Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**<71231051>**
Nama Lengkap	**<Amida A Ronsumbre>**
Minggu ke / Materi	**13/ 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

- 12.3.1 Pengenalan dan Mendefinisikan Set

Set adalah tipe data untuk menyimpan elemen unik.

Anggota Set harus immutable (contoh: integer, float, string, tuple).

Set itu sendiri mutable (bisa diubah isinya).

Gunakan {} atau set() untuk mendefinisikan Set, tetapi gunakan set() untuk Set kosong. {} mendefinisikan dictionary kosong,Elemen-elemen dalam Set disebut anggota (member).

Anggota dari Set harus bersifat immutable, seperti integer, float, string, dan tuple. Oleh karena itu, list dan dictionary (yang bersifat mutable) tidak dapat dimasukkan ke dalam Set.Mutabilitas Set:

Set itu sendiri bersifat mutable, artinya Anda dapat menambah atau mengurangi elemen dalam Set. Namun, karena elemen dalam Set harus bersifat immutable, Set tidak dapat menjadi anggota dari Set lain.Anda dapat mendefinisikan Set menggunakan notasi {} atau fungsi.

- 12.3.2 Pengaksesan Set

Hal ini disebabkan karena pada Set tidak ada indeks, sehingga tidak ada urutan posisi anggota. Pada tipe data Set, posisi anggota tidaklah penting.

Berikut ini adalah contoh program yang mendemonstrasikan penghapusan anggota dari sebuah

hapus 97 bilangan_prima. # ambil dan hapus salah satu bilangan = bilangan_prima. # kosongkan set bilangan_prima.

Output yang dihasilkan dari program tersebut adalah sebagai berikut

Untuk mengubah nilai ’koi’ menjadi ’teri’, yang perlu dilakukan adalah hapus dulu anggota ’koi’, kemudian masukkan anggota baru dengan nilai ’teri’. Setiap kali ada operasi penambahan dan penghapusan maka urutan anggota di dalam Set biasanya akan berubah.

- 12.3.3 Operasi-Operasi pada Set

Operator Union. Dapat menggunakan operator | maupun fungsi union. Operator Intersection. Dapat menggunakan operator & maupun fungsi intersection.

Operator Difference. Dapat menggunakan operator- maupun fungsi difference. Operator Symmetric Difference. Dapat menggunakan operator ^ maupun fungsi symmetric_difference.

Output yang dihasilkan dari program tersebut adalah anggota dari Set merek_hp digabungkan dengan anggota dari Set merek_ac, menghasilkan Set baru bernama gabungan.

Intersection berarti anggota-anggota yang berada di dalam Set renang dan Set tenis sekaligus, yaitu mail, upin dan ipin.

Operator difference akan menghasilkan Set yang anggotanya merupakan selisih dari kedua Set yang dibandingkan.

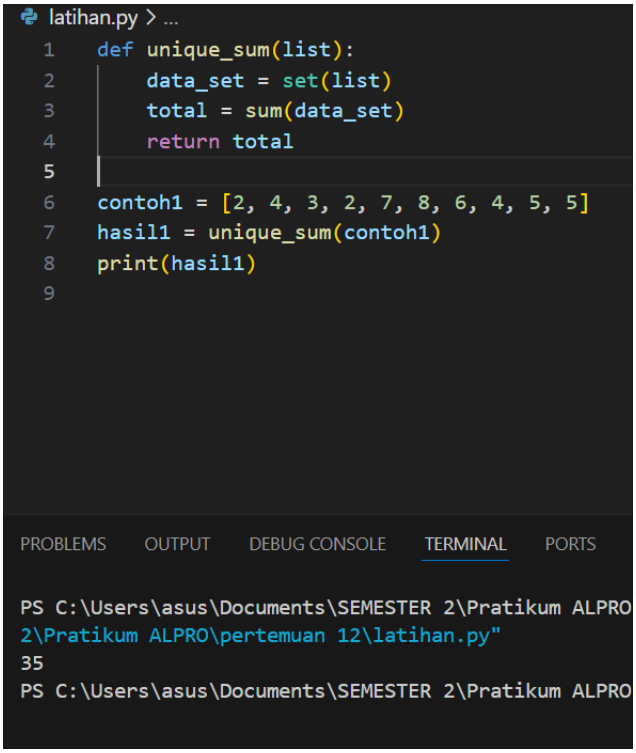
Contoh yang diberikan memanfaatkan operator symmetric difference untuk mendapatkan siapa saja yang hanya dapat berbicara dalam satu bahasa . Union- english.

