Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71200645**
Nama Lengkap	**Edith Felicia Putri**
Minggu ke / Materi	**11 / 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
2023

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 bisa dikatakan menyerupai *list*. Nilai yang disimpan dalam *tuple* dapat berupa apapun dan diberikan indeks bilangan bulat (*integer*). Tipe data *sequence* yang digunakan untuk menampung nilai kolektif dimana perbedaan utama *list* dengan *tuple* terletak pada sifat *tuple* yang *immutable*, yaitu elemen dalam *tuple* tidak bisa diubah, tetapi bisa diganti dengan elemen lain berdasarkan *index*. *Tuple* sendiri dapat dibandingkan (*compare*) dan bersifat *hashable*, sehingga dapat dimasukkan dalam *list* sebagai *key* pada *dictionary* python.

```
t1 = ("Kristus")
print(type(t1))
t2 = ("Yesus",)
print(type(t2))
```

SOURCE CODE:

```
PS D:\71200645_PrakAlpro_11> & C:/Users/HP/AppData/Local/Programs/Python/Python311/python.exe d:/71200645_PrakAlpro_11/modul.py
<class 'str'>
<class 'tuple'>
PS D:\71200645_PrakAlpro_11>
```

Cara membuat Tuple

```
tuple_1 = ()
tuple_2 = tuple()
```

Mengakses Tuple

```
kata = ("Akhir", "Zaman", "Tiba")
print(kata[1])
```

OUTPUT: “Zaman”

Menggabungkan Tuple

```
tuple_1 = "Yesus", "mengasihi", "kamu"
tuple_2 = "selama", "lamanya"
tuple_3 = (tuple_1, tuple_2)
print(tuple_3)
```

OUTPUT:

```
PS D:\71200645_PrakAlpro_11> & C:/Users/HP/AppData/Local/Program
s/Python/Python311/python.exe d:/71200645_PrakAlpro_11/modul.py
(('Yesus', 'mengasihi', 'kamu'), ('selama', 'lamanya'))
PS D:\71200645_PrakAlpro_11>
```

```
a = (1, 2, 3)
b = (50, 60, 70)
c = a + b
print(c)
```

OUTPUT:

```
PS D:\71200645_PrakAlpro_11> & C:/Users/HP/AppData/Local/Programs
/Python/Python311/python.exe d:/71200645_PrakAlpro_11/modul.py
(1, 2, 3, 50, 60, 70)
PS D:\71200645_PrakAlpro_11>
```

Menghapus Tuple

```
daftar = ("Dosa", "Dunia", 2000, 2024)
print(daftar)
del daftar

daftar = ("Hidup", "Suci", 2024)
print("Daftar baru:", daftar)
```

OUTPUT:

```
PS D:\71200645_PrakAlpro_11> & C:/Users/HP/AppData/Local/Programs
/Python/Python311/python.exe d:/71200645_PrakAlpro_11/modul.py
('Dosa', 'Dunia', 2000, 2024)
Daftar baru: ('Hidup', 'Suci', 2024)
PS D:\71200645_PrakAlpro_11>
```

Fungsi-fungsi Tuple

Tuple juga memiliki fungsi bawaan dari Python seperti *list*.

1. *len():* menghitung jumlah *item* pada *tuple*.
2. *max()*: Mencari nilai paling besar dari sebuah *tuple*.
3. *min():* Mencari nilai paling kecil dari sebuah *tuple*.
4. *tuple(seq):* Mengubah seq menjadi *tuple*.

Penggunaan List, Tuple, dan Dictionary

- Menggunakan *list* ketika membutuhkan urutan elemen yang dapat diubah, seperti daftar barang belanjaan.
- Menggunakan *tuple* ketika ingin memastikan bahwa data tidak berubah setelah dibuat, seperti koordinat geografis.

