



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	71231052
Nama Lengkap	GABRIEL SACHIO ATMADJAJA
Minggu ke / Materi	12 / 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%)

Sumber: Modul Pratikum Algoritma dan Pemrograman

MATERI 1

Tuple bisa dibidang mirip list, bedanya yaitu tuple bersifat *immutable* atau anggota dalam tuple tidak bisa diubah. Contoh tuple sebagai berikut:

```
tuple.py > ...
  Click here to ask Blackbox to help you code faster
1  t1 = 'a', 'b', 'c', 'd', 'e'
2  t2 = ('a', 'b', 'c', 'd', 'e')
3  print(type(t1))
4  print(type(t2))
```

Gambar 1.1: Contoh tuple

```
PS D:\Sachio\Kampus\PrakAlPro\Pert12> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe
<class 'tuple'>
<class 'tuple'>
```

Gambar 1.2: output

Dalam tuple, elemen/anggotanya bisa saling dibandingkan (menggunakan operator perbandingan). Contohnya sebagai berikut:

```
15  # MEMBANDINGKAN TUPLE
16  print((0, 1, 2) < (0, 3, 4))
17  print((0, 1, 2000000) < (0, 3, 4))
```

Gambar 1.3: Perbandingan tuple

```
PS D:\Sachio\Kampus\PrakAlPro\Pert12> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe
True
True
```

Gambar 1.4: Output

Lalu dalam tuple, bisa juga mengurutkan elemennya dari panjang karakter, entah itu dari terkecil atau terbesar. Dalam kasus ini, saya akan contohkan programnya dari yang terbesar ke terkecil.

```
19  kalimat = 'But but soft what light in yonder window breaks'
20  dafkata = kalimat.split()
21  t = list()
22  for kata in dafkata:
23      t.append((len(kata), kata))
24
25  t.sort(reverse=True)
26  # t.sort(reverse=False)
27
28  urutan = list()
29  for length, kata in t:
30      urutan.append(kata)
31
32  print(urutan)
```

Gambar 1.5: Contoh cara mengurutkan elemen tuple dari terbesar ke terkecil

```
PS D:\Sachio\Kampus\PrakAlPro\Pert12> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe c
['yonder', 'window', 'breaks', 'light', 'what', 'soft', 'but', 'But', 'in']
```

Gambar 1.6: Output

Penugasan Tuple

Pada Python, terdapat fitur yang mengijinkan untuk menetapkan lebih dari satu variabel pada sisi sebelah kiri secara berurutan. Berikut contohnya:

```
35 # PENUGASAN TUPLE
36 m = [ 'have', 'fun' ]
37 x, y = m
38
39 # x = m[0]
40 # y = m[1]
41
42 print(x)      # have
43 print(y)      # fun
```

Gambar 1.7: Contoh

```
PS D:\Sachio\Kampus\PrakAlPro\Pert12> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe c
have
fun
```

Gambar 1.8: Output

Perhatikan juga di gambar 1.7, pada baris ke-39 dan baris ke-40, itu adalah cara untuk mengakses tuple yang umum (menggunakan indeks) dan menyimpannya di variabel.

Semisal, variabel di sebelah kiri kurang atau melebihi jumlah elemen yang ada di tuple, maka outputnya pasti error. Berikut contohnya:

```
45 email = 'didanendya@ti.ukdw.ac.id'
46 # username, domain = email.split('@')
47 username, domain = email.split('.')          # Error: too many values to unpack
48
49 print(username)
50 print(domain)
```

Gambar 1.9: Contoh penugasan tuple yang menghasilkan error

```
PS D:\Sachio\Kampus\PrakAlPro\Pert12> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe c
Traceback (most recent call last):
  File "d:\Sachio\Kampus\PrakAlPro\Pert12\tuple.py", line 47, in <module>
    username, domain = email.split('.')          # Error: too many values to unpack
    ^^^^^^^^^^^^^^^^^
ValueError: too many values to unpack (expected 2)
```

Gambar 1.10: Output

Dari gambar 1.9, email akan dibagi menjadi token-token berdasarkan titik (.) dan terdapat 4 anggota di list. Karena variabel yang menampung di sebelah kiri hanya 2, maka terjadi error.