MATERI 2

Penjelasan materi 2, dst… sesuai format ini.

- 12.4.1 Contoh-contoh Kasus Set
  - Jumlah seluruh elemen yang unik di dalam List

Pada kasus ini anda diberi satu buah List yang di dalamnya sudah ada beberapa elemen-elemen bertipe integer. Anda diminta untuk menghitung jumlah seluruh elemen yang unik di dalam list tersebut. Untuk memecahkan masalah ini, kita akan menggunakan bantuan Set.Contohnya:



- Duplicate chars in string

Buatlah program yang meminta input n kategori aplikasi, lalu program meminta pengguna untuk memasukkan nama-nama aplikasi sebanyak 5 untuk masing-masing kategori.

Dengan operasi intersection, tampilkan nama aplikasi yang muncul di semua kategori yang ada.

ambil semua daftar aplikasi dari setiap kategori for aplikasi in data_aplikasi. Kemudian dari setiap kategori dilakukan operasi intersection untuk mencari apakah ada nama aplikasi yang muncul di semua kategori.

latihan.py > ...

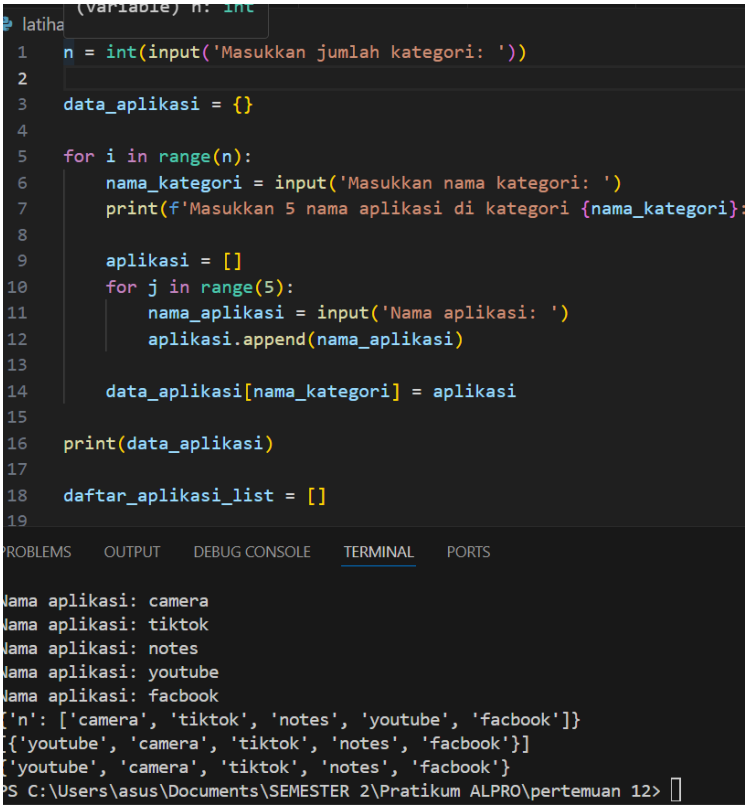
```python
def cek_duplikat(string):
    karakter_set = set()

    for karakter in string:
        if karakter in karakter_set:
            return True
        else:
            karakter_set.add(karakter)

    return False

# Test case
string1 = 'Alexander the Great' # Ada duplikat
print(cek_duplikat(string1))

string2 = 'UKDW' # Semua karakter unik
print(cek_duplikat(string2))
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

2\Pratikum ALPRO\pertemuan 12\latihan.py"
True
False
PS C:\Users\asus\Documents\SEMESTER 2\Pratikum ALPRO\pertemuan 12> pyt
2\Pratikum ALPRO\pertemuan 12\latihan.py"
True
False
PS C:\Users\asus\Documents\SEMESTER 2\Pratikum ALPRO\pertemuan 12>
```

- Kategori aplikasi di Play Store

Dengan operasi intersection, tampilkan nama aplikasi yang muncul di semua kategori yang ada.

ambil semua daftar aplikasi dari setiap kategori for aplikasi in data_aplikasi. Pada contoh tersebut, pengguna memasukkan 2 kategori, yaitu Finance dan Utilities.



