



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	<71231051>
Nama Lengkap	<Amida A Ronsumbre>
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%)

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

- 11.3.1 Tuple Immutable

Karakteristik Tuple:

Immutable: Nilai tidak bisa diubah setelah dibuat.

Hashable: Dapat digunakan sebagai kunci dalam dictionary.

Comparable: Dapat dibandingkan dengan tuple lainnya.

Penulisan Tuple:

Tanpa kurung: `t = 'a', 'b', 'c'`

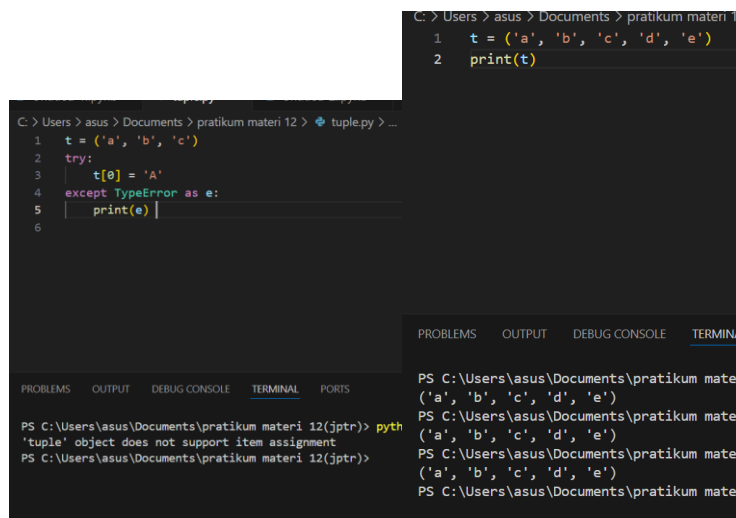
Dengan kurung: `t = ('a', 'b', 'c')`

Satu elemen (pakai koma): `t1 = ('a',)`. Sebagai Kunci Dictionary: Karena sifat hashable-nya, tuple dapat digunakan sebagai kunci dalam dictionary.

Membandingkan Nilai: Tuple dapat dibandingkan menggunakan operator perbandingan.

Pengelompokan Data: Tuple sering digunakan untuk mengelompokkan data yang tidak perlu diubah.

Dengan memahami karakteristik dan penggunaan tuple, Anda bisa memanfaatkan struktur data ini secara efektif dalam program Python Anda.



```
C:\> Users\asus\Documents\pratikum materi 12
1 t = ('a', 'b', 'c', 'd', 'e')
2 print(t)

C:\> Users\asus\Documents\pratikum materi 12 > tuple.py > ...
1 t = ('a', 'b', 'c')
2 try:
3     t[0] = 'A'
4 except TypeError as e:
5     print(e)
6

PS C:\Users\asus\Documents\pratikum materi 12(jptr)> pyth
'tuple' object does not support item assignment
PS C:\Users\asus\Documents\pratikum materi 12(jptr)>
```

Alasan: Fungsi tuple() dapat mengkonversi urutan (seperti string, list, atau tuple lain) menjadi tuple. Dalam contoh ini, string 'dutawacana' dipecah menjadi karakter-karakter individu dan dimasukkan ke dalam tuple. Alasan: Karena tuple bersifat immutable, Anda tidak dapat mengubah elemen-elemen di dalamnya secara langsung. Sebagai gantinya, Anda dapat membuat tuple baru dengan menggabungkan bagian-bagian dari tuple yang ada dengan nilai yang baru. Alasan: Tuples bersifat immutable, artinya Anda tidak dapat mengubah nilai elemen setelah tuple dibuat. Mencoba melakukannya akan menghasilkan TypeError. Contoh ini menunjukkan bahwa Python tidak mengizinkan perubahan elemen tuple setelah tuple didefinisikan. Alasan: Tuples bersifat immutable, artinya Anda tidak dapat mengubah nilai elemen setelah tuple dibuat. Mencoba melakukannya akan menghasilkan TypeError. Contoh ini menunjukkan bahwa Python tidak mengizinkan perubahan elemen tuple setelah tuple didefinisikan.

- 11.3.3 Penugasan Tuple

Dari contoh diatas, dapat dilihat bahwa kedua statemnt merupakan tuple. Bagian kiri merupakan tuple dari variable dan bagian kanan merupakan tuple dari expresions.

ValueError: too many values to unpack

Pada sisi sebelah kanan biasanya terdapat data sekuensial string, list atau tuple. "Nilai yang dikembalikan dari split terdiri dari dua elemen yang dipisahkan tanda "@", elemen pertama berisi username dan elemen selanjutnya berisi domain.