Dictionaries dan Tuple

Dalam dictionary, terdapat method items() yang berfungsi untuk mengakses semua item di dictionaries (pasangan key dan value).

```
52 # DICTIONARIES DAN TUPLE
53 d = {'a':10, 'b':1, 'c':22}
54 t = list(d.items())
55 print(t)
```

Gambar 1.11: Kode

```
PS D:\Sachio\Kampus\PrakAlPro\Pert12> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe
[('a', 10), ('b', 1), ('c', 22)]
```

Gambar 1.12: Output

Multipenugasan dengan dictionaries

Cara lain dalam mengakses pasangan key dan value adalah dengan menggunakan kombinasi method items, penugasan tuple, dan perulangan for. Berikut contohnya:

```
57 # MULTIPENUGASAN DENGAN DICTIONARIES
58 d = {'a':10, 'b':1, 'c':22}
59 l = list()
60
61 for key, val in d.items() :
62     l.append( (val, key) )
63 print(l)                                # [(10, 'a'), (22, 'c'), (1, 'b')]
64
65 l.sort(reverse=True)
66 print(l)                                # [(22, 'c'), (10, 'a'), (1, 'b')]
```

Gambar 1.13: Contoh multipenugasan dengan dictionaries

```
PS D:\Sachio\Kampus\PrakAlPro\Pert12> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe
[(10, 'a'), (1, 'b'), (22, 'c')]
[(22, 'c'), (10, 'a'), (1, 'b')]
```

Gambar 1.14: Output

Kata yang sering muncul / yang paling sedikit muncul

Dalam kasus ini saya membuat program yang akan coba menampilkan kata yang paling sedikit dengan maksimal kata yang berbeda adalah 10. Berikut contohnya:

```
68 # KATA YANG SERING MUNCUL
69 import string
70 fhand = open('romeo-full.txt')
71 counts = dict()
72 for line in fhand:
73     line = line.translate(str.maketrans('', '', string.punctuation))
74     line = line.lower()
75     words = line.split()
76
77     for word in words:
78         if word not in counts:
79             counts[word] = 1
80         else:
81             counts[word] += 1
82         # urutkan dictionary by value
83         lst = list()
84         for key, val in list(counts.items()):
85             lst.append((val, key))
86
87 # urutkan dictionary by value
88 lst = list()
89 for key, val in list(counts.items()):
90     lst.append((val, key))
91
92 # lst.sort(reverse=True)
93 lst.sort(reverse=False)
94
95 for key, val in lst[:10]:
96     print(key, val)
```

Gambar 1.15: Program menampilkan kata

```
PS D:\Sachio\Kampus\PrakAlPro\Pert12> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe
1 act
1 adieu
1 adventure
1 afeard
1 against
1 air
1 alack
1 almost
1 aloud
1 already
```

Gambar 1.16: Output

Tuple sebagai kunci Dictionaries

Maksudnya adalah dalam python, ada cara agar kita bisa memasukkan/menambahkan kunci dan value dalam dictionaries, caranya adalah dengan menggunakan tuple tersebut. Tuple bersifat *hashable* dan list tidak. Perhatikan gambar di bawah ini:

```
98 # TUPLE SEBAGAI KUNCI DICTIONARIES
99 last = 'nendya'
100 first = 'dida'
101 number = '088112266'
102
103 directory = dict()
104 directory[last, first] = number
105
106 for last, first in directory:
107     print(first, last, directory[last,first])
```

Gambar 1.17: Menggunakan tuple untuk menambahkan key dan value dictionaries

```
PS D:\Sachio\Kampus\PrakAlPro\Pert12> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe
dida nendya 088112266
```

