Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71230978**
Nama Lengkap	**Jonathan Satriani Gracio Andrianto**
Minggu ke / Materi	**12 / Tipe Data Tuple**

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%)

Tuple Immutable

Tuple dapat dianggap mirip dengan list. Nilai yang disimpan dalam tuple bisa berupa apa saja dan diberi indeks berupa bilangan bulat (integer). Perbedaan utamanya adalah tuple bersifat immutable, artinya anggota dalam tuple tidak bisa diubah. Tuple juga dapat dibandingkan (compare) dan bersifat hashable, sehingga bisa digunakan sebagai key dalam dictionary di Python.

Sintaks penulisan tuple :

```
t = 'a','b','c','d','e'
```

Tuple juga dapat ditulis menggunakan tanda kurung :

```
t = ('a','b','c','d','e')
```

Tuple dengan satu element, ditambahkan koma (,) dibelakang penulisan tuple.

```
t1 = ('a',)
print(type(t1)) # output = <class 'tuple'>
```

Jika tidak menanbahkan tanda koma, akan dianggap sebagai string.

```
t1 = ('a')
print(type(t1)) # output = <class 'str'>
```

Model sintaks lain dengan menggunakan fungsi built-in tuple. Ketika tidak diisi dengan argument apapun, secara langsung akan membentuk tuple kosong.

```
t = tuple()
print(t) #output = ()
```

Jika argumennya berupa urutan (string, list, atau tuple), akan mengembalikan nilai tuple dengan elemen-elemen yang berurutan.

```
t = tuple('jodahganteng')
print(t) #output = ('j', 'o', 'd', 'a', 'h', 'g', 'a', 'n', 't', 'e', 'n', 'g')
```

Tanda kurung kotak [ ] menandakan indeks element dari tuple.

```
t = tuple('jodahganteng')
print(t[0]) #output = j
```

Untuk menampilkan rentang nilai dari element tuple dapat menggunakan :

```
t = tuple('jodahganteng')
print(t[1:5]) #output = ('o', 'd', 'a', 'h')
```

Tuple bersifat immutable artinya element pada tuple tidak dapat berubah. Jika anda berusaha mengubah tuple, maka akan didapatkan error.

```
t = tuple('jodahganteng')
t[0] = "c" #output = TypeError: 'tuple' object does not support item assignment
```

Element pada tuple tidak dapat dirubah tetapi dapat diganti dengan elemen lain.

```
t = tuple('jodahganteng')
t = ('w',) + t[1:]
print(t) #output = ('w', 'o', 'd', 'a', 'h', 'g', 'a', 'n', 't', 'e', 'n', 'g')
```

Membandingkan Tuple

Operator perbandingan (compare) dapat digunakan pada tuple dan struktur sekuensial lainnya seperti list, dictionary, dan set. Cara kerjanya adalah dengan membandingkan elemen pertama dari setiap sekuensial. Jika elemen pertama sama, perbandingan akan dilanjutkan ke elemen berikutnya. Proses ini terus berlanjut sampai ditemukan perbedaan di antara elemen-elemen tersebut.

```
print((0, 1, 2) < (0, 3, 4)) #output = True
print((0, 1, 2000000) < (0, 3, 4)) #output = True
```

Fungsi sort pada Python bekerja dengan cara yang serupa. Tahap awal mengurutkan berdasarkan elemen pertama. Dalam beberapa kasus, pengurutan akan dilanjutkan berdasarkan elemen kedua dan seterusnya. Metode ini dikenal sebagai DSU – (Decorate, Sort, Undecorate).

- Decorate : Urutan (sekuensial) membentuk daftar tuple dengan satu atau lebih key pengurutan sebelum elemen-elemen dari urutan tersebut.
- Sort : List tuple menggunakan sort (fungsi bawaan di python)
- Undercorate : Mengekstrak elemen yang telah diurutkan kembali ke dalam urutan sekuensial.

Untuk memahaminya, perhatikan contoh di bawah ini. Kita memiliki sebuah kalimat dan akan mengurutkan kata-kata dalam kalimat tersebut dari yang terpanjang hingga yang terpendek.

```
kalimat = 'but soft what light in yonder window breaks'
dafkata = kalimat.split()
t = list()
for kata in dafkata:
    t.append((len(kata), kata))
```

```
t.sort(reverse=True)

urutan = list()
for length, kata in t:
    urutan.append(kata)

print(urutan) #output = ['yonder', 'window', 'breaks', 'light', 'what', 'soft',
'but', 'in']
```

Pada iterasi pertama, akan dibuat daftar tuple yang berisi kata-kata beserta panjangnya. Fungsi sort akan membandingkan elemen pertama dari daftar berdasarkan panjang kata, dan jika ada kesamaan, perbandingan akan berlanjut ke elemen kedua. Penggunaan keyword reverse=True memastikan pengurutan dilakukan secara menurun (descending). Iterasi kedua akan menyusun daftar tuple dalam urutan alfabet berdasarkan panjang kata. Dalam contoh di atas, empat kata dalam kalimat akan diurutkan dalam urutan alfabet terbalik. Misalnya, kata "what" akan muncul sebelum "soft" dalam daftar.