Penugasan menggunakan tuple di Python memungkinkan menetapkan beberapa variabel sekaligus dalam satu pernyataan, membuat kode lebih ringkas dan mudah dibaca. Berikut ringkasannya:

1. Penugasan Banyak Variabel:

- Menetapkan beberapa variabel dari urutan seperti list atau tuple dalam satu baris. Contoh: ``x, y = ['have', 'fun']``.

2. Penggunaan Tanda Kurung (Opsional):

- Bisa menggunakan tanda kurung atau tidak: ``(x, y) = m`` atau ``x, y = m``.

3. Menukar Nilai Variabel:

- Mudah menukar nilai variabel: ``a, b = b, a``.

4. Kesamaan Jumlah Elemen:

- Jumlah variabel di sisi kiri harus sama dengan elemen di sisi kanan, jika tidak akan muncul error.

5. Contoh Membagi Alamat Email:

- Membagi email menjadi username dan domain: ``username, domain = email.split('@')``.

Penugasan tuple membuat kode lebih efisien, jelas, dan mudah dikelola. Seperti contoh dibawah di ini:

```
C:\Users\asus> Documents > pratikum materi 12 > tuple.py > ...
1 email = 'didanendya@ti.ukdw.ac.id'
2 username, domain = email.split('@')
3 print(username) |
4 print(domain)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

PS C:\Users\asus\Documents\pratikum materi 12(jptr)> python -u
didanendya
ti.ukdw.ac.id
PS C:\Users\asus\Documents\pratikum materi 12(jptr)>
```

- 11.3.4 Dictionaries and Tuple

Metode `items` pada dictionary di Python mengembalikan daftar pasangan kunci-nilai sebagai tuple. Misalnya, jika Anda memiliki dictionary `d = {'a': 10, 'b': 1, 'c': 22}`, menggunakan `d.items()` akan memberikan daftar tuple seperti `[('a', 10), ('b', 1), ('c', 22)]`.

Daftar tuple ini bisa diurutkan. Secara default, pengurutan akan dilakukan berdasarkan kunci secara alfabetis. Jadi, setelah menggunakan `t.sort()` pada daftar tuple, hasilnya akan diurutkan seperti `[('a', 10), ('b', 1), ('c', 22)]`.

Intinya, `items` membantu mengubah dictionary menjadi daftar tuple yang bisa diurutkan sesuai kebutuhan.

- 11.3.5 Multipenugasan dengan dictionaries

Kita bisa menggunakan `for key, val in d.items():` untuk mengiterasi setiap pasangan kunci-nilai dalam dictionary.

Pada setiap iterasi, `key` dan `val` akan mendapatkan kunci dan nilai dari dictionary.

Contohnya, jika kita punya dictionary `d = {'a': 10, 'b': 1, 'c': 22}`, kita bisa menulis:

Alasan penggunaan ini adalah untuk mempermudah pengaksesan dan penampilan setiap elemen dalam dictionary.

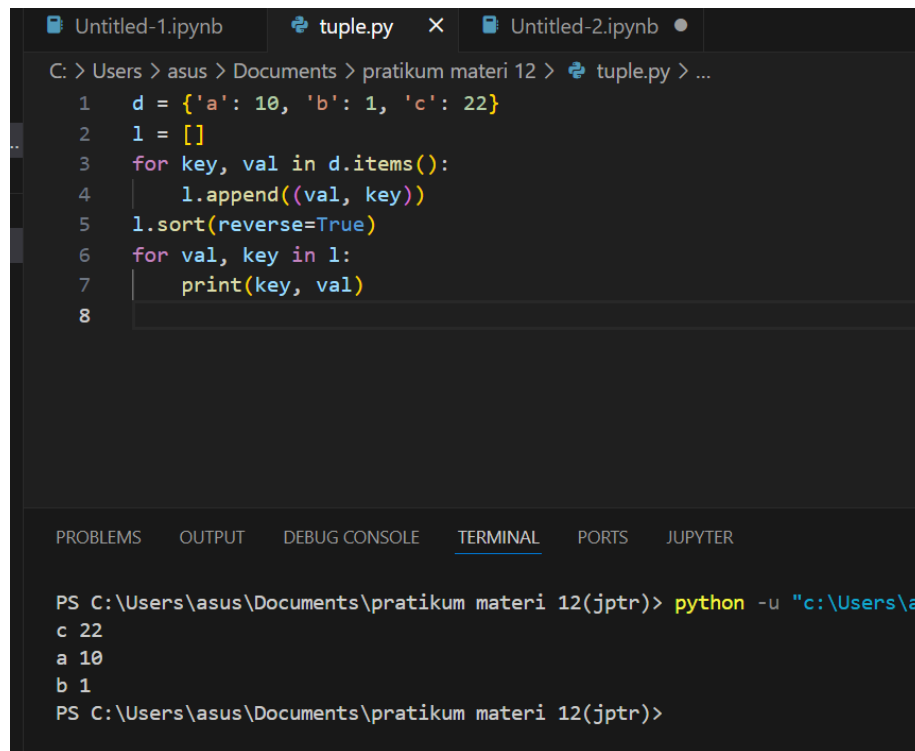
Mengurutkan Isi Dictionary Berdasarkan Nilai:

Pertama, kita buat daftar tuple di mana setiap tuple adalah pasangan (nilai, kunci) dari dictionary.

