Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**<71230989>**
Nama Lengkap	**<Yohanes Nevan Adventus Wibawa>**
Minggu ke / Materi	**12 / Tipe Data Set**

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.

Pengenalan dan Mendefinisikan Set

Set adalah salah satu tipe data di Python yang digunakan untuk menyimpan kumpulan data unik, yang sering disebut sebagai himpunan. Berikut adalah beberapa karakteristik dari Set di Python:

- Elemen dalam Set disebut anggota.
- Anggota dari Set harus bersifat immutable. Beberapa tipe data immutable di Python meliputi integer, float, string, dan tuple. Oleh karena itu, list dan dictionary, yang bersifat mutable, tidak dapat dimasukkan ke dalam Set.
- Set sendiri bersifat mutable, artinya kita bisa menambah atau mengurangi elemen di dalamnya. Karena itu, Set tidak dapat menjadi elemen dari Set lain.

Untuk mendefinisikan Set, Anda bisa menggunakan notasi kurung kurawal {} atau fungsi set(), seperti pada contoh berikut ini:

```
# dengan menggunakan fungsi set()
bilangan_genap = {2, 4, 6, 8, 10, 12}
bilangan_ganjil = {1, 3, 5, 7, 9, 11}

# dengan menggunakan fungsi set()
pernah_ke_bulan = set('Neil Armstrong', 'Buzz Aldrin')

# dengan fungsi set()
pernah_ke_mars = set() # menghasilkan set kosong
data = {} # ini akan menghasilkan dictionary kosong
```

Pengaksesan Set

Set tidak memiliki indeks, sehingga Anda tidak bisa mengakses elemen-elemen dalam Set secara langsung menggunakan indeks. Berikut adalah contoh program yang menunjukkan hal ini:

```
nim = {'71200120', '71200195', '71200214'}
jumlah_nim = len(nim)
print(jumlah_nim) # akan menghasilkan output 3
# tampilkan isi set satu-persatu
for n in nim:
    print(n)
```

Output =

```
3
71200120
71200214
71200195
```

Jika diperhatikan, urutan output yang dihasilkan berbeda dari deklarasi Set sebelumnya. Hal ini terjadi karena Set tidak memiliki indeks, sehingga tidak ada urutan posisi elemen. Pada tipe data Set, posisi elemen tidaklah penting. Set adalah tipe data yang mutable, artinya elemen-elemennya bisa ditambah atau dikurangi. Program berikut ini adalah cara menambahkan elemen ke dalam sebuah Set menggunakan fungsi add():

```
# definisikan sebuah set kosong
plat_nomor = set()
# tambahkan plat nomor 'AB 1890 XA'
plat_nomor.add('AB 1890 XA')
# tambahkan plat nomor 'AD 6810 MT'
plat_nomor.add('AD 6810 MT')
# jumlah anggota di dalam Set
print(len(plat_nomor))
# tambahkan plat yang sama sekali lagi
plat_nomor.add('AB 1890 XA')
# tampilkan semua plat nomor
for plat in plat_nomor:
    print(plat)
```

Outout =

```
2
AD 6810 MT
AB 1890 XA
```

Set memiliki mekanisme untuk memeriksa apakah elemen baru yang akan ditambahkan sudah ada dalam Set (pengecekan duplikasi). Jika elemen tersebut belum ada, maka elemen tersebut akan dimasukkan ke dalam Set. Namun, jika elemen dengan nilai yang sama sudah ada dalam Set, pemanggilan fungsi add() tidak akan menambah elemen tersebut. Pengecekan duplikasi ini sudah diurus oleh fungsi add(), sehingga Anda tidak perlu melakukannya secara manual.

Untuk menghapus elemen dari sebuah Set, ada beberapa metode yang dapat digunakan, yaitu fungsi discard(), remove(), pop(), dan clear(). Perbedaan antara keempat fungsi tersebut dapat dilihat pada dibawah ini. Berikut ini adalah contoh program yang menunjukkan cara menghapus elemen dari sebuah Set:

discard()	remove()	pop()	clear()
Menghapus satu elemen yang disebutkan	Menghapus satu elemen yang disebutkan	Mengambil salah satu dan menghapusnya dari set (tidak tentu)	Menghapus seluruh elemen di dalam set
Tidak ada error	Muncul error jika elemen yang dihapus tidak ada	Error jika set kosong	Tidak ada error

```python
bilangan_prima = {13, 23, 7, 29, 11, 5}
# hapus 5 dari set tersebut
bilangan_prima.remove(5)
print(bilangan_prima)

# hapus 97 (tidak ada)
bilangan_prima.discard(97)
print(bilangan_prima)

# ambil dan hapus salah satu
bilangan = bilangan_prima.pop()
print(bilangan)
print(bilangan_prima)

# kosongkan set
bilangan_prima.clear()
print(bilangan_prima)
```

