



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	<Deo Dewanto >
Nama Lengkap	<71230981 >
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%)

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.

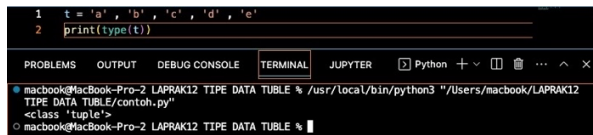
MATERI 1

Tuple Immutable

Tuple mirip dengan list karena keduanya dapat menyimpan nilai apa pun dan diindeks dengan bilangan bulat. Namun, perbedaan utamanya terletak pada sifat tuple yang tidak dapat diubah (immutable). Ini berarti bahwa setelah sebuah tuple dibuat, nilainya tidak dapat diubah lagi. Tuple juga dapat dibandingkan dan bersifat hashable, yang berarti dapat digunakan sebagai kunci dalam dictionary. Python.

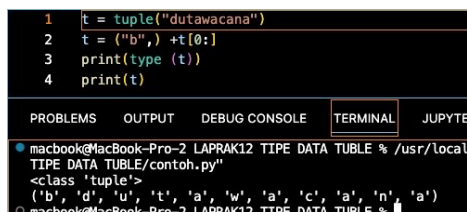
Objek disebut hashable jika nilai hashnya tetap tidak berubah selama hidupnya dan dapat dibandingkan dengan objek lain. Kemampuan hashability ini memungkinkan objek untuk digunakan sebagai kunci dalam kamus Python, karena penggunaan nilai hash secara internal. Objek bawaan Python umumnya hashable, sementara wadah yang tidak dapat diubah (seperti list atau dictionary) tidak. Objek yang merupakan instance dari kelas yang didefinisikan pengguna secara default hashable; mereka membandingkan yang tidak sama dan nilai hashnya adalah `id()` objek tersebut.

```
1 t = 'a', 'b', 'c', 'd', 'e'
2 print(type(t))
```



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

```
1 t = tuple("dutawacana")
2 t = ("b",) + t[0:]
3 print(type(t))
4 print(t)
```



Kode tersebut mengubah string "dutawacana" menjadi sebuah tuple yang berisi setiap karakter dalam string tersebut, sehingga menghasilkan ('d', 'u', 't', 'a', 'w', 'a', 'c', 'a', 'n', 'a'). Kemudian, elemen 'b' ditambahkan di awal tuple ini, menghasilkan tuple baru ('b', 'd', 'u', 't', 'a', 'w', 'a', 'c', 'a', 'n', 'a'). Setelah itu, tipe dari t3 dicetak, yang menunjukkan bahwa t3 adalah <class 'tuple'>, dan isi dari t3 juga dicetak, yang menunjukkan tuple yang telah dimodifikasi. Hasil akhir yang ditampilkan adalah tipe tuple dan isinya yaitu ('b', 'd', 'u', 't', 'a', 'w', 'a', 'c', 'a', 'n', 'a').

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.

```
1 t = (0, 1, 2) < (0, 3, 4)
2 t = (0, 1, 2000000) < (8, 3, 4)
3 print(t)
4 print(t)
```

PROBLEMS OUTPUT DEBUG CONSOLE

macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE
TIPE DATA TUBLE/contoh.py"
True
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.

```
1 kalimat = 'but soft what light in yonder window breaks'
2 dafkata = kalimat.split()
3 t = list()
4 for kata in dafkata:
5     t.append((len(kata), kata))
6 t.sort(reverse=True)
7
8 urutan = list()
9 for length, kata in t:
10     urutan.append(kata)
11 print(urutan)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE % /usr/local/bin/python
TIPE DATA TUBLE/contoh.py"
['yonder', 'window', 'breaks', 'light', 'what', 'soft', 'but', 'in']
macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE %

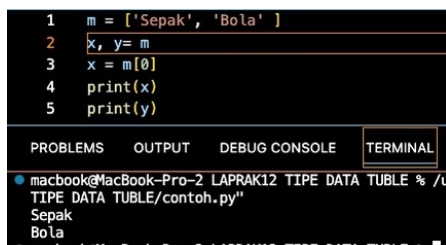
Pada iterasi pertama, akan dibuat daftar tuple yang berisi kata-kata beserta panjangnya. Fungsi sort akan membandingkan elemen pertama dari daftar berdasarkan panjang kata, dan jika ada kesamaan perbandingan akan berlanjut ke elemen kedua. Penggunaan keyword reverse True

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 untuk memiliki tuple di sisi kiri dari pernyataan penugasan. 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.

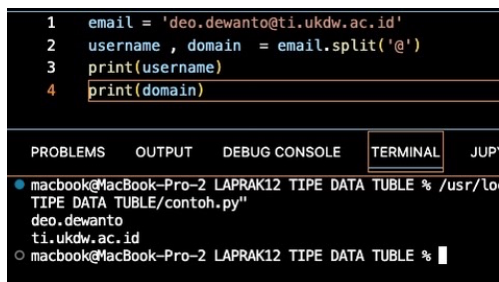
```
1 m = ['Sepak', 'Bola']
2 x, y = m
3 x = m[0]
4 print(x)
5 print(y)
```



macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUPLE % /u
TIPE DATA TUPLE/contoh.py"
Sepak
Bola
macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUPLE %

Dari contoh diatas, dapat dilihat bahwa kedua statemnt merupakan tuple. Bagian kiri merupakan tuple dari variable dan bagian kanan merupakan tuple dari expresions. Tiap nilai pada bagian kanan diberikan/ditugaskan ke masing-masing variable di sebelah kiri. Semua expresions pada bagian kiri dievaluasi sebelum diberikan penugasan.