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

INPUT:

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

data_aplikasi = {}

for i in range(n):
    nama_kategori = input('Masukkan nama kategori: ')
    print(f'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("\nData aplikasi per kategori:")
print(data_aplikasi)

daftar_aplikasi_list = []
aplikasi_count = {}

for aplikasi in data_aplikasi.values():
    aplikasi_set = set(aplikasi)
    daftar_aplikasi_list.append(aplikasi_set)
```

```python
    for app in aplikasi_set:
        if app in aplikasi_count:
            aplikasi_count[app] += 1
        else:
            aplikasi_count[app] = 1
print("\nDaftar aplikasi per kategori dalam bentuk set:")
print(daftar_aplikasi_list)

hasil_intersection = daftar_aplikasi_list[0]
for i in range(1, len(daftar_aplikasi_list)):
    hasil_intersection = hasil_intersection.intersection(daftar_aplikasi_list[i])

print("\nAplikasi yang muncul di semua kategori:")
print(hasil_intersection)

aplikasi_unik = {app for app, count in aplikasi_count.items() if count == 1}
print("\nAplikasi yang hanya muncul di satu kategori saja:")
print(aplikasi_unik)

aplikasi_dua_kategori = {app for app, count in aplikasi_count.items() if count == 2}
print("\nAplikasi yang muncul tepat di dua kategori:")
print(aplikasi_dua_kategori)
```

OUTPUT:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

\pertemuan 12\latihan.py"
Masukkan jumlah kategori: 2
Masukkan nama kategori: n
Masukkan 5 nama aplikasi di kategori n:
Nama aplikasi: tiktok
Nama aplikasi: camera
Nama aplikasi: foto
Nama aplikasi: notes
Nama aplikasi: musik
Masukkan nama kategori: n
Masukkan 5 nama aplikasi di kategori n:
Nama aplikasi: musik
Nama aplikasi: foto
Nama aplikasi: notes
Nama aplikasi: tiktok
Nama aplikasi: camera

Data aplikasi per kategori:
{'n': ['musik', 'foto', 'notes', 'tiktok', 'camera']}

Daftar aplikasi per kategori dalam bentuk set:
[{'musik', 'camera', 'tiktok', 'foto', 'notes'}]

Aplikasi yang muncul di semua kategori:
{'musik', 'camera', 'tiktok', 'foto', 'notes'}

Aplikasi yang hanya muncul di satu kategori saja:
{'musik', 'camera', 'tiktok', 'foto', 'notes'}

Aplikasi yang muncul tepat di dua kategori:
set()
PS C:\Users\asus\Documents\SEMESTER 2\Pratikum ALPRO\pertemuan 12>
```

Menghitung Kemunculan Aplikasi:

Kita menggunakan dictionary aplikasi_count untuk menghitung berapa kali setiap aplikasi muncul di semua kategori.

Menampilkan Aplikasi yang Hanya Muncul di Satu Kategori:

Kita menampilkan aplikasi yang memiliki hitungan kemunculan 1 dalam aplikasi_count.

Menampilkan Aplikasi yang Muncul Tepat di Dua Kategori:

Kita menampilkan aplikasi yang memiliki hitungan kemunculan 2 dalam aplikasi_count.

Input jumlah kategori:

Program meminta pengguna untuk memasukkan jumlah kategori (n).Input nama kategori dan aplikasi:

Program meminta nama kategori.Untuk setiap kategori, program meminta 5 nama aplikasi dan menyimpannya dalam sebuah list.

Setiap list aplikasi kemudian disimpan dalam dictionary data_aplikasi dengan nama kategori sebagai kuncinya.

Langkah 2: Mengubah List Aplikasi ke Set dan Menghitung Kemunculan

Mengubah list ke set:

Program mengubah setiap list aplikasi menjadi set untuk memastikan aplikasi dalam setiap kategori unik.Setiap set aplikasi disimpan dalam list daftar_aplikasi_list.

Menghitung kemunculan aplikasi:Program menggunakan dictionary aplikasi_count untuk menghitung berapa kali setiap aplikasi muncul di semua kategori.

Saat aplikasi dimasukkan ke dalam set, program juga menghitung kemunculannya.

Langkah 3: Menampilkan Hasil

Aplikasi yang muncul di semua kategori:Menggunakan operasi intersection pada semua set dalam daftar_aplikasi_list untuk menemukan aplikasi yang muncul di semua kategori.

Aplikasi yang hanya muncul di satu kategori:Mengambil aplikasi dari aplikasi_count yang memiliki nilai 1, yang berarti aplikasi tersebut hanya muncul di satu kategori.

