Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**<71230972>**
Nama Lengkap	**<Oktavian Christ Putranto>**
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.

Tipe data set

Sederhananya tipe data set merupakan sebuah himpunan seperti yang sering kita pakai dalam matematika, dalam satu set tidak akan ada data yang ter double dan tidak terdiri dari indeks sehingga cara mengaksesnya sangat berbeda dengan tipe data list, member/isi dari set juga dapat berupa gabungan beberapa tipe data seperti integer,string,char dan set sendiri termasuk immutable.

Set ditandai dengan lambang kurung {} sama dengan dictionary oleh karena itu pelu diperhatikan bahwa untuk membuat sebuah set kosong maka kita perlu memakai set() contoh :

```
tes={}
print(type(tes))
```

bila kita menulis seperti contoh tersebut maka outputnya justru menampilakan <class 'dict'>

oleh sebab itu maka input yang kita masukan ialah :

```
tes=set()
print(type(tes))
```

ketika kita ingin membuat set kosong maka perintah diataslah yang efektif karena akan menghasilkan output <class 'set'>

selain perbedaan tersebut kita juga harus memperhatikan beberapa perbedaan oprasi dalam menangani tipe data set salah satunya ialah cara mengisi set kosong contoh :

```
tes=set()
for i in range(1,15):
    tes.append(i)
print(tes)
```

dalam mengolah data list kita terbiasa memakai append untuk mengisi list kosong apabila kita memakai append dalam set maka yang terjadi justru eror

oleh karena itu kita memakai fungsi add untuk tipe data set :

```
tes=set()
for i in range(1,15):
    tes.add(i)
print (tes)
```

dengan begini maka outputnya akan menghasilkan nilai yang diinginkan yaitu :

{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14}

Selain itu tipe data set juga tidak memiliki indeks, tidak seperti list yang dapat diakses berdasarkan indeksnya set justru tidak bisa contoh :

```
tes = []
for i in range(1,15):
    tes.append(i)
print (tes[3])
```

Output : 4

Dalam set hal ini akan mengahasilkan eror, contoh :

```
tes=set()
for i in range(1,15):
    tes.add(i)
print (tes[1])
```

Output : TypeError: 'set' object is not subscriptable

selain tidak punya indeks set juga akan mengacak urutan dari isi/member set nya ketika set tersebut terdiri dari data char/string dan akan langsung mengurutkan/sorting untuk data integer

perlu di ingat juga bahwa data dalam set tidak akan ke double

contoh :

```
tes={1,5,7,2,3,5,6,6,6,90,12,1}
print(tes)
```

output : {1, 2, 3, 5, 6, 7, 12, 90}

data yang dihasilkan akan langsung terurut dan tidak ke double

contoh untuk data string :

```
tes={"apel","banana","melon","jeruk","apel"}
print(tes)
```

Output pertama : {'jeruk', 'banana', 'melon', 'apel'}

Output kedua : {'jeruk', 'melon', 'banana', 'apel'}

Output ketiga : {'jeruk', 'banana', 'apel', 'melon'}

Dari contoh tersebut dapat kita lihat bahwa output dari set tersebut akan random urutannya ketika di isi oleh data string

Contoh data campuran :

```
tes={"apel","banana",1,4,2,"melon","jeruk","apel",11,15,90}
print(tes)
```

output_1 : {1, 'apel', 2, 4, 'banana', 'melon', 'jeruk', 11, 15, 90}

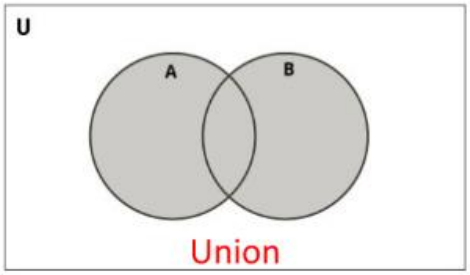
Output_2 : {'melon', 1, 2, 4, 'apel', 11, 'jeruk', 15, 'banana', 90}

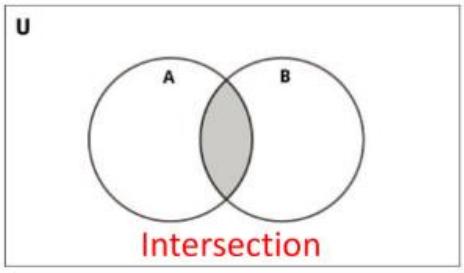
Output _3 : {1, 2, 4, 'banana', 11, 15, 'jeruk', 'apel', 'melon', 90}

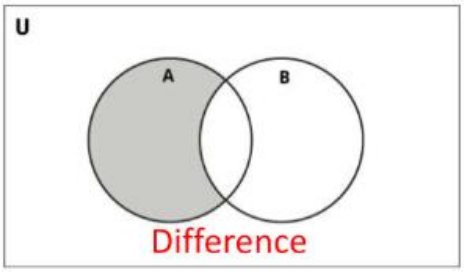
Ketika data berisi campuran dari integer dan string amaka urutana kan teracak dengan integer yang tetap ter sorting.