```
1 email = 'deo.dewanto@ti.ukdw.ac.id'
2 username, domain = email.split('@')
3 print(username)
4 print(domain)
```



macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUPLE % /usr/loc
TIPE DATA TUPLE/contoh.py"
deo.dewanto
ti.ukdw.ac.id
macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUPLE %

Dictionaries and Tuple

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

```
1 d={'a':15, 'b':20, 'c':25}
2 t = list(d.items())
3 print(t)
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL**

```
macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE % /us
TIPE DATA TUBLE/contoh.py"
[('a', 15), ('b', 20), ('c', 25)]
macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE %
```

Multipenugasan Dengan Dictionaries

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

```
1 d = {'a': 15, 'b':20, 'c':25}
2
3 for key, val in list(d.items()):
4     print(val, key)
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL**

```
macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE % /us
TIPE DATA TUBLE/contoh.py"
15 a
20 b
25 c
macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE %
```

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 – 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.

```
1 import string
2 fhand = open('romeo-full.txt')
3 counts = dict()
4 for line in fhand:
5     line = line.translate(str.maketrans('', '', string.punctuation))
6     line = line.lower()
7     words = line.split()
8     for word in words:
9         if word not in counts:
10             counts[word] = 1
11         else:
12             counts[word] += 1
13 lst = list()
14 for key, val in list(counts.items()):
15     lst.append((val, key))
16
17 lst.sort(reverse=True)
18 for key, val in lst[:10]:
19     print(key, val)
```

```

macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE % /usr/local/bin/python3 "/Users/macbook/LAPRAK12
TIPE DATA TUBLE/contoh.py"
61 i
42 and
40 romeo
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.

Tubel 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.

```

1 last = 'dewanto'
2 first = 'deo'
3 number = '71230981'
4 directory = dict()
5 directory[last, first] = number
6 for last, first in directory:
7     print(first, last, directory[last,first])

```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** Python + - [] [] ... ^

```

macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE % /usr/local/bin/python3 "/Users/macbook/LAPRAK12
TIPE DATA TUBLE/contoh.py"
deo dewanto 71230981
macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE %

```

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

Source code and Output

```
#Latihan 1
def cek(t):
    | return len(set(t)) == 1

tA = (90,90,90,90)
print(cek(tA))

tb = (90,90,60,90)
print(cek(tb))
```

[5] ✓ 0.0s Python

... True
False

Penjelasan :

Fungsi cek(t) digunakan untuk memeriksa apakah semua elemen dalam tuple t adalah sama. Fungsi Ini bekerja dengan mengonversi tuple menjadi set, yang secara otomatis menghapus elemen duplikat, kemudian memeriksa apakah panjang set tersebut adalah 1. Jika panjangnya 1, itu berarti semua elemen dalam tuple adalah identik. Misalnya, untuk tuple tA yang berisi (90, 90, 90, 90), fungsi cek(tA) akan mengembalikan True karena semua elemen adalah 90. Sebaliknya, untuk tuple tb yang berisi (90, 90, 60, 90), fungsi cek(tb) akan mengembalikan False karena ada elemen yang berbeda (60),

SOAL 2

Source code and Output

```
#Latihan 2
def data(nama_lengkap, nim, alamat):
    gabung = nama_lengkap, nim, alamat
    print("Data:", tuple(gabung))
    print("\n")
    print("NIM      :", ' '.join(nim))
    print("NAMA      :", ' '.join(nama_lengkap))
    print("ALAMAT     :", alamat)
    print('\n')
    print("NIM:", tuple(nim))
    print("NAMA DEPAN:", tuple(nama_lengkap.split()[0]))
    print("NAMA TERBALIK:", tuple(nama_lengkap.split()[::-1]))

nama_lengkap = 'Deo Dewanto'
nim = '71230981'
alamat = 'Toraja, Sulawesi Selatan'

data(nama_lengkap, nim, alamat)
```