Aplikasi yang muncul tepat di dua kategori:Mengambil aplikasi dari aplikasi_count yang memiliki nilai 2, yang berarti aplikasi tersebut muncul tepat di dua kategori.

Soal 2

```python
list_data = [1, 2, 3, 4, 5, 1, 2]
print("List sebelum konversi:", list_data)

set_data = set(list_data)
print("Set setelah konversi dari List:", set_data)

set_data = {1, 2, 3, 4, 5}
print("\nSet sebelum konversi:", set_data)

list_data = list(set_data)
print("List setelah konversi dari Set:", list_data)

tuple_data = (1, 2, 3, 4, 5, 1, 2)
print("\nTuple sebelum konversi:", tuple_data)

set_data = set(tuple_data)
print("Set setelah konversi dari Tuple:", set_data)

set_data = {1, 2, 3, 4, 5}
print("\nSet sebelum konversi:", set_data)

tuple_data = tuple(set_data)
print("Tuple setelah konversi dari Set:", tuple_data)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\Documents\SEMESTER 2\Pratikum ALPRO\pertemuan 12> pyth
\pertemuan 12\latihan.py"
List sebelum konversi: [1, 2, 3, 4, 5, 1, 2]
Set setelah konversi dari List: {1, 2, 3, 4, 5}

Set sebelum konversi: {1, 2, 3, 4, 5}
List setelah konversi dari Set: [1, 2, 3, 4, 5]

Tuple sebelum konversi: (1, 2, 3, 4, 5, 1, 2)
Set setelah konversi dari Tuple: {1, 2, 3, 4, 5}

Set sebelum konversi: {1, 2, 3, 4, 5}
Tuple setelah konversi dari Set: (1, 2, 3, 4, 5)
PS C:\Users\asus\Documents\SEMESTER 2\Pratikum ALPRO\pertemuan 12>
```

List ke Set: Berguna untuk menghilangkan elemen duplikat.

Set ke List: Berguna untuk mendapatkan urutan elemen dan manipulasi berbasis indeks.

Tuple ke Set: Berguna untuk menghilangkan elemen duplikat dari struktur data yang immutable.

Set ke Tuple: Berguna untuk membuat elemen-elemen unik menjadi immutable.

Konversi ini memungkinkan kita untuk memanfaatkan karakteristik khusus dari setiap tipe data sesuai dengan kebutuhan spesifik kita.

List menjadi Set

Namun, jika kita perlu menghilangkan duplikat dan hanya menyimpan elemen-elemen unik, kita dapat mengkonversi List menjadi Set. Set adalah koleksi yang tidak mengizinkan elemen duplikat, sehingga semua elemen dalam Set akan unik.

Tuple menjadi Set

Jika kita ingin menghilangkan duplikat dari Tuple dan tidak peduli tentang urutan elemen, kita dapat mengkonversi Tuple menjadi Set. Konversi ini memastikan bahwa hanya elemen-elemen unik yang disimpan.

Set menjadi Tuple

Dengan mengonversi Set menjadi Tuple, kita memastikan bahwa elemen-elemen tersebut tidak dapat diubah setelahnya, yang bisa berguna untuk memastikan konsistensi dan mencegah modifikasi yang tidak disengaja.