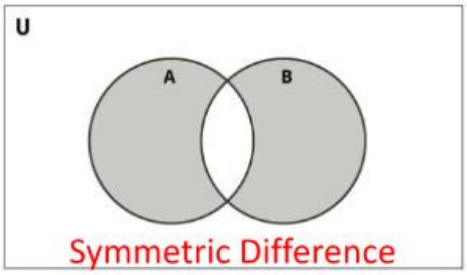
Oprasi hitung pada tipe data Set

Seperti penjelasan ditatas set merupakan sebuah himpunan oleh karena itu kita juga dapat menggunakan fungsi fungsi khusus berupa union, difference, intersection, dan symmetric difference









Contoh penggunaan union :

```
set1={1,4,5,2,12}
set2={9,0,13,1,2,5,17,90}

print (set1.union(set2))
```

Output :

{0, 1, 2, 4, 5, 9, 12, 13, 17, 90}

Contoh lain :

```
set1={1,4,5,2,12}
set2={9,0,13,1,2,5,17,90}
set3={1,100}
set4=set1|set2|set3
print (set1|set2|set3)
print(set4)
```

Output :

{0, 1, 2, 4, 5, 100, 9, 12, 13, 17, 90}

{0, 1, 2, 4, 5, 100, 9, 12, 13, 17, 90}

Dari kedua contoh penggunanaan union kita dapat mengetahui bahwa union dapat dituliskan sebagai

```
(set1.union(set2))
```

Maupun menggunakan lambang (|)

```
(set1|set2|set3)
```

Contoh penggunaan intersection:

```
set1={1,4,5,2,12}
set2={9,0,13,1,2,5,17,90}
set3={1,100}
print(set1.intersection(set2))
```

Output : {1, 2, 5}

Penggunaan (&) dalam intersection :

```
set1={1,4,5,2,12}
set2={9,0,13,1,2,5,17,90}
set3={1,100}
print(set1.intersection(set2))
print(set1&set2&set3)
```

Output : {1, 2, 5} , {1}

Intersection berguna untuk mengambil value yang terdapat di 2/lebih himpunan ketika di intersection ketiganya maka nilai 1 saja yang keluar karena value satu berada di ketiga set tersebut.

Penggunaan difference :

```
set1={1,4,5,2,12}
set2={9,0,13,1,2,5,17,90}

print(set1.difference(set2))
```

Output : {4, 12}

Contoh penggunaan difference memakai oprasi (-):

```
set1={1,4,5,2,12}
set2={9,0,13,1,2,5,17,90}
print(set1-set2)
```

Output : {4, 12}

Oprasi difference ini akan menampilkan hasil Dimana anggota himpunan tidak ada di himpunan lainnya dalam kasus diatas maka yang ditampilkan ialah anggota himpunan set 1 yang tidak ada di himpunan set 2

Symmetric difference :

```
set1={1,4,5,2,12}
set2={9,0,13,1,2,5,17,90}
print(set1.symmetric_difference(set2))
print(set1^set2)
```

Output : {0, 4, 9, 12, 13, 17, 90},{0, 4, 9, 12, 13, 17, 90}

Pengertian symmetric difference ialah menampilkan himpunan yang tidak termasuk dalam irisan.

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

Kita diminta memodifikasi code dari contoh Latihan diatas agar dapat menampilkan nama-nama aplikasi yang hanya muncul di satu kategori saja. dan Untuk input n>2, menampilkan nama-nama aplikasi yang muncul tepat di dua kategori sekaligus

```
# input n kategori
n = int(input('Masukkan jumlah kategori: '))
# siapkan dictionary kosong
data_aplikasi = {}
#input nama kategori & 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(4):
        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(f"semua daftar applikasi : {daftar_aplikasi_list}")


# intersection semua kategori
hasil_1 = daftar_aplikasi_list[0]
for i in range(1, len(daftar_aplikasi_list)):
    hasil_1 = hasil_1.intersection(daftar_aplikasi_list[i])
print(f"aplikasi yang muncul di seluruh kategori:  {hasil_1}")
```

```
# yang muncul di 1 kategori saja
hasil_2 = daftar_aplikasi_list[0]
for i in range(1, len(daftar_aplikasi_list)):
    hasil_2 = hasil_2.symmetric_difference(daftar_aplikasi_list[i])

print(f"aplikasi yang hanya muncul di 1 katgori: {hasil_2-hasil_1}")

#yang muncul tepat di 2 kategori
for i in range(0, len(daftar_aplikasi_list)-1):
    tot=(daftar_aplikasi_list[i]&daftar_aplikasi_list[i+1])-hasil_1
tot=tot|(daftar_aplikasi_list[0]&daftar_aplikasi_list[len(daftar_aplikasi_list)-
1])-hasil_1
print (f"aplikasi yang muncul tepat di 2 kategori {tot}")
```

Output :

Masukkan jumlah kategori: 3

Masukkan nama kategori:1

Masukkan 5 nama aplikasi di kategori 1

Nama aplikasi: a

Nama aplikasi: b

Nama aplikasi: c

Nama aplikasi: d

Nama aplikasi: e

Masukkan nama kategori:2

Masukkan 5 nama aplikasi di kategori 2

Nama aplikasi: a

Nama aplikasi: k

Nama aplikasi: l

Nama aplikasi: p

Nama aplikasi: o

Masukkan nama kategori:3

Masukkan 5 nama aplikasi di kategori 3

Nama aplikasi: a

Nama aplikasi: c

Nama aplikasi: k

Nama aplikasi: r

Nama aplikasi: t

{'1': ['a', 'b', 'c', 'd', 'e'], '2': ['a', 'k', 'l', 'p', 'o'], '3': ['a', 'c', 'k', 'r', 't']}

semua daftar applikasi : [{'c', 'd', 'a', 'b', 'e'}, {'l', 'p', 'a', 'k', 'o'}, {'c', 'a', 'k', 'r', 't'}]

aplikasi yang muncul di seluruh kategori: {'a'}

aplikasi yang hanya muncul di 1 katgori: {'l', 'p', 'd', 'o', 'b', 'r', 't', 'e'}

aplikasi yang muncul tepat di 2 kategori {'c', 'k'}aplikasi yang muncul tepat di 2 kategori {'j'}

Penjelasan : pada code yang dibuat kita diminta memodifikasi agar dapat memunculkan aplikasi yang tepat di 2 kategori saja dan yang muncul di 1 kategori saja untuk itu maka modifikasinya ialah

```
# intersection semua kategori
hasil_1 = daftar_aplikasi_list[0]
for i in range(1, len(daftar_aplikasi_list)):
    hasil_1 = hasil_1.intersection(daftar_aplikasi_list[i])
print(f"aplikasi yang muncul di seluruh kategori:  {hasil_1}")

# yang muncul di 1 kategori saja
hasil_2 = daftar_aplikasi_list[0]
for i in range(1, len(daftar_aplikasi_list)):
    hasil_2 = hasil_2.symmetric_difference(daftar_aplikasi_list[i])

print(f"aplikasi yang hanya muncul di 1 katgori: {hasil_2-hasil_1}")

#yang muncul tepat di 2 kategori
for i in range(0, len(daftar_aplikasi_list)-1):
    tot=(daftar_aplikasi_list[i]&daftar_aplikasi_list[i+1])-hasil_1
tot=tot|(daftar_aplikasi_list[0]&daftar_aplikasi_list[len(daftar_aplikasi_list)-
1])-hasil_1
print (f"aplikasi yang muncul tepat di 2 kategori {tot}")
```

Dimana pertama tam akita perlu untuk mengetahui aplikasi mana yang muncul di seluruh kategori dnegan cara

```
hasil_1 = daftar_aplikasi_list[0]
for i in range(1, len(daftar_aplikasi_list)):
    hasil_1 = hasil_1.intersection(daftar_aplikasi_list[i])
```

```
print(f"aplikasi yang muncul di seluruh kategori:  {hasil_1}")
```

pada code ini ada perulangan untuk meng intersection keseluruhan list dalam daftar aplikasi setelah itu maka hasilnya ialah aplikasi yang muncul di seluruh kategori, setelahnya maka kita mencari yang muncul pada satu kategori dengan code :

```
hasil_2 = daftar_aplikasi_list[0]
for i in range(1, len(daftar_aplikasi_list)):
    hasil_2 = hasil_2.symmetric_difference(daftar_aplikasi_list[i])

print(f"aplikasi yang hanya muncul di 1 katgori: {hasil_2-hasil_1}")
```

