAFLI', 20, 6000000] silahkan edit data tersebut
Masukkan Nama karyawan yang baru : rafli
Masukkan umur karyawan : 21
Masukkan gaji pokok karyawan : 7000000
Data berhasil di edit
-----

```

Data karyawan tidak ditemukan.

```

Silahkan pilih menu operasi yang anda inginkan :
1. Membuat data karyawan
2. Menampilkan database karyawan
3. Mengupdate data karyawan
4. Menghapus data karyawan
5. Keluar
Pilihah anda : 2

```

Berikut adalah data Karyawan:

Nama Karyawan	Umur Karyawan	Gaji Karyawan
RAFLI	21	7000000
NABILA	19	4000000

```

Silahkan pilih menu operasi yang anda inginkan :

```

```

1. Membuat data karyawan
2. Menampilkan database karyawan
3. Mengupdate data karyawan
4. Menghapus data karyawan
5. Keluar
Pilihah anda : 4
Untuk menghapus data karyawan, silahkan masukkan nama karyawan !!!
Masukkan nama karyawan yang ingin dihapus: nabila
Data ditemukan: ['NABILA', 19, 4000000], silakan hapus data tersebut.
Apakah Anda ingin menghapus data ini? y/n: y
Data berhasil dihapus.

```

```

Silahkan pilih menu operasi yang anda inginkan :

```

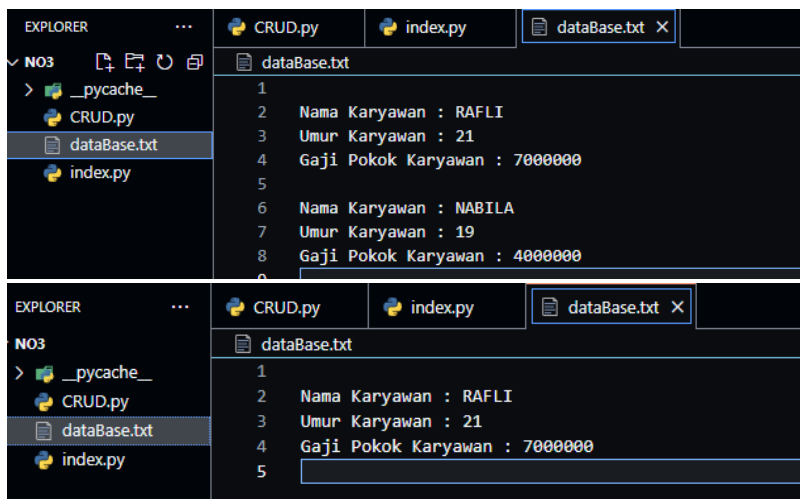
```

1. Membuat data karyawan
2. Menampilkan database karyawan
3. Mengupdate data karyawan
4. Menghapus data karyawan
5. Keluar
Pilihah anda : 2

```

Berikut adalah data Karyawan:

Nama Karyawan	Umur Karyawan	Gaji Karyawan
RAFLI	21	7000000



Penjelasan :

Pada modul CRUD, program akan membuat fungsi dataKaryawan yang berisi list kosong dan fungsi daftarKaryawan lalu di dalam nya terdapat inputan user yang berisi nama, umur, dan gaji pokok karyawan, dan mereturn nya, lalu dibuat juga fungsi simpanDataKaryawan yang menerima parameter umur dan gaji lalu meng append nya ke list kosong yang sudah di buat di atas setelah itu kita return list tersebut

pada modul CRUD didalamnya juga terdapat fungsi cetakDataKaryawan yang menerima parameter databaseKaryawan, setelah itu melakukan perulangan dan mencetak seluruh data yang ada di dalam list databaseKaryawan

Pada modul CRUD juga terdapat fungsi editData yang menerima parameter databaseKaryawan lalu di dalam nya kita melakukan pengecekan yang jika data di dalam list databaseKaryawan kosong maka data akan menampilkan bahwa data masih kosong, jika tidak user akan di minta memasukkan kata kunci berupa nama karyawan, jika ada nama karyawan yang sama dengan yang ada di dalam databaseKaryawan, maka data tersebut dapat di ganti dan di update

modul ini juga terdapat fungsiData yang cara kerja nya sama dengan fungsi editData di modul update, tetapi di akhir terdapat pilihan lagi sebagai konfirmasi ulang jika user memilih y, maka data akan di hapus dan selain itu data tidak akan di hapus.

Dan yang terakhir ada fungsi updateDataBase yang menerima parameter dataBaseKaryawan yang membuka file dataBase.txt dengan mode 'w' untuk melakukan clear dan membuka nya lagi dengan mode 'a' untuk menuliskan data yang terdapat dalam dataBaseKaryawan ke file dataBase.txt

1.3. Kesimpulan dan Saran

1.3.1. Kesimpulan

Dari praktikum ini dapat disimpulkan :

1. Python dapat melakukan handling file dengan menggunakan fungsi open() dengan beberapa mode yaitu read, write, dan append
2. Ketika kita membuka sebuah file dengan open() kita juga harus menutupnya dengan close()

3. Pada mode write, apabila file yang ingin di buka belum ada, maka file tersebut akan otomatis di buat
4. Write akan melakukan clear sebelum menulis kembali data, sedangkan append samas seperti write hanya saja tidak melakukan clear pada awal
5. With adalah metode paling aman karena dapat melakukan close file secara otomatis

1.3.2. Saran

Saran terhadap praktikum ini yaitu:

1. Pastikan kita menggunakan mode yang tepat ketika membuka sebuah file atau melakukan handling file
2. Ketika membuka file, kita di haruskan untuk menutupnya kembali dengan fungsi close() untuk mencegah error sekaligus menyimpan data tersebut
3. Anda bisa menggunakan with agar menghindari lupa menutup file, karena with akan menutup file secara otomatis kode yang di luar indentasi nya

DAFTAR PUSTAKA

<https://dqlab.id/alternatif-mode-operasi-python-untuk-handling-file>

Modul *Perkuliahan dan Praktikum Algoritma dan Pemrograman* Oleh Ilhamsyah, S.Si., M.Cs.