✓ 0.0s Python

Data: ('Deo Dewanto', '71230981', 'Toraja, Sulawesi Selatan')

NIM : 7 1 2 3 0 9 8 1
NAMA : D e o D e w a n t o
ALAMAT : T o r a j a , S u l a w e s i S e l a t a n

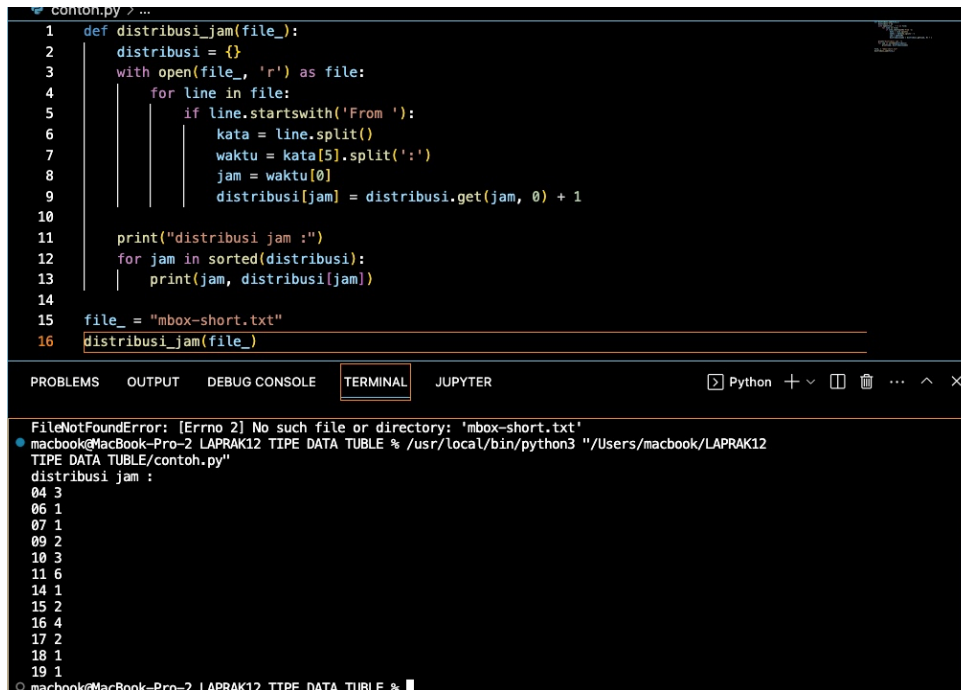
NIM: ('7', '1', '2', '3', '0', '9', '8', '1')
NAMA DEPAN: ('D', 'e', 'o')
NAMA TERBALIK: ('Dewanto', 'Deo')

Penjelasan:

Fungsi data menerima tiga parameter: nama lengkap, nim, dan alamat user . Fungsi ini menggabungkan ketiga parameter tersebut menjadi satu tuple dan mencetaknya. Selanjutnya, fungsi mencetak NIM, nama lengkap, dan alamat dengan setiap karakter NIM dan nama lengkap dipisahkan oleh spasi. Kemudian, fungsi mencetak NIM sebagai tuple, nama depan sebagai tuple (diambil dari kata pertama nama lengkap), dan nama lengkap secara terbalik sebagai tuple (urutan kata-kata dalam nama lengkap dibalik).

SOAL 3

Source code and Output



```
1 def distribusi_jam(file_):
2     distribusi = {}
3     with open(file_, 'r') as file:
4         for line in file:
5             if line.startswith('From '):
6                 kata = line.split()
7                 waktu = kata[5].split(':')
8                 jam = waktu[0]
9                 distribusi[jam] = distribusi.get(jam, 0) + 1
10
11     print("distribusi jam :")
12     for jam in sorted(distribusi):
13         print(jam, distribusi[jam])
14
15 file_ = "mbox-short.txt"
16 distribusi_jam(file_)
```

FileNotFoundError: [Errno 2] No such file or directory: 'mbox-short.txt'

```
macbook@MacBook-Pro-2 LAPRAK12 TIPE DATA TUBLE % /usr/local/bin/python3 "/Users/macbook/LAPRAK12
TIPE DATA TUBLE/contoh.py"
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 tama kita membuat fungsi distribusi jam, kemudian kita buka file mbox-short.txt dengan mode read. 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/Deodewanto07/LAPRAK12.git>