Penugasan Tuple

Salah satu fitur unik Python adalah kemampuannya menempatkan tuple di sisi kiri dari pernyataan penugasan. Hal ini memungkinkan penetapan beberapa variabel sekaligus secara berurutan di sisi kiri. Misalnya, jika ada dua daftar elemen yang berurutan, kita bisa menetapkan elemen pertama dan kedua dari urutan tersebut ke dalam variabel x dan y dalam satu pernyataan.

```
m = [ 'jodah', 'ganteng' ]
x, y = m
print(x) #jodah
print(y) #ganteng
```

Python akan menterjemahkan sintaks tuple dalam langkah-langkah sebagai berikut:

```
m = [ 'jodah', 'ganteng' ]
x = m[0]
y = m[1]
print(x) #jodah
print(y) #ganteng
```

Tuple yang digunakan pada contoh di atas memakai pernyataan penugasan di sebelah kiri tanpa tanda kurung. Berikut ini adalah contoh yang sama dengan menggunakan tanda kurung (parentheses) :

```
m = [ 'jodah', 'ganteng' ]
(x, y) = m
print(x) #jodah
print(y) #ganteng
```

Dictionaries and Tuple

Dictionaries memiliki metode bernama items yang mengembalikan daftar tuple, di mana setiap tuple adalah pasangan key-value (kunci dan nilai). Seperti pada dictionary umumnya, item yang dikembalikan tidak berurutan. Namun, karena daftar tuple tersebut adalah sebuah list dan tuple dapat dibandingkan, kita bisa melakukan pengurutan (sort) pada tuple-tuple tersebut. Mengonversi dictionary menjadi daftar tuple adalah cara untuk menampilkan isi dictionary yang diurutkan berdasarkan kuncinya.

```
d = {'a':2, 'b':67, 'c':82}
t = list(d.items())
print(t) #output = [('a', 2), ('b', 67), ('c', 82)]
```

Multipenugasan dengan dictionaries

Kombinasi antara items, tupple assignment dan for akan menghasilkan kode tertentu pada keys dan values dari dictionary dalam satu loop.

```
d = {'a':2, 'b':67, 'c':82}
for key, val in list(d.items()):
    print(val, key)
'''''
output =
2 a
67 b
82 c
'''''
```

Pada bagian looping ini, terdapat dua variabel iterasi karena items mengembalikan daftar tuple. Penugasan tuple `key, val` melakukan iterasi berulang melalui pasangan key-value pada dictionary. Pada setiap iterasi melalui loop, baik key maupun value akan bergerak maju ke pasangan key-value berikutnya dalam dictionary (masih dalam urutan hash). Dengan menggabungkan kedua teknik di atas, kita bisa mencetak isi dictionary yang diurutkan berdasarkan nilai yang disimpan di setiap pasangan key-value. Langkah pertama adalah membuat daftar tuple di mana masing-masing tuple berisi (value, key). Metode items akan memberikan daftar tuple (value, key) yang akan diurutkan berdasarkan value, bukan key. Setelah daftar tuple value-key terbentuk, langkah berikutnya adalah mengurutkan daftar tersebut secara terbalik dan mencetak daftar baru yang telah diurutkan.

Kata yang sering muncul

Pada bagian ini, kita akan mencoba menampilkan kata-kata yang paling sering muncul dalam teks Romeo and Juliet Act 2, Scene 2. Kita akan membuat program yang menggunakan daftar tuple untuk menampilkan 10 kata yang paling sering muncul.

```
import string
fhand = open('romeo-full.txt')
counts = dict()
```

```
for line in fhand:
    line = line.translate(str.maketrans('', '', string.punctuation))
    line = line.lower()
    words = line.split()
    for word in words:
        if word not in counts:
            counts[word] = 1
        else:
            counts[word] += 1

# urutkan dictionary by value
lst = list()
for key, val in list(counts.items()):
    lst.append((val, key))

lst.sort(reverse=True)
for key, val in lst[:10]:
    print(key, val)
'''
output
34 to
34 the
32 thou
32 juliet
30 that
29 my
24 thee
'''
```

Bagian pertama dari program ini digunakan untuk membaca file dan menghitung jumlah kemunculan setiap kata, menyimpannya dalam sebuah dictionary yang memetakan setiap kata ke jumlah kemunculannya dalam dokumen. Dengan menambahkan perintah print untuk menampilkan counts, daftar tuple (val, key) dibuat dan diurutkan dalam daftar secara terbalik. Perbandingan dilakukan berdasarkan nilai pertama. Jika ada lebih dari satu tuple dengan nilai yang sama, maka akan dilihat elemen kedua (key), sehingga tuple dengan nilai yang sama akan diurutkan berdasarkan abjad sesuai key.