Gambar 1.18: Output

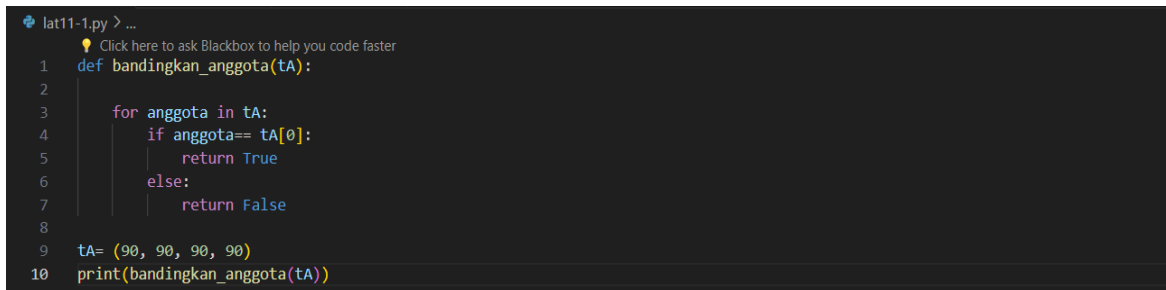
Perhatikan di gambar 1.17, di baris 104, *expression* yang ada di dalam kurung siku adalah tuple. Kode itu akan menambahkan [last, first] sebagai *composite key* dan number sebagai valuenya. Sederhananya, composite key memungkinkan Anda untuk menyimpan dan mengakses data berdasarkan dua kriteria sekaligus. Jika di cek, hasilnya akan seperti ini:

```
PS D:\Sachio\Kampus\PrakAlPro\Pert12> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe
{'nendya', 'dida': '088112266'}
```

BAGIAN 2: LATIHAN MANDIRI (60%)

Sumber: <https://github.com/gabrielsachioa/PrakAIPro11.git>

SOAL 1



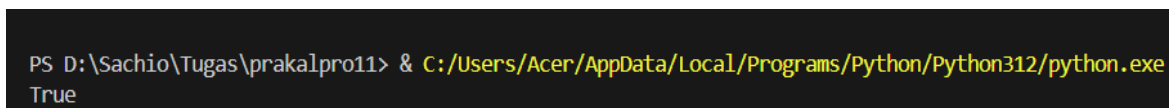
```
lat11-1.py > ...
Click here to ask Blackbox to help you code faster
1 def bandingkan_anggota(tA):
2
3     for anggota in tA:
4         if anggota == tA[0]:
5             return True
6         else:
7             return False
8
9 tA = (90, 90, 90, 90)
10 print(bandingkan_anggota(tA))
```

Gambar 2.1: Program membandingkan semua anggota di tuple

Keterangan:

Baris 1 sampai 7 itu adalah fungsi untuk membandingkan anggota di tuple. Parameternya adalah tA dari variabel atau contoh input yang ada di baris 9. Lalu di baris 3, kode perulangan for anggota in tA maksudnya adalah untuk setiap elemen/anggota di tA, cek apakah setiap anggota tersebut sama dengan anggota ke-0, jika True, maka kembalikan nilai True. Selain dari itu, kembalikan nilai False. Lalu di baris 10 untuk memanggil fungsi sekaligus menampilkan outputnya.

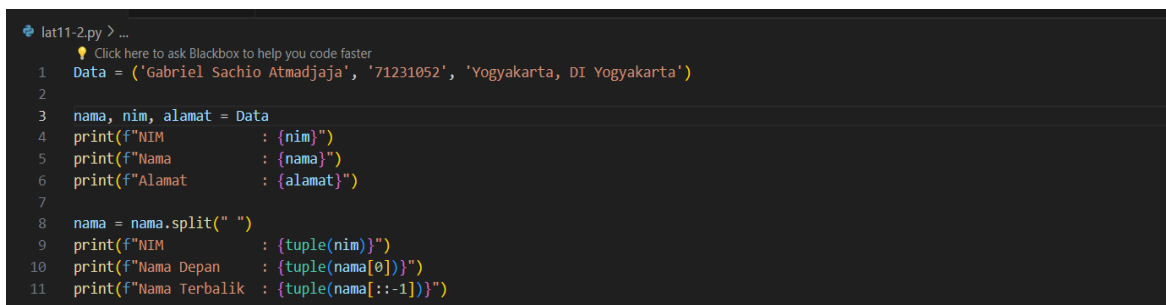
Output:



```
PS D:\Sachio\Tugas\prakalpro11> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe
True
```