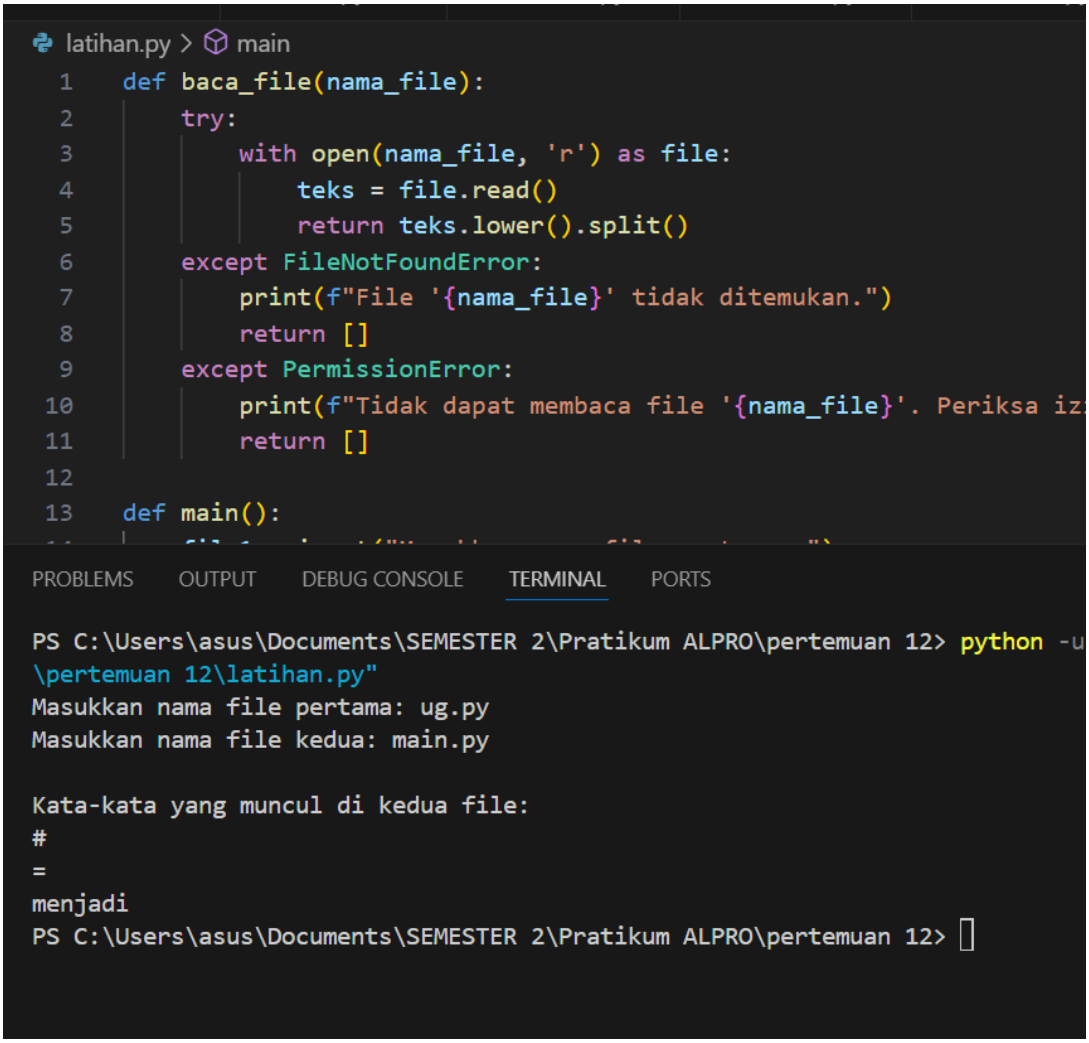
Soal 3

INPUT:

latihan.py > main

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

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

    kata_file1 = baca_file(file1)
    kata_file2 = baca_file(file2)

    set_kata_file1 = set(kata_file1)
    set_kata_file2 = set(kata_file2)

    kata_yang_sama = set_kata_file1.intersection(set_kata_file2)
    if kata_yang_sama:
        print("\nKata-kata yang muncul di kedua file:")
        for kata in kata_yang_sama:
            print(kata)
    else:
        print("\nTidak ada kata yang muncul di kedua file.")
```

OUTPUT:



Langkah 1: Persiapkan File-File Teks

Ini penting karena program akan membaca isi kedua file tersebut untuk menemukan kata-kata yang muncul di kedua file.

Langkah 2: Tulis Program Python

Ini penting untuk melakukan tugas yang diminta, yaitu menampilkan kata-kata yang muncul di kedua file.

Memiliki Data Masukan: Langkah pertama adalah memastikan bahwa Anda memiliki setidaknya dua file teks yang akan digunakan sebagai contoh. Ini penting karena program akan membaca isi kedua file tersebut untuk menemukan kata-kata yang muncul di kedua file.

Langkah 2: Tulis Program PythonMembuat Logika Program: Dengan menulis program Python, Anda membuat alur kerja yang diperlukan untuk membaca isi file teks, memproses kata-kata, dan

menemukan kata-kata yang muncul di kedua file. Ini penting untuk melakukan tugas yang diminta, yaitu menampilkan kata-kata yang muncul di kedua file.

Langkah 3: Jalankan Program

Eksekusi Program: Langkah terakhir adalah menjalankan program yang telah Anda tulis. Dengan menjalankan program, Anda dapat melihat hasil dari logika yang telah Anda buat. Ini memungkinkan Anda untuk memeriksa apakah program berjalan dengan benar dan apakah hasilnya sesuai dengan yang Anda harapkan.