Lalu, kita urutkan daftar ini berdasarkan nilai.

Alasan untuk melakukan ini adalah untuk mendapatkan tampilan isi dictionary yang diurutkan berdasarkan nilai, yang bisa sangat berguna ketika nilai lebih penting daripada kunci dalam konteks tertentu.

Dengan cara ini, kita bisa mengurutkan dan mencetak isi dictionary berdasarkan nilai dengan mudah dan efisien, memanfaatkan fitur-fitur yang disediakan oleh Python untuk penanganan data yang lebih baik.



```
Untitled-1.ipynb  tuple.py  X  Untitled-2.ipynb  ●
C: > Users > asus > Documents > pratikum materi 12 > tuple.py > ...
1  d = {'a': 10, 'b': 1, 'c': 22}
2  l = []
3  for key, val in d.items():
4      l.append((val, key))
5  l.sort(reverse=True)
6  for val, key in l:
7      print(key, val)
8

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  JUPYTER

PS C:\Users\asus\Documents\pratikum materi 12(jptr)> python -u "c:\Users\as
c 22
a 10
b 1
PS C:\Users\asus\Documents\pratikum materi 12(jptr)>
```

- 11.3.6 Kata yang sering muncul

Membaca dan Menghitung Frekuensi Kata:

- Pertama, kita membuka file teks "Romeo and Juliet Act 2, Scene 2" dan membaca isinya.
- Setiap baris teks dihilangkan tanda baca dan diubah menjadi huruf kecil untuk konsistensi.
- Kata-kata dipisahkan dan frekuensi kemunculannya dihitung menggunakan dictionary. Konversi ke List Tuple dan Pengurutan:
- Setelah menghitung frekuensi kata, dictionary diubah menjadi list tuple.
- Setiap tuple berisi jumlah kemunculan kata dan kata itu sendiri.
- List tuple diurutkan berdasarkan frekuensi kata, sehingga kata yang paling sering muncul berada di bagian depan list. Menampilkan 10 Kata yang Paling Sering Muncul: Dari list tuple yang sudah diurutkan, kita mencetak 10 kata yang paling sering muncul. Karena list sudah diurutkan dari frekuensi tertinggi ke terendah, 10 kata pertama dalam list adalah yang paling sering muncul.

Alasan Python Dipilih untuk Analisis Teks:-Python menyediakan alat bawaan yang kuat untuk manipulasi string dan struktur data, yang membuatnya ideal untuk analisis teks. Kode yang relatif singkat dan

mudah dipahami memungkinkan analisis yang kompleks dilakukan dengan cepat dan efisien. Kemampuan Python untuk memanipulasi file dan melakukan operasi pada teks dengan mudah menjadikannya pilihan yang baik untuk tugas seperti ini. Dengan demikian, menggunakan Python, kita dapat dengan cepat dan mudah menganalisis teks dan mendapatkan wawasan berharga dari data teks.

```
Untitled-1.ipynb  tuple.py  X  Untitled-2.ipynb
C: > Users > asus > Documents > pratikum materi 12 > tuple.py > ...
1  import string
2
3  fhand = open('romeo-full1.txt')
4  counts = dict()
5  for line in fhand:
6      line = line.translate(str.maketrans('', '', string.punctuation))
7      line = line.lower()
8      words = line.split()
9      for word in words:
10         if word not in counts:
11             counts[word] = 1
12         else:
13             counts[word] += 1
14
```

11.3.7 Tuple sebagai kunci dictionaries

Dalam Python, tuple dianggap sebagai objek yang tidak bisa diubah (immutable), sehingga dapat digunakan sebagai kunci dalam sebuah dictionary. Sebaliknya, list dianggap sebagai objek yang dapat diubah (mutable), dan tidak dapat digunakan sebagai kunci dalam dictionary.

Misalnya, jika kita ingin membuat sebuah direktori telepon yang memetakan pasangan last-name, first-name ke nomor telepon, kita dapat menggunakan tuple sebagai kunci. Berikut adalah cara penggunaannya:

```
Untitled-1.ipynb  tuple.py  X  Untitled-2.ipynb
C: > Users > asus > Documents > pratikum materi 12 > tuple.py > ...
1  last = 'nendya'
2  first = 'dida'
3  number = '088112266'
4
5  directory = dict()
6  directory[last, first] = number
7
8  for last, first in directory:
9      print(first, last, directory[last, first])
10
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

```
PS C:\Users\asus\Documents\pratikum materi 12(jptr)> python -u "c:\Users\asus\Documents\pratikum materi 12\tuple.py"
dida nendya 088112266
PS C:\Users\asus\Documents\pratikum materi 12(jptr)>
```

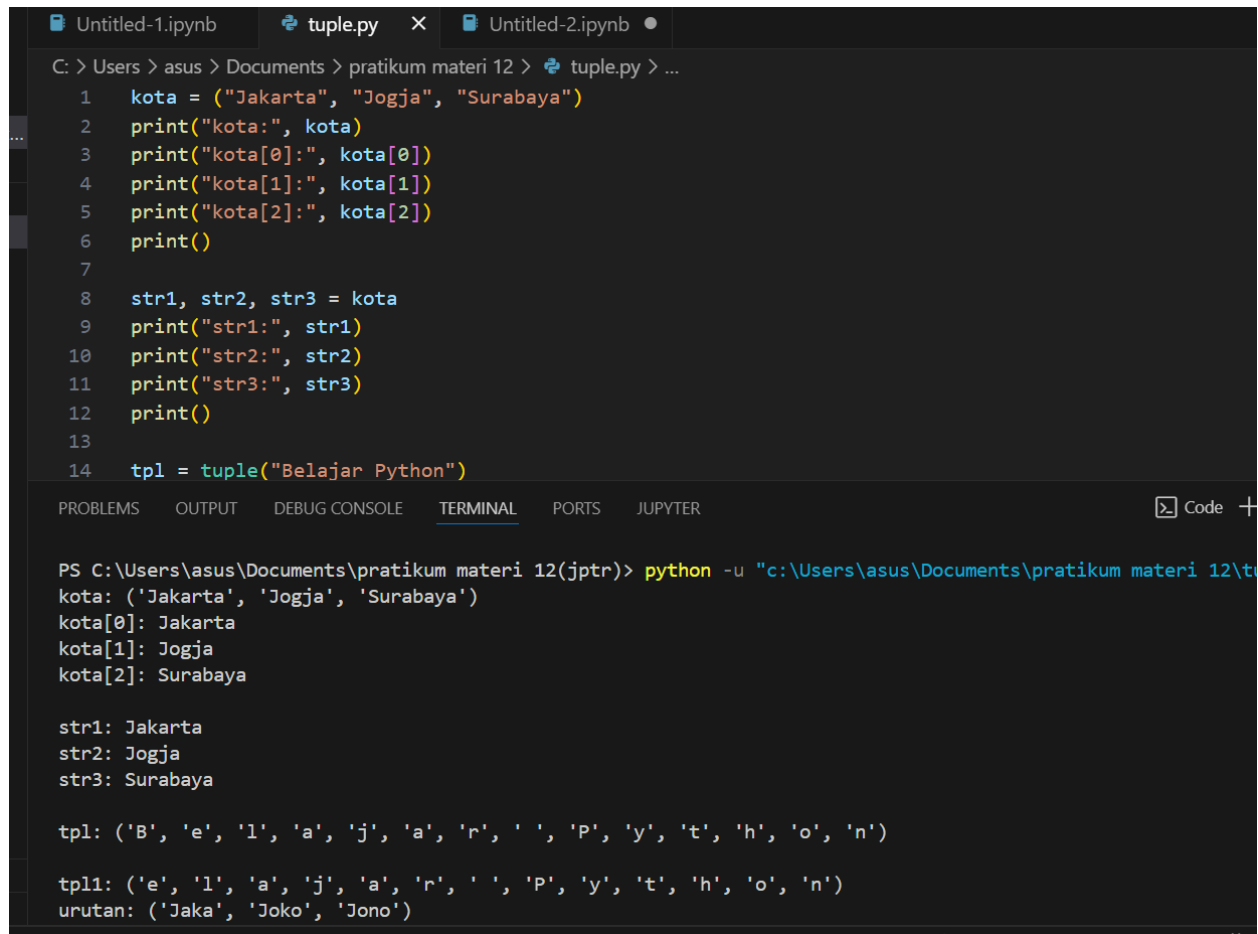
Dalam contoh tersebut, kita mendefinisikan variabel last, first, dan number. Kemudian, kita membuat sebuah dictionary directory, dimana kunci-kuncinya adalah pasangan tuple (last, first) dan nilainya adalah nomor telepon.

Pada looping for, kita melakukan iterasi pada kunci-kunci dalam dictionary directory, yang merupakan tuple (last, first). Setiap elemen dari tuple tersebut, yaitu last dan first, ditetapkