Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71230977**
Nama Lengkap	**MICHAEL HOSEA**
Minggu ke / Materi	**11 / 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.

Materi 1: Pengantar Set

Set adalah tipe data bawaan Python yang tidak berurutan dan tidak memiliki elemen duplikat. Set digunakan untuk menyimpan beberapa item dalam satu variabel, dan operasi matematis seperti union, intersection, difference, dan symmetric difference dapat dilakukan di atasnya. Artikel ini akan membahas konsep dasar, operasi, metode, dan contoh penggunaan set di Python.

Set dibuat dengan menggunakan tanda kurung {} atau fungsi set(). Berikut beberapa cara membuat set:

```
# Membuat set dengan cara kurung
fruits = {"apple", "banana", "cherry"}
print(fruits)

# Membuat set menggunakan fungsi set()
numbers = set([1, 2, 3, 4, 5])
print(numbers)
```

Terdapat 2 Karakteristik Set, yaitu:

1. Tidak Berurutan: Set tidak menyimpan elemen dalam urutan tertentu.
2. Tidak Ada Duplikasi: Set secara otomatis menghilangkan elemen duplikat.

```
# Set dengan elemen duplikat
sample_set = {1, 2, 2, 3, 4}
print(sample_set)  # Output: {1, 2, 3, 4}
```

Materi 2.1: Operasi dasar pada Set

1. Menambahkan Elemen
   Metode add() digunakan untuk menambahkan satu elemen ke dalam set.

```
my_set = {1, 2, 3}
my_set.add(4)
print(my_set)  # Output: {1, 2, 3, 4}
```

2. Menghapus Elemen
   Metode remove() atau discard() digunakan untuk menghapus elemen dari set. Bedanya, remove() akan menimbulkan error jika elemen tidak ditemukan, sedangkan discard() tidak.

```
my_set = {1, 2, 3, 4}
my_set.remove(4)
print(my_set)  # Output: {1, 2, 3}

my_set.discard(3)
print(my_set)  # Output: {1, 2}
```

3. Mengecek Keanggotaan
   Pengguna dapat menggunakan operator in untuk memeriksa apakah sebuah item ada dalam set. Ini berguna untuk menghindari kesalahan saat mencoba mengakses item yang tidak ada dalam set.

```
my_set = {1, 2, 3, 4, 5}
if 5 in my_set:
    print("5 ada dalam Set")
# Output: 5 ada dalam Set
```

4. Gabungan Set
   Pengguna dapat menggabungkan dua set menggunakan metode union() atau operator |. Ini akan mengembalikan set baru yang berisi semua elemen dari kedua set.

```
my_set = {1, 2, 3}
other_set = {4, 5, 6, 7}
union_set = my_set.union(other_set)
print(union_set)  # Output: {1, 2, 3, 4, 5, 6, 7}
```

5. Panjang Set
   Panjang atau jumlah item dalam set dapat dihitung menggunakan fungsi len().

```
my_set = {1, 2, 3, 4, 5}
print("Panjang Set:", len(my_set))  # Output: Panjang Set: 5
```

6. Menghapus Semua Elemen
   Untuk menghapus semua item dari set, Anda dapat menggunakan metode clear().

```
my_set = {1, 2, 3, 4, 5}
my_set.clear()
```

```
print(my_set)  # Output: set()
```

7. Iterasi Set
   Pengguna dapat melakukan iterasi atau loop melalui item-item dalam set menggunakan loop for.

```
my_set = {1, 2, 3, 4, 5}
for item in my_set:
    print(item)
# Output:
# 1
# 2
# 3
# 4
# 5
```

Materi 2.2: Operasi Set

1. Union (Gabungan)
   Operasi union menggabungkan dua set dan mengembalikan set baru yang berisi semua elemen unik dari kedua set.

```
set_a = {1, 2, 3}
set_b = {3, 4, 5}
set_union = set_a.union(set_b)
print(set_union)  # Output: {1, 2, 3, 4, 5}
```

2. Intersection (Irisan)
   Operasi intersection mengembalikan set baru yang berisi elemen-elemen yang ada di kedua set.

```
set_a = {1, 2, 3}
set_b = {3, 4, 5}
set_intersection = set_a.intersection(set_b)
print(set_intersection)  # Output: {3}
```

3. Difference (Selisih)
   Operasi difference mengembalikan set baru yang berisi elemen dari set pertama yang tidak ada di set kedua.

```
set_a = {1, 2, 3}
set_b = {3, 4, 5}
```

```
set_difference = set_a.difference(set_b)
print(set_difference)  # Output: {1, 2}
```

4. Symmetric Difference (Selisih Simetris)
   Operasi symmetric difference mengembalikan set baru yang berisi elemen yang ada di salah satu set, tetapi tidak keduanya.

```
set_a = {1, 2, 3}
set_b = {3, 4, 5}
set_sym_diff = set_a.symmetric_difference(set_b)
print(set_sym_diff)  # Output: {1, 2, 4, 5}
```

Materi 2.3: Metode Lain pada Set

1. Update()
   Metode ini digunakan untuk menambahkan elemen-elemen dari iterable lain ke dalam set.

```
set_a = {1, 2, 3}
set_a.update([3, 4, 5])
print(set_a)  # Output: {1, 2, 3, 4, 5}
```

2. Clear()
   Metode ini digunakan untuk menghapus semua elemen dari set.

```
set_a = {1, 2, 3}
set_a.clear()
print(set_a)  # Output: set()
```

Materi 2.4: Frozen Set

Frozen set adalah varian dari set yang tidak dapat diubah setelah dibuat. Ini berguna ketika Anda ingin menggunakan set sebagai kunci di dictionary atau memasukkannya ke dalam set lain.

```
frozen_set = frozenset([1, 2, 3, 4])
print(frozen_set)

# Mencoba menambahkan elemen akan menimbulkan error
# frozen_set.add(5)  # AttributeError
```

Materi 3: Penggunaan Set dalam Program

1. Menghapus Duplikat dari list

```python
my_list = [1, 2, 2, 3, 4, 4, 5]
unique_items = list(set(my_list))
print(unique_items)  # Output: [1, 2, 3, 4, 5]
```

2. Operasi Himpunan

```python
engineers = {"Alice", "Bob", "Charlie"}
managers = {"Bob", "David"}

# Siapa yang adalah engineer tetapi bukan manager?
hanya_engineers = engineers.difference(managers)
print(hanya_engineers)  # Output: {"Alice", "Charlie"}
```

3. Validasi Cepat dalam Algoritma
   Dalam beberapa algoritma, kita mungkin perlu memeriksa apakah elemen tertentu sudah pernah dilihat sebelumnya. Set memungkinkan kita melakukan ini dengan sangat cepat.

```python
seen = set()
numbers = [1, 2, 3, 2, 4, 5, 1]

for number in numbers:
    if number in seen:
        print(f"{number} sudah ada sebelumnya")
    else:
        seen.add(number)
```

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 MANDIRI 1:

Dari contoh kasus kategori kasus di Play Store, tambahkan kemampuan-kemampuan berikut ini:

- Tampilkan nama-nama aplikasi yang hanya muncul di satu kategori saja.
- Untuk input n > 2, tampilkan nama-nama aplikasi yang muncul tepat di dua kategori sekaligus.

Source Code dan Output:

```
def aplikasiPlayStore(n):
    # Membuat dictionary untuk menyimpan kategori dan aplikasinya
    kategoriDict = {}
    for i in range(n):
        kategori = str(input(f'Masukkan nama kategori ke-{i + 1}: '))
        aplikasiList = []
        for j in range(5):
            aplikasi = str(input(f'Masukkan nama aplikasi ke-{j + 1} dalam kategori
{kategori}: '))
            aplikasiList.append(aplikasi)
        kategoriDict[kategori] = aplikasiList
        print()

    print(kategoriDict)
    print()

    # Membuat list untuk menyimpan aplikasi dari setiap kategori
    semuaAplikasi = []
    for k in kategoriDict.keys():
        semuaAplikasi.append(kategoriDict[k])

    # Mencari aplikasi yang muncul di semua kategori
    hasil = set(semuaAplikasi[0])
    for i in range(1, len(semuaAplikasi)):
        hasil = hasil.intersection(set(semuaAplikasi[i]))
    if hasil:
```

```
        print("Aplikasi yang ada di semua kategori:", hasil)
    else:
        print('Tidak ada aplikasi yang muncul di semua kategori!')

    # Mencari aplikasi yang hanya ada di satu kategori
    unikHasil = set(semuaAplikasi[0])
    for i in range(1, len(semuaAplikasi)):
        unikHasil = unikHasil.symmetric_difference(set(semuaAplikasi[i]))
    if unikHasil:
        print("Aplikasi yang hanya ada di satu kategori:", unikHasil)
    else:
        print('Semua aplikasi muncul di lebih dari satu kategori!')

    # Mencari aplikasi yang muncul tepat di dua kategori
    if n > 2:
        aplikasiCounter = {}
        for lst in semuaAplikasi:
            for aplikasi in lst:
                if aplikasi in aplikasiCounter:
                    aplikasiCounter[aplikasi] += 1
                else:
                    aplikasiCounter[aplikasi] = 1
        duaKategoriHasil = {app for app, count in aplikasiCounter.items() if count ==
2}
        if duaKategoriHasil:
            print('Aplikasi yang muncul tepat di dua kategori:', duaKategoriHasil)
        else:
            print('Tidak ada aplikasi yang muncul tepat di dua kategori!')

# Meminta input jumlah kategori dari pengguna
n = int(input('Masukkan jumlah kategori: '))
aplikasiPlayStore(n)
```

PENJELASAN:

Kode ini adalah sebuah program yang meminta pengguna untuk memasukkan beberapa kategori aplikasi dan aplikasi-aplikasi yang termasuk dalam setiap kategori tersebut. Program ini kemudian menyimpan data ini dalam sebuah dictionary, di mana setiap kunci adalah nama kategori dan nilainya adalah daftar aplikasi yang terkait.

Setelah data dikumpulkan, program melakukan beberapa analisis pada aplikasi-aplikasi tersebut. Pertama, program mencari dan mencetak aplikasi yang muncul di semua kategori. Ini dilakukan dengan menggunakan operasi himpunan untuk menemukan irisan dari semua daftar aplikasi. Kedua, program mencari aplikasi yang hanya muncul di satu kategori saja, menggunakan operasi symmetric difference pada himpunan aplikasi dari setiap kategori. Terakhir, jika jumlah kategori lebih dari dua, program menghitung dan mencetak aplikasi yang muncul tepat di dua kategori. Ini dilakukan dengan membuat dictionary tambahan untuk menghitung frekuensi kemunculan setiap aplikasi di berbagai kategori, kemudian mencari aplikasi yang frekuensinya tepat dua.

Program ini memberikan output yang menunjukkan aplikasi-aplikasi mana yang umum di semua kategori, mana yang unik untuk satu kategori, dan mana yang muncul tepat di dua kategori, sehingga dapat membantu dalam analisis dan pengelompokan aplikasi berdasarkan kategorinya.

SOAL MANDIRI 2:

Buatlah sebuah program yang mendemonstrasikan konversi dari:

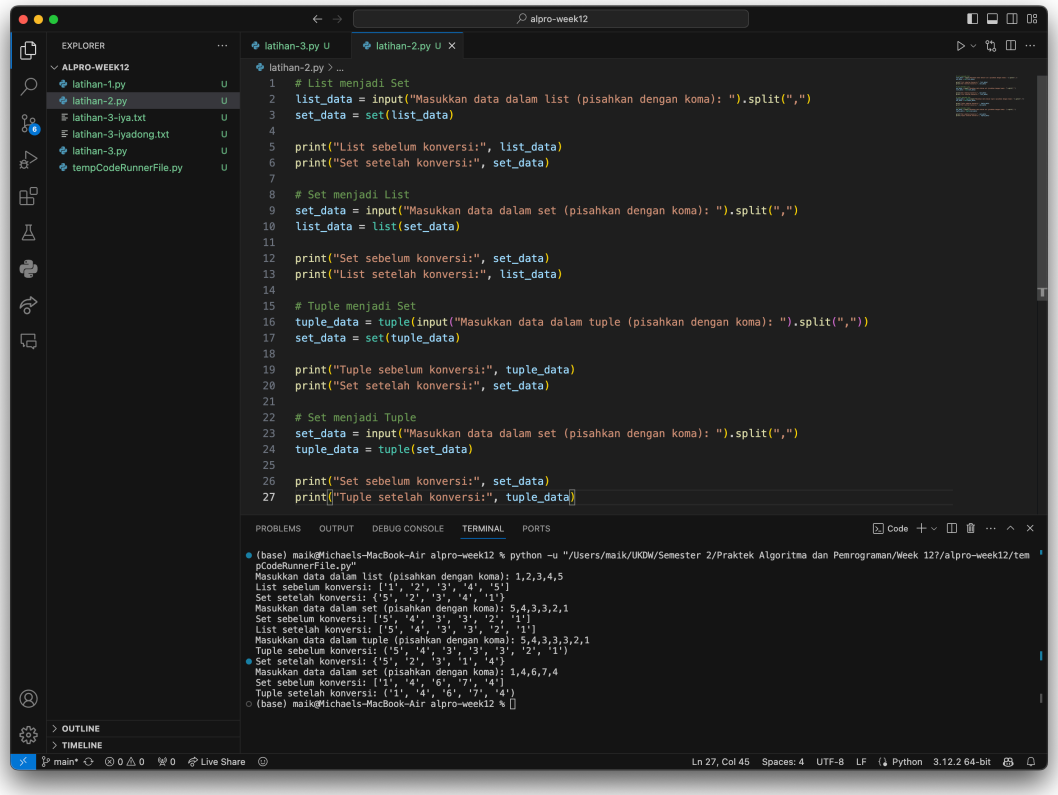
- List menjadi Set
- Set menjadi List
- Tuple menjadi Set
- Set menjadi Tuple

Tampilkan isi data sebelum dan sesudah konversi.

- Source Code dan Output

```
nama = input("Masukkan Nama: ")
nim = input("Masukkan NIM: ")
alamat = input("Masukkan Alamat: ")

data_diri = (nama, nim, alamat)

# Menampilkan data diri
print("NIM    : ", data_diri[1])
print("NAMA   : ", data_diri[0])
print("ALAMAT : ", data_diri[2])

# Memisahkan NIM menjadi tuple
```

```python
nim_tuple = tuple(data_diri[1])
print("NIM: ", nim_tuple)

# Memisahkan nama depan menjadi tuple
nama_depan = tuple(data_diri[0].split()[0])
print("NAMA DEPAN: ", nama_depan)

# Membalikkan nama menjadi tuple
nama_terbalik = tuple(data_diri[0].split()[::-1])
print("NAMA TERBALIK: ", nama_terbalik)
```



- Penjelasan
  1. Input Data Diri:
     - Mengambil input nama, NIM, dan alamat dari pengguna.
  2. Menyimpan Data dalam Tuple:
     - Menyimpan nama, NIM, dan alamat dalam tuple data_diri.
  3. Menampilkan Data Diri:
     - Menampilkan NIM, nama, dan alamat dari tuple data_diri.
  4. Memisahkan NIM Menjadi Tuple:
     - Mengubah NIM menjadi tuple karakter dan menampilkannya.

5. Memisahkan Nama Depan Menjadi Tuple:
   - Mengambil nama depan (kata pertama dari nama), mengubahnya menjadi tuple karakter, dan menampilkannya.
6. Membalikkan Nama Menjadi Tuple:
   - Membalik urutan nama (dari nama lengkap) menjadi tuple dan menampilkannya.

SOAL MANDIRI 3:

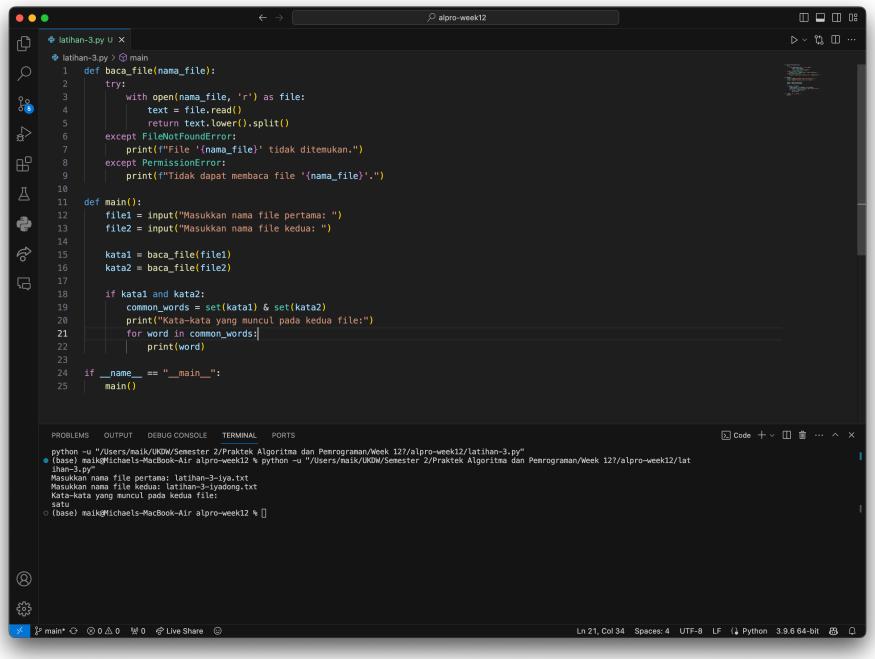
Buatlah sebuah program yang dapat membaca dua file teks dan menampilkan semua kata-kata yang muncul pada kedua file tersebut. Beberapa hal yang perlu anda perhatikan:

- Nama file adalah input user. Tampilkan pesan error jika file tidak ditemukan/tidak bisa dibaca.
- Semua kata dikonversi dulu menjadi lowercase.
- Sertakan contoh file teks yang anda pakai saat mengumpulkan laporan.

- Source Code dan Output

```
def baca_file(nama_file):
    try:
        with open(nama_file, 'r') as file:
            text = file.read()
            return text.lower().split()
    except FileNotFoundError:
        print(f"File '{nama_file}' tidak ditemukan.")
    except PermissionError:
        print(f"Tidak dapat membaca file '{nama_file}'.")

def main():
    file1 = input("Masukkan nama file pertama: ")
    file2 = input("Masukkan nama file kedua: ")

    kata1 = baca_file(file1)
    kata2 = baca_file(file2)

    if kata1 and kata2:
        common_words = set(kata1) & set(kata2)
        print("Kata-kata yang muncul pada kedua file:")
```

```
        for word in common_words:
            print(word)

if __name__ == "__main__":
    main()
```



- Penjelasan
  1. Fungsi baca_file

     Pada fungsi ini memiliki tujuan untuk membaca isi file dan mengembalikan daftar kata-kata dalam file tersebut. Text dibuat .lower() agar tidak terjadi perbedaan antara huruf kapital dan huruf kecil. Text juga dipisah setiap katanya dengan menggunakan .split() agar kita dapat mencari kata apa yang sama.

  2. Fungsi main

     Pada fungsi ini pengguna diminta untuk menuliskan nama file yang akan di bandingkan. Jika kedua file berhasil dibaca (kata1 dan kata2 tidak None), lakukan: Mengonversi daftar kata dari kedua file menjadi set dan mencari irisan (intersection) dari kedua set untuk menemukan kata-kata yang umum di kedua file (common_words), akan mencetak kata-kata yang muncul di kedua file.