Gambar 2.2: Output

SOAL 2



```
lat11-2.py > ...
Click here to ask Blackbox to help you code faster
1 Data = ('Gabriel Sachio Atmadjaja', '71231052', 'Yogyakarta, DI Yogyakarta')
2
3 nama, nim, alamat = Data
4 print(f"NIM      : {nim}")
5 print(f>Nama     : {nama}")
6 print(f"Alamat   : {alamat}")
7
8 nama = nama.split(" ")
9 print(f"NIM      : {tuple(nim)}")
10 print(f>Nama Depan : {tuple(nama[0])}")
11 print(f>Nama Terbalik : {tuple(nama[::-1])}")
```

Gambar 2.3:

Keterangan:

Baris pertama adalah tuple yang berisi data seperti nama lengkap, nim, serta alamat. Lalu baris ke-3 itu penugasan tuple, karena ada 3 anggota dalam tuple, maka perlu membuat 3 variabel di sebelah kiri yang masing-masing akan menampung anggota tuple. Lalu baris 4 sampai 6 itu menampilkan output berupa nim, nama, dan alamat. Lalu baris ke-8 akan membagi string menjadi token-token berdasarkan spasi, ini dibuat dengan tujuan digunakan untuk menampilkan nama depan dan nama terbalik nantinya. Lalu baris 9 akan menampilkan nim sebagai tuple, karena itu perlu fungsi tuple() yang otomatis membuat setiap elemennya menjadi anggota dalam tuple. Lalu baris 10 akan menampilkan nama depan, ini juga membuat tuple, tetapi anggotanya itu dari tiap elemen di nama[0] ("Gabriel"). Lalu baris 11 menampilkan nama secara terbalik, karena pada baris 8 sudah di split, maka bisa dengan mudah membalikkan anggota di tuple nama dengan menambahkan[::-1] setelah variabel nama.

Output:

```
lat11-2.py > ...
  Click here to ask Blackbox to help you code faster
1 Data = ('Gabriel Sachio Atmadjaja', '71231052', 'Yogyakarta, DI Yogyakarta')
2
3 nama, nim, alamat = Data
4 print(f"NIM      : {nim}")
5 print(f"NAMA     : {nama}")
6 print(f"ALAMAT   : {alamat}\n")
7
8 nama = nama.split(" ")
9 print(f"NIM      : {tuple(nim)}\n")
10 print(f"NAMA DEPAN : {tuple(nama[0])}\n")
11 print(f"NAMA TERBALIK : {tuple(nama[::-1])}\n")
```

Gambar 2.4: Output

SOAL 3

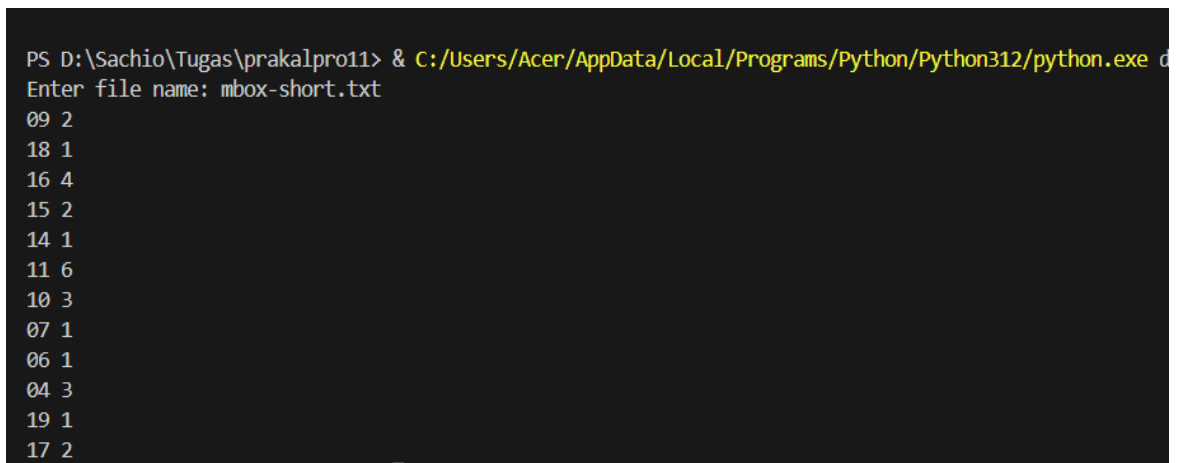
```
lat11-3.py > ...
  Click here to ask Blackbox to help you code faster
1 def distribusi_jam():
2
3     n = input("Enter file name: ")
4     d = {}
5
6     with open(n, "r") as file:
7         file = file.readlines()
8         # print(file)
9         for kalimat in file:
10             if "From" in kalimat and not "From:" in kalimat:
11                 kalimat = kalimat.split()
12                 jam = kalimat[5].split(":")[0]
13                 # print(jam)
14
15                 if jam not in d:
16                     d[jam] = 1
17                 else:
18                     d[jam] += 1
19
20             for key, value in d.items():
21                 print(key, value)
22
23
24
25
26 distribusi_jam()
```