- Menggunakan *dictionary* ketika ingin menyimpan dan mengakses data dengan cepat menggunakan kunci tertentu, seperti basis data kecil.

source: https://blog.unmaha.ac.id/memahami-list-tuple-dan-dictionary-di-python

Membandingkan Tuple

Operator perbandingan (compare) dapat bekerja pada *tuple* dan model sekuensial lainnya (*list, dictionary, set*). Cara kerjanya dengan membandingkan elemen pertama dari setiap sekuensial yang ada. Jika ditemukan adanya kesamaan, akan berlanjut ke elemen berikutnya. Proses ini berlangsung terus menerus hingga ditemukan adanya perbedaan

Fungsi (*sort*) pada python bekerja dengan cara yang sama. Tahap pertama akan melakukan pengurutan berdasarkan elemen pertama. Pada kasus tertentu akan mengurutkan berdasarkan elemen kedua dan sebagainya. Fitur ini disebut dengan **DSU – (Decorate, Sort, Undercorate).**

1. *Decorate*: urutan (sekuensial) membangun daftar *tuple* dengan satu atau lebih key pengurutan sebelum elemen dari urutan.
2. *Sort*: *list tuple* menggunakan *sort*.
3. *Undercorate*: melakukan ekstraksi pada elemen yang telah diurutkan pada satu sekuensial.

Penugasan Tuple

Python dapat memiliki *tuple* di sisi kiri dari *statement* penugasan. Hal ini mengizinkan untuk menetapkan lebih dari satu variabel pada sisi sebelah kiri secara berurutan.

Sequence Unpacking

Mengekstrak isi dari tuple ke dalam variabel-variabel tunggal secara berurutan.

```
kata = ("Sampai jumpa", "saudaraku")
pertama, kedua = kata
```

```
kata = "Sampai jumpa", "saudaraku"
(pertama, kedua) = kata
```

```
kata = "Sampai jumpa", "saudaraku"
pertama = kata[0]
kedua = kata[1]
```

Slicing Tuple

```
kata = tuple(["Beruang", "Kucing", "Panda", "Kupu-kupu"])
print(kata[0:1])
print(kata[0:2])
print(kata[1:3])
print(kata[0:-1])
print(kata[-1:-3])
print(kata[-1:3])
print(kata[-3:-1])
```

OUTPUT:

```
PS D:\71200645_PrakAlpro_11> & C:/Users/HP/AppData/Local/Programs
/Python/Python311/python.exe d:/71200645_PrakAlpro_11/modul.py
('Beruang',)
('Beruang', 'Kucing')
('Kucing', 'Panda')
('Beruang', 'Kucing', 'Panda')
()
()
('Kucing', 'Panda')
PS D:\71200645_PrakAlpro_11>
```

Mengubah tuple melalui index.

```
kata = tuple(["Beruang", "Kucing", "Panda", "Kupu-kupu"])
kata = ("Penguin",) + kata[1:]
print(kata)
```

OUTPUT:

```
PS D:\71200645_PrakAlpro_11> & C:/Users/HP/AppData/Local/Program
s/Python/Python311/python.exe d:/71200645_PrakAlpro_11/modul.py
('Penguin', 'Kucing', 'Panda', 'Kupu-kupu')
PS D:\71200645_PrakAlpro_11>
```

Menggunakan daftar *list* berbentuk *tuple* berisi nama dan *list* nilai untuk menemukan murid dengan nilai rata-rata tertinggi.

```
def nilai_rata_tertinggi(murid):
    avg_tertinggi = 0
    murid_tertinggi = None

    for i in murid:
        nama, nilai = i
        avg = sum(nilai) / len(nilai)
        if avg > avg_tertinggi:
            avg_tertinggi= avg
            murid_tertinggi = nama

    return murid_tertinggi, avg_tertinggi

murid = [("Yesaya", [85, 90, 92]), ("Yeremia", [78, 85, 80]),
("Yehezkiel", [90, 92, 95])]
murid_tertinggi, avg_tertinggi = nilai_rata_tertinggi(murid)

print("Nilai rata tertinggi")
print("Nama:", murid_tertinggi)
print("Nilai rata-rata:", avg_tertinggi)
```

OUTPUT:

```
PS D:\71200645_PrakAlpro_11> & C:/Users/HP/AppData/Local/Program
s/Python/Python311/python.exe d:/71200645_PrakAlpro_11/modul.py
Nilai rata tertinggi
Nama: Yehezkiel
Nilai rata-rata: 92.33333333333333
PS D:\71200645_PrakAlpro_11>
```

MATERI 2

DICTIONARIES AND TUPLE

Dictionaries mempunyai metode yang disebut *items* untuk mengembalikan nilai *list* dari *tuple* dimana tiap *tuple*-nya merupakan *key-value* pair (pasangan kunci dan nilai).