memakai symmetric difference untuk mengecek semuanya dalam perulangan dan dikurangi dengan aplikasi yang ada di seluruh kategori, karena ketika dipakai dalam function symmetric_difference ke semua list nya maka akan kelebihan contoh : 'l', 'p', 'd', 'o', 'b', 'r', 't', 'e',”a” maka dari itu dikurangi lagi dengan “a” / aplikasi yang muncul di seluruh kategori, maka dari itu harus di kurangi dengan hasil_1

setelah itu kita cari yang muncul di 2 kategori saja dengan code :

```
for i in range(0, len(daftar_aplikasi_list)-1):
    tot=(daftar_aplikasi_list[i]&daftar_aplikasi_list[i+1])-hasil_1
tot=tot|(daftar_aplikasi_list[0]&daftar_aplikasi_list[len(daftar_aplikasi_list)-
1])-hasil_1
print (f"aplikasi yang muncul tepat di 2 kategori {tot}")
```

code ini sederhanyanya mengambil irisan dari indeks[0] dan [1] lalu dikurangi hasil_1, irisan indeks[1] dan indeks[2] dikurangi hasil_1, dan terakhir ditambahkan irisan indeks[0] dan indeks [2] dikurangi hasil_1

setelahnya ditampilkan sesuai dengan ketentuan.

SOAL 2

Input :

```
# 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.

# List menjadi Set
ini_list=[1,2,3,10,454,5,62,7]
print(ini_list)
print(type(ini_list))
```

```
list_jadi_set=set(ini_list)
print(list_jadi_set)
print(type(list_jadi_set))

# Set menjadi List
ini_set={1,5,7,1,3,8,0,3,24,12}
print(ini_set)
print(type(ini_set))

set_jadi_list=list(ini_list)
print(set_jadi_list)
print(type(set_jadi_list))

# Tuple menjadi Set
ini_tuple=(1,4,5,7,2,3,)
print(ini_tuple)
print(type(ini_tuple))

tuple_jadi_set=set(ini_tuple)
print(tuple_jadi_set)
print(type(tuple_jadi_set))

# set menjadi tuple
ini_set_2={1,4,5,7,2,4,5,1,4,89}
print(ini_set_2)
print(type(ini_set_2))

set_jadi_tuple=tuple(ini_set_2)
print(set_jadi_tuple)
print(type(set_jadi_tuple))
```

Output :

[1, 2, 3, 10, 454, 5, 62, 7]

<class 'list'>

{1, 2, 3, 5, 454, 7, 10, 62}

<class 'set'>

{0, 1, 3, 5, 7, 8, 12, 24}

<class 'set'>

[1, 2, 3, 10, 454, 5, 62, 7]

<class 'list'>

(1, 4, 5, 7, 2, 3)

<class 'tuple'>

{1, 2, 3, 4, 5, 7}

<class 'set'>

{1, 2, 4, 5, 7, 89}

<class 'set'>

(1, 2, 4, 5, 7, 89)

<class 'tuple'>

Penjelasan :

Kita konversi tipe data memakai tipe di depan lalu tanda kurung contoh set(variable) maka akan menjadi tipe data set dan list(variable) maka variable akan menjadi tipe data list.

Dan ditampilkan sebagai output

SOAL 3

Input :

```
def cek_2_file():
    file1 = input("Masukkan nama file pertama: ")
    try:
        with open(file1, 'r') as file:
            isi = file.read()
            isi=isi.lower()
            isi=isi.split(' ')
            kata_kata_1 = (isi)
    except:
        print(f"File '{file1}' tidak ditemukan.")

    file2 = input("Masukkan nama file kedua: ")
    try:
        with open(file2, 'r') as file:
            isi = file.read()
            isi=isi.lower()
            isi=isi.split(' ')
            kata_kata_2= (isi)
```

```
    except:
        print(f"File '{file1}' tidak ditemukan.")


    kata_sama = set(kata_kata_1) & set(kata_kata_2)

    if kata_sama:
        print("kata yang muncul pada kedua file:")
        for word in kata_sama:
            print(word)
    else:
        print("Tidak ada kesamaan kata pada kedua file.")

cek_2_file()
```

Output :

Masukkan nama file pertama: text2.txt

Masukkan nama file kedua: text1.txt

kata yang muncul pada kedua file:

jenis

bentuk

files

untuk

yang

dapat

atau

dan

merupakan

kita

data

flat

penjelasan :

memninta input pengguna untuk nama filenya saat bisa dibuka maka file langsung di read dijadikan lowercase dan di split berdasarkan spasi untuk mencari per katanya

setelah itu dibandingkan tiap kata pada file pertama dan kedua memakai intersection dan ditampilkan di output apabila ada persamaan kata di kedua filenya.

Source : https://github.com/vianchr/71230972_data_set