Gambar 2.5: Program menghitung keluaran distribusi jam

Keterangan:

Baris 1 sampai dengan 22 adalah fungsi distribusi jam. Baris 3 adalah variabel `n` yang meminta inputan dari user berupa nama file (string). Lalu baris 4 adalah variabel `d` yang berfungsi untuk menampung distribusi jam dan jumlah yang muncul sebagai dictionaries. Baris 6 membuka file dalam mode read dan menampungnya di variabel `file`. Baris 7 untuk membaca seluruh baris dalam file. Baris 8 adalah komentar, untuk melihat baris yang ditampilkan. Lalu baris 9 adalah perulangan for kalimat in file, maksudnya untuk setiap kalimat di file. Jika di `print(kalimat)` nanti hasilnya sama seperti `print(file)` di baris 8. Lalu baris 10 adalah kondisi If "From" in kalimat and not "From:" in kalimat → maksudnya itu jika ada kalimat From dan bukan kalimat From: di variabel kalimat, maka jalankan kode di dalamnya. Alasan menggunakan "From" dan bukan "From:" itu karena teks yang menampilkan waktu itu hanya ada di kalimat yang menggunakan "From" dan bukan "From:". Lalu baris 11, method split itu akan membagi kalimat yang mengikuti "From" menjadi token-token berdasarkan spasi. Baris 12, variabel `jam` berfungsi untuk menampung angka jam-nya, itu dilakukan dengan `kalimat[5].split(":")[0]` → maksudnya itu dari elemen kalimat di indeks ke-5, bagi lagi menjadi token berdasarkan titik dua, lalu ambil indeks ke-0 karena itu yang menunjukkan angka jam-nya. Lalu baris 13 itu komentar, jika ingin mengecek output variabel `jam`. Lalu baris 15, kondisi if `jam not in d` → artinya kalau `jam` tidak ada di dalam `d` (dictionary), maka akan menjalankan baris 16 yaitu memasukkan `jam` tersebut ke dalam dictionary dan set counternya jadi 1 (`d[jam] = 1`). Lalu baris 17, kondisi else akan menjalankan kode di baris 18 → maksudnya itu jika `jam` sudah ada di dalam `d` atau dictionaries, maka counter untuk `jam` tersebut akan ditambah satu. Baris 21, maksudnya itu untuk setiap key dan value yang ada di `d.items` (akan mengakses pasangan kunci dan nilai di variabel `d`), jalankan kode di baris 22 untuk menampilkan outputnya (key itu menunjukkan jam-nya dan value itu menunjukkan counternya). Lalu baris 26 itu memanggil fungsi `distribusi_jam()`

Output:



```
PS D:\Sachio\Tugas\prakalpro11> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d
Enter file name: mbox-short.txt
09 2
18 1
16 4
15 2
14 1
11 6
10 3
07 1
06 1
04 3
19 1
17 2
```

Gambar 2.6: Output