Tuple sebagai kunci Dictionaries

Tuple merupakan hashable dan *list* tidak. Ketika kita ingin membuat *composite key* yang digunakan dalam *dictionary*, kita dapat menggunakan *tuple* sebagai *key*.

Misalnya menggunakan *composite key* jika ingin membuat direktori telepon yang memetakan dari pasangan last-name, first-name ke nomor telepon. Dengan asumsi bahwa kita telah mendefinisikan variabel last, first, dan nomor. Penulisan pernyataan penugasan dalam *dictionary* sebagai berikut: directory[last,first] = number

Expression yang ada didalam kurung kotak adalah *tuple*.

```python
kata = {"a": 10, "b": 1, "c": 22}
daftar = list()
for key, value in kata.items():
    daftar.append((value, key))
print(daftar)
```

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

Latihan 11.1 Buatlah program untuk melakukan pengecekan apakah semua anggota yang ada didalam tuple sama.

```
Contoh:
tA= (90, 90, 90, 90)

Output
True
```

Pengerjaan soal dilakukan dengan menggunakan fungsi *len()* untuk mengetahui panjang atau berapa banyak *item* yang ada pada variabel, lalu menggunakan fungsi *set()* agar setiap *item* pada variabel berbeda-beda (unik), tetapi dilakukan *assignment* satu, agar hanya ada satu elemen pada variabel.

SOURCE CODE:

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

tA = (90, 90, 90, 90)
print(elemen_sama(tA))

tB = (1, 2, 3, 4)
print(elemen_sama(tB))
```

OUTPUT:

```
PS D:\71200645_PrakAlpro_11> & C:/Users/HP/AppData/Local/Programs
/Python/Python311/python.exe "d:/71200645_PrakAlpro_11/Latihan 11
.1.py"
True
False
PS D:\71200645_PrakAlpro_11>
```

SOAL 2

Latihan 11.2 Buatlah program dengan menggunakan tuple yang dapat melalakukan proses seperti pada kasus 11.1, Gunakan data diri anda masing-masing dan lakukan perubahan supaya didapatkan output seperti contoh berikut ini :

```
Contoh:
Data:  ('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:  ('a', 't', 'a', 'h', 'a', 'r', 'i')

NAMA TERBALIK:  ('Nendya', 'Bhakti', 'Matahari')
```

Pengerjaan soal dilakukan dengan menggunakan fungsi *print()* untuk data, NIM pada data di-*index* satu, NAMA pada data di-*index* nol, ALAMAT pada data di-*index* dua. Selanjutnya, NIM dijadikan bentuk *tuple* untuk memisahkan huruf-huruf, sehingga menjadi karakter pada *tuple*. Untuk nama depan diambil data pada *index* nol dimana tersimpan nama, kemudian menggunakan fungsi *split()* untuk *index* nol, sehingga hanya nama depan saja yang diambil dan fungsi *tuple()* untuk menjadi karakter *tuple*. Untuk nama terbalik, diambil data pada *index* nol, kemudian fungsi *split()* di setiap *whitespace* dan *index* [::-1] untuk mengambil data dari akhir sampai awal. Nama terbalik dijadikan *tuple*, lalu fungsi split() agar terpisah berdasarkan *whitespace*.

SOURCE CODE:

```
def data_diri(data):
    print("Data:", tuple(data))
    print()
    print("NIM :", data[1])
    print("NAMA :", data[0])
    print("ALAMAT :", data[2])
    print()

    nim = tuple(data[1])

    nama_depan = tuple(data[0].split()[0])
    nama_terbalik = ' '.join(data[0].split()[::-1])
    tuple_nama_terbalik = tuple(nama_terbalik.split())

    print(f"NIM: {nim}")
    print(f"NAMA DEPAN: {nama_depan}")
    print(f"NAMA TERBALIK: {tuple_nama_terbalik}")

daftar = ('Edith Felicia Putri', '71200645', 'Pekanbaru, Riau')
data_diri(daftar)
```

OUTPUT:

```
PS D:\71200645_PrakAlpro_11> & C:/Users/HP/AppData/Local/Programs/Python/Python311/python.exe "d:/71200645_PrakAlpro_11/Latihan 11.2.py"
Data: ('Edith Felicia Putri', '71200645', 'Pekanbaru, Riau')

NIM : 71200645
NAMA : Edith Felicia Putri
ALAMAT : Pekanbaru, Riau

NIM: ('7', '1', '2', '0', '0', '6', '4', '5')
NAMA DEPAN: ('E', 'd', 'i', 't', 'h')
NAMA TERBALIK: ('Putri', 'Felicia', 'Edith')
PS D:\71200645_PrakAlpro_11>
```

SOAL 3

Latihan 11.3 Buatlah program untuk menghitung distribusi jam dalam satu hari dimana ada pesan yang diterima dari setiap email yang masuk. Gunakan file mbox-short.txt untuk sebagai datanya. Berikut ini adalah contoh output dari programmnya.

```
Contoh:
Enter a file name: mbox-short.txt
04 3
06 1
07 1
09 2
10 3
11 6
14 1
15 2
16 4
17 2
18 1
19 1
```

Pengerjaan soal dilakukan dengan menggunakan fungsi *open()* untuk membuka *file* 'mbox-short.txt' dalam mode membaca. Kemudian membuat *list* kosong dengan nama variabel jam_email untuk menyimpan jam dan jumlah *email* yang diterima setiap jam. *Key*-nya berupa jam dan *value*-nya berupa jumlah *email* yang diterima pada jam tersebut. Menggunakan for loop untuk membaca setiap baris (i) pada *file*, jika baris dimulai dengan "From " diambil waktu dengan *split()* pada *index* kelima, kemudian hanya mengambil bagian jam dengan fungsi *split()* lagi pada *index* nol. Lalu menambahkan *tuple* baru pada jam_email berisi jam dan jumlah *email* awal (1). Setelah menambahkan semua *tuple*, dibuat *dictionary* hitung_email untuk menghitung jumlah *email* setiap jam. Bentuk *dictionary* diubah menjadi *list-list tuple* dengan *items()*, sehingga terdapat *list* pasangan jam dan jumlah *email*. Fungsi *sort()* untuk mengurutkan hasil sehingga *ascending* berdasarkan jam (*key*).

SOURCE CODE:

```
def hitung_jam(filename):
    with open(filename, 'r') as file:
        jam_email = []
```

```
        for i in file:
            if i.startswith("From "):
                waktu = i.split()[5]
                jam = waktu.split(':')[0]
                jam_email.append((jam, 1))

    hitung_email = {}
    for jam, _ in jam_email:
        hitung_email[jam] = hitung_email.get(jam, 0) + 1

    jam_email_tuple = list(hitung_email.items())
    jam_email_tuple.sort()

    for jam, jumlah in jam_email_tuple:
        print(jam, jumlah)

filename = "mbox-short.txt"
hitung_jam(filename)
```

OUTPUT:

```
PS D:\71200645_PrakAlpro_11> & C:/Users/HP/AppData/Local/Programs/Python/Python311/python.exe "d:/71200645_PrakAlpro_11/Latihan 11.3.py"
04 3
06 1
07 1
09 2
10 3
11 6
14 1
15 2
16 4
17 2
18 1
19 1
PS D:\71200645_PrakAlpro_11>
```

Link Github:
https://github.com/EdithFelicia/71200645_Guided/tree/main/71200645_PrakAlpro_11