Output =
```
{29, 23, 7, 11, 13}
{29, 23, 7, 11, 13}
29
{23, 7, 11, 13}
set()
```

Fungsi discard() tidak akan menyebabkan error jika elemen yang ingin dihapus tidak ada dalam Set. Sebaliknya, fungsi pop() akan menghapus dan mengembalikan salah satu elemen secara acak dari Set. Fungsi pop() berguna jika ingin memproses elemen-elemen dalam Set satu per satu tanpa memperhatikan urutan atau posisinya.

```python
# Buat Set dari List
ikan = set(['koi', 'koki', 'kembung', 'salmon'])
print(ikan)
# ganti koi menjadi teri
ikan.remove('koi')
ikan.add('teri')
print(ikan)
```

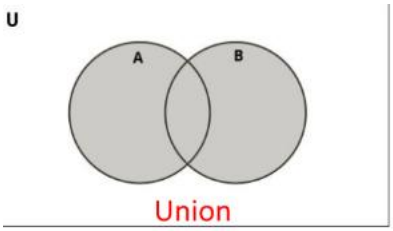
Output =
```
{'kembung', 'koi', 'koki', 'salmon'}
{'kembung', 'teri', 'koki', 'salmon'}
```

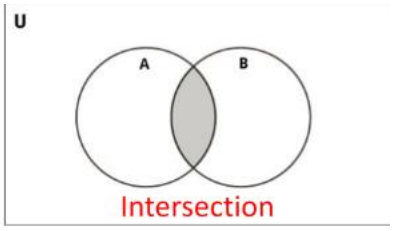
Untuk mengubah nilai 'koi' menjadi 'teri', langkah pertama adalah menghapus elemen 'koi' dari Set, kemudian menambahkan elemen baru dengan nilai 'teri'. Perlu dicatat bahwa dalam Set, tidak ada konsep posisi atau urutan data. Jadi, setelah perubahan dilakukan, 'koi' sudah tidak ada dalam Set dan telah digantikan oleh 'teri'. Selain itu, urutan elemen dalam Set mungkin akan berubah setiap kali ada operasi penambahan atau penghapusan.

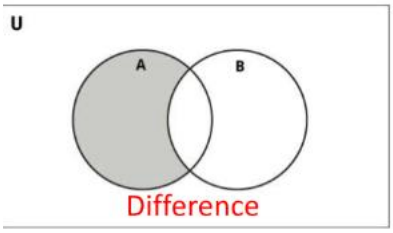
Operasi-Operasi pada Set

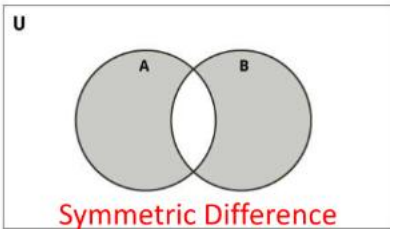
Operasi-operasi pada Set melibatkan operasi-operasi pada himpunan. Berikut adalah daftar operasi Set di Python:

- **Operator Union:** Menggabungkan dua Set menjadi satu. Bisa menggunakan operator `|` atau fungsi `union()`.
- **Operator Intersection:** Menghasilkan irisan dari dua Set. Bisa menggunakan operator `&` atau fungsi `intersection()`.
- **Operator Difference:** Menghasilkan Set baru yang merupakan selisih dari dua Set. Bisa menggunakan operator `-` atau fungsi `difference()`.
- **Operator Symmetric Difference:** Menghasilkan Set baru yang merupakan gabungan dari dua Set, kecuali elemen yang ada di keduanya. Bisa menggunakan operator `^` atau fungsi `symmetric_difference()`.









Ilustrasi dari operasi-operasi tersebut dapat dilihat di bawah ini:

Operator Union =

```
merek_hp = {'Samsung', 'Apple', 'Xiaomi', 'Sony'}
merek_ac = {'LG', 'Samsung', 'Panasonic', 'Daikin', 'Sony'}
# union dari merek_hp dan merek_ac
gabungan = merek_hp | merek_ac
print(gabungan)
```

Output =

```
{'Samsung', 'Sony', 'Daikin', 'Xiaomi', 'LG', 'Apple', 'Panasonic'}
PS C:\Users\LENOVO\Tugas PrakAlPro\Tugas 12>
```

Output yang dihasilkan dari program tersebut adalah gabungan dari elemen-elemen dalam Set merek_hp dan Set merek_ac, menghasilkan sebuah Set baru bernama gabungan. Karena ada elemen yang sama, yaitu 'Samsung' dan 'Sony', elemen-elemen ini hanya muncul satu kali dalam Set gabungan, karena setiap elemen dalam Set harus unik.

Operator Intersection =

```
renang = {'siti', 'mail', 'ikhsan', 'upin', 'ipin'}
tenis = {'joko', 'mail', 'ipin', 'upin', 'tejo'}
# suka renang dan tenis
renang_tenis = renang & tenis
print(renang_tenis)
```

Output =

```
{'ipin', 'mail', 'upin'}
PS C:\Users\LENOVO\Tugas
```

Program tersebut akan menghasilkan output berupa hasil operasi intersection dari Set renang dan Set tenis. Intersection ini berarti elemen-elemen yang ada di dalam kedua Set tersebut sekaligus, yaitu ipin, mail, dan upin.

Operator Difference =

```
english = {'desi', 'tono', 'evan', 'miko', 'takashi', 'chaewon'}
# bisa berbahasa korea
korean = {'chaewon', 'yeona', 'erika', 'miko'}
# siapa yang hanya bisa bahasa korea?
only_korean = korean - english
print(len(only_korean))
# siapa yang hanya bisa bahasa english?
only_english = english
print(only_english)
```

Output =

```
{'erika', 'yeona'}
{'desi', 'takashi', 'tono', 'miko', 'evan', 'chaewon'}
PS C:\Users\LENOVO\Tugas PrakAlPro\Tugas 12>
```

Operator difference akan menghasilkan sebuah Set yang anggotanya merupakan selisih dari dua Set yang dibandingkan. Pada contoh tersebut, operator difference digunakan untuk mendapatkan anggota yang hanya bisa berbahasa Korea dengan mencari selisih antara Set korean dan Set english. Sebaliknya, jika ingin mengetahui siapa saja yang hanya bisa berbahasa Inggris, Anda dapat mencari selisih antara Set english dan Set korean.

Operator Symmetric Difference =

```
english = {'desi', 'tono', 'evan', 'miko', 'takashi', 'chaewon'}
korean = {'chaewon', 'yeona', 'erika', 'miko'}
# hanya bisa bicara satu bahasa saja
one_language = english ^ korean
print(one_language)
```

Output =
```
{'tono', 'takashi', 'evan', 'desi', 'erika', 'yeona'}
PS C:\Users\LENOVO\Tugas PrakAlPro\Tugas 12>
```

Operator symmetric difference akan menghasilkan sebuah Set baru yang merupakan gabungan dari dua Set, tetapi tidak termasuk elemen yang ada di keduanya. Contoh yang diberikan menggunakan operator symmetric difference untuk menentukan siapa saja yang hanya bisa berbicara dalam satu bahasa, sehingga elemen yang terdapat di kedua Set tidak termasuk.

```
one_languange = english.union(korean) - english.intersection(korean)
```

Kegiatan Praktikum

Kasus 12.1

Diberikan sebuah list yang berisi nilai-nilai integer, tugasnya adalah menghitung jumlah dari semua elemen unik di dalam list tersebut. Fungsi yang diminta adalah unique_sum(list). Pada kasus ini, Anda memiliki sebuah List yang berisi beberapa elemen bertipe integer.

```
def unique_sum(list):
# ubah dalam bentuk Set
    data_set = set(list)
# hitung jumlah seluruh anggota data_set
    total = 0
    for data in data_set:
        total = total + data
# return hasil akhir
    return total
# contoh penggunaan fungsi unique_sum()
contoh1 = [2, 4, 3, 2, 7, 8, 6, 4, 5, 5]
hasil1 = unique_sum(contoh1)
print(hasil1)
```

Output =
```
35
PS C:\Users\LENOVO\Tugas PrakAlPro\Tugas 12>
```

Kasus 12.2

Buatlah fungsi cek_duplikat(string) yang dapat mengecek apakah dalam string terdapat karakter yang muncul lebih dari satu kali (duplikat). Fungsi akan return True jika ada duplikat, dan sebaliknya return False jika tidak ada duplikat karakter dalam string yang diberikan.

```python
def cek_duplikat(string):
    # buat set kosong
    karakter_set = set()
    # cek semua karakter dalam string satu-persatu
    for karakter in string:
        # apakah karakter ini ada dalam set?
        if karakter in karakter_set:
            # ternyata ada, berarti duplikat
            # hentikan pengecekan, langsung return
            return True
        else:
            # jika belum ada, masukkan dalam set
            karakter_set.add(karakter)
    # setelah looping semua karakter, tidak ada yang sama
    return False

# test case
string1 = 'Alexander the Great'  # duplikat
print(cek_duplikat(string1))  # Output: True

string2 = 'UKDW'  # semua unik
print(cek_duplikat(string2))  # Output: False
```

Output =

```
True
False
PS C:\Users\LENOVO\Tugas PrakAlPro\Tugas 12>
```

Kasus 12.3

Buatlah program yang meminta input n kategori aplikasi, lalu program meminta pengguna untuk memasukkan nama-nama aplikasi sebanyak 5 untuk masing-masing kategori. Setelah pengguna selesai memasukkan semua nama aplikasi, program akan menampilkan:

- Daftar nama aplikasi di setiap kategori
- Program yang muncul di semua kategori

```
# input n kategori
n = int(input('Masukkan jumlah kategori: '))

# siapkan dictionary kosong
data_aplikasi = {}

# input nama kategori dan aplikasi di dalamnya
for i in range(n):
    nama_kategori = input('Masukkan nama kategori: ')
    print('Masukkan 5 nama aplikasi di kategori', nama_kategori)

    # siapkan list kosong untuk nama-nama aplikasi
    aplikasi = []
    for j in range(5):
        nama_aplikasi = input('Nama aplikasi: ')
        aplikasi.append(nama_aplikasi)

    # masukkan dalam dictionary
    data_aplikasi[nama_kategori] = aplikasi

# tampilkan dictionary data_aplikasi
print(data_aplikasi)

daftar_aplikasi_list = []

# ambil semua daftar aplikasi dari setiap kategori
for aplikasi in data_aplikasi.values():
    daftar_aplikasi_list.append(set(aplikasi))

print(daftar_aplikasi_list)

# lakukan intersection ke semua set yang ada
hasil = daftar_aplikasi_list[0]
for i in range(1, len(daftar_aplikasi_list)):
    hasil = hasil.intersection(daftar_aplikasi_list[i])

print(hasil)
```

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

```
n = int(input('Masukkan jumlah kategori: '))

data_aplikasi = {}

for i in range(n):
    nama_kategori = input('Masukkan nama kategori: ')
    print('Masukkan 5 nama aplikasi di kategori', nama_kategori)

    aplikasi = []
    for j in range(5):
        nama_aplikasi = input('Nama aplikasi: ')
        aplikasi.append(nama_aplikasi)

    data_aplikasi[nama_kategori] = aplikasi

print("Data aplikasi per kategori:")
print(data_aplikasi)

daftar_aplikasi_list = [set(aplikasi) for aplikasi in data_aplikasi.values()]

all_apps = set().union(*daftar_aplikasi_list)

app_count = {}
for app_set in daftar_aplikasi_list:
    for app in app_set:
        if app in app_count:
            app_count[app] += 1
        else:
            app_count[app] = 1

unique_apps = [app for app, count in app_count.items() if count == 1]
print("Aplikasi yang hanya muncul di satu kategori:")
print(unique_apps)

if n > 2:
    double_category_apps = [app for app, count in app_count.items() if count == 2]
    print("Aplikasi yang muncul tepat di dua kategori sekaligus:")
```

```
    print(double_category_apps)
else:
    print("Jumlah kategori tidak lebih dari 2, tidak ada aplikasi yang muncul
tepat di dua kategori sekaligus.")

if daftar_aplikasi_list:
    hasil = daftar_aplikasi_list[0]
    for i in range(1, len(daftar_aplikasi_list)):
        hasil = hasil.intersection(daftar_aplikasi_list[i])
    print("Aplikasi yang muncul di semua kategori:")
    print(hasil)
else:
    print("Tidak ada data aplikasi untuk dihitung.")
```

Output =

```
Masukkan jumlah kategori: 3
Masukkan nama kategori: Social Media
Masukkan 5 nama aplikasi di kategori Social Media
Nama aplikasi: Facebook
Nama aplikasi: Instagram
Nama aplikasi: Twitter
Nama aplikasi: Telegram
Nama aplikasi: Tiktok
Masukkan nama kategori: Produktivitas
Masukkan 5 nama aplikasi di kategori Produktivitas
Nama aplikasi: Microsoft Word
Nama aplikasi: Google Docs
Nama aplikasi: Notion
Nama aplikasi: Tiktok
Nama aplikasi: Tiktok
Nama aplikasi: Spotify
Nama aplikasi: Disney+
Data aplikasi per kategori:
{'Social Media': ['Facebook', 'Instagram', 'Twitter', 'Telegram', 'Tiktok'], 'Produktivitas': ['Microsoft Word', 'Google Docs', 'Notion', '
Trello', 'Tiktok'], 'Hiburan': ['Netfilx', 'Youtube', 'Tiktok', 'Spotify', 'Disney+']}
Aplikasi yang hanya muncul di satu kategori:
['Twitter', 'Telegram', 'Facebook', 'Instagram', 'Google Docs', 'Notion', 'Trello', 'Microsoft Word', 'Spotify', 'Disney+', 'Youtube', 'Net
filx']
Aplikasi yang muncul tepat di dua kategori sekaligus:
[]
Aplikasi yang muncul di semua kategori:
{'Tiktok'}
PS C:\Users\LENOVO\Tugas PrakAlPro\Tugas 12>
```

Penjelasan =

Kode tersebut meminta pengguna untuk memasukkan jumlah kategori dan nama aplikasi dalam setiap kategori. Setelah mengumpulkan data, program menghitung aplikasi yang hanya muncul di satu kategori, aplikasi yang muncul tepat di dua kategori, dan aplikasi yang muncul di semua kategori. Hasilnya kemudian ditampilkan dalam bentuk teks. Langkah-langkahnya adalah sebagai berikut:

Pertama, pengguna diminta untuk memasukkan jumlah kategori. Selanjutnya, program meminta nama kategori dan lima nama aplikasi untuk setiap kategori yang dimasukkan pengguna. Data aplikasi dimasukkan ke dalam dictionary yang menyimpan kategori sebagai kunci dan daftar aplikasi sebagai nilai.

Kemudian, program menghitung berapa kali setiap aplikasi muncul di semua kategori. Aplikasi yang hanya muncul di satu kategori diidentifikasi, begitu pula dengan aplikasi yang muncul tepat di dua kategori jika jumlah kategori lebih dari dua.

Hasil perhitungan tersebut kemudian ditampilkan dalam teks, termasuk daftar aplikasi yang hanya muncul di satu kategori, aplikasi yang muncul tepat di dua kategori (jika ada), dan aplikasi yang muncul di semua kategori.

Contoh Test Case

Masukkan jumlah kategori: 3

Masukkan nama kategori: Sosial Media

Masukkan 5 nama aplikasi di kategori Sosial Media

Nama aplikasi: Facebook

Nama aplikasi: Instagram

Nama aplikasi: Twitter

Nama aplikasi: Snapchat

Nama aplikasi: TikTok

Masukkan nama kategori: Produktivitas

Masukkan 5 nama aplikasi di kategori Produktivitas

Nama aplikasi: Microsoft Word

Nama aplikasi: Google Docs

Nama aplikasi: Notion

Nama aplikasi: Trello

Nama aplikasi: TikTok

Masukkan nama kategori: Hiburan

Masukkan 5 nama aplikasi di kategori Hiburan

Nama aplikasi: Netflix

Nama aplikasi: YouTube

Nama aplikasi: TikTok

Nama aplikasi: Spotify

Nama aplikasi: Disney+

SOAL 2

```python
# List awal
list_awal = [1, 2, 3, 4, 5]
print("List awal:", list_awal)

# Konversi List menjadi Set
set_hasil_1 = set(list_awal)
print("Set hasil konversi dari List:", set_hasil_1)

# Set awal
set_awal = {5, 6, 7, 8, 9}
print("\nSet awal:", set_awal)

# Konversi Set menjadi List
list_hasil_2 = list(set_awal)
print("List hasil konversi dari Set:", list_hasil_2)

# Tuple awal
tuple_awal = (9, 10, 11, 12, 13)
print("\nTuple awal:", tuple_awal)

# Konversi Tuple menjadi Set
set_hasil_3 = set(tuple_awal)
print("Set hasil konversi dari Tuple:", set_hasil_3)

# Set awal
set_awal_2 = {13, 14, 15, 16, 17}
print("\nSet awal:", set_awal_2)

# Konversi Set menjadi Tuple
tuple_hasil_4 = tuple(set_awal_2)
print("Tuple hasil konversi dari Set:", tuple_hasil_4)
```

Output =

```
List awal: [1, 2, 3, 4, 5]
Set hasil konversi dari List: {1, 2, 3, 4, 5}

Set awal: {5, 6, 7, 8, 9}
List hasil konversi dari Set: [5, 6, 7, 8, 9]

Tuple awal: (9, 10, 11, 12, 13)
Set hasil konversi dari Tuple: {9, 10, 11, 12, 13}

Set awal: {16, 17, 13, 14, 15}
Tuple hasil konversi dari Set: (16, 17, 13, 14, 15)
PS C:\Users\LENOVO\Tugas PrakAlPro\Tugas 12>
```

Penjelasan =

Program ini melakukan serangkaian konversi antara berbagai tipe data dan menampilkan hasilnya. Dimulai dengan membuat dan menampilkan sebuah list awal `[1, 2, 3, 4, 5]`, kemudian mengonversinya menjadi set dan menampilkannya. Selanjutnya, dibuat set awal `{5, 6, 7, 8, 9}` yang kemudian dikonversi menjadi list dan ditampilkan. Program juga membuat tuple awal `(9, 10, 11, 12, 13)`, mengonversinya menjadi set, dan menampilkannya. Terakhir, dibuat set awal kedua `{13, 14, 15, 16, 17}`, yang dikonversi menjadi tuple dan ditampilkan. Program ini menunjukkan proses konversi dari list ke set, set ke list, tuple ke set, dan set ke tuple, sambil menampilkan hasil setiap langkah.

Soal 3

```python
def baca_kata_dari_file(nama_file):
    try:
        with open(nama_file, 'r') as file:
            isi = file.read().lower()
            return set(isi.split())
    except FileNotFoundError:
        print(f"Error: File '{nama_file}' tidak ditemukan atau tidak bisa
dibaca.")
        return set()

nama_file1 = input("Masukkan nama file pertama: ")
nama_file2 = input("Masukkan nama file kedua: ")

kata_set1 = baca_kata_dari_file(nama_file1)
kata_set2 = baca_kata_dari_file(nama_file2)

kata_kedua_file = kata_set1 & kata_set2

if kata_kedua_file:
    print("Kata-kata yang muncul di kedua file:")
    print(kata_kedua_file)
else:
    print("Tidak ada kata yang muncul di kedua file.")
```

Output =

```
Masukkan nama file pertama: file1.txt
Masukkan nama file kedua: file2.txt
Kata-kata yang muncul di kedua file:
{'berisi', 'beberapa', 'contoh', 'ini', 'file', 'kata', 'adalah'}
PS C:\Users\LENOVO\Tugas PrakAlPro\Tugas 12>
```

Penjelasan =

Program ini memiliki satu fungsi utama, yaitu `baca_kata_dari_file(nama_file)`, yang bertanggung jawab membaca isi dari file yang diberikan. Fungsi ini menggunakan blok `try-except` untuk menangani apabila file tidak ditemukan.

Selanjutnya, program meminta pengguna untuk memasukkan nama dua file teks. Kemudian, program membaca kata-kata dari kedua file menggunakan fungsi sebelumnya dan menyimpannya sebagai set. Dengan menggunakan operator `&`, program mencari irisan (intersection) dari dua set kata-kata tersebut. Jika terdapat kata-kata yang sama di kedua file, program akan menampilkannya kepada pengguna; jika tidak, program akan memberitahu bahwa tidak ada kata yang sama.

Link Github

Link Github = https://github.com/YohanesNevan/Tugas-Laporan-Alpro-12.git