Tuple sebagai kunci dictionaries

Tuple adalah objek yang bersifat hashable, sedangkan list tidak. Ketika kita ingin membuat kunci (key) komposit yang digunakan dalam dictionary, kita bisa menggunakan tuple. Sebagai contoh, jika kita ingin membuat sebuah direktori telepon yang memetakan dari pasangan last-name, first-name ke nomor telepon, dengan asumsi bahwa kita sudah mendefinisikan variabel last, first, dan nomor. Penulisan pernyataan penugasan dalam dictionary sebagai berikut :

```
directory[last,first] = number
```

Ekspresi di dalam kurung kotak adalah sebuah tuple. Langkah selanjutnya adalah memberikan penugasan tuple tersebut pada loop for yang terkait dengan dictionary.

```
last = 'jo'
first = 'ganteng'
number = '086988822'
directory = dict()
directory[last, first] = number
for last, first in directory:
    print(first, last, directory[last,first]) #ganteng jo 086988822
```

BAGIAN 2: LATIHAN MANDIRI (60%)

SOAL 1

Source Code

```
def sama_gak_yah(t):
    return len(set(t)) == 1

t1 = (90, 90, 90, 90)
t2 = (90, 90, 90, 80)
print(sama_gak_yah(t1))
print(sama_gak_yah(t2))
```

Output

```
True
False
```

Penjelasan

‘return len(set(t)) == 1` bertujuan untuk memeriksa apakah semua elemen dalam tuple `t` memiliki nilai yang sama. Ini dilakukan dengan mengubah tuple menjadi set (untuk menghilangkan duplikat) dan kemudian memeriksa apakah panjang set tersebut adalah 1, yang berarti semua elemennya sama.

SOAL 2

Source Code

```
def data_diri(nim, nama, alamat) :
    data = nama, nim, alamat
    print(data)
    print()
    print(f"NIM     : {nim}")
    print(f"Nama    : {nama}")
    print(f"Alamat  : {alamat}")
    print()
    print(f"Nim : {tuple(nim)}")
    print(f"Nama Depan : {tuple(nama.split()[0])}")
    print(f"Nama Terbalik : {tuple(nama.split()[::-1])}")


nim = "22064091"
nama = "Matahari Bhakti Nendya"
alamat = "Bantul, DI Yogyakarta"
data_diri(nim, nama, alamat)
```

Output

```
('Matahari Bhakti Nendya', '22064091', 'Bantul, DI Yogyakarta')

NIM     : 22064091
Nama    : Matahari Bhakti Nendya
Alamat  : Bantul, DI Yogyakarta

Nim : ('2', '2', '0', '6', '4', '0', '9', '1')
Nama Depan : ('M', 'a', 't', 'a', 'h', 'a', 'r', 'i')
Nama Terbalik : ('Nendya', 'Bhakti', 'Matahari')
```

Penjelasan

Pertama kita buat fungsi data_diri kemudian kita masukkan parameter (nim, nama, alamat). Kemudian kita buat variable baru yaitu data yang nantinya akan menjadi tuple paling atas. Kemudian kita cetak nim, nama, dan alamat. Selanjutnya kita buat tuple dari nim. Untuk nama depan kita gunakan fungsi ‘(nama.split()[0])’ yang berfungsi untuk memisahkan nama berdasarkan spasi dan mengambil index 0 yaitu nama depan. Untuk nama terbalik kita gunakan fungsi ‘nama.split()[::-1]’ yang memisahkan nama berdasarkan spasi dan menampilkannya mulai dari index -1(yang paling belakang) sampai ke nama depan.

SOAL 3

Souce Code

```
def distribusi_jam(file_):
    distribusi = {}
    with open(file_, 'r') as file:
        for line in file:
            if line.startswith('From '):
                kata = line.split()
                waktu = kata[5].split(':')
                jam = waktu[0]
                distribusi[jam] = distribusi.get(jam, 0) + 1

    print("distribusi jam :")
    for jam in sorted(distribusi):
        print(jam, distribusi[jam])

file_ = "mbox-short.txt"
distribusi_jam(file_)
```

Output

```
distribusi jam :
04 3
06 1
07 1
09 2
10 3
11 6
14 1
15 2
16 4
17 2
18 1
19 1
```

Penjelasan

Jadi pertama kita buat fungsi distribusi_jam, kemudian kita buka file mbox-short.txt dengn mode rread. Setelah itu kita periksa satu per satu baris, jika ditemukan “From :” maka akan dipisahkan baris tersebut berdasar spasi dan diletakan dalam variable kata. Variabel kata index ke lima akan dipisah lagi berdasar titik dua dan yang di bagian kiri titik dua yang diletakan pada variable jam. Kemudian kita masukkan variable jam kedalam tuple distribusi dan kita hitung juga frekuensi kemunculannya. Setelah itu kita buat looping untuk menampilkan distribusi jam.

Link Github

https://github.com/JonathanAndrianto123/alpro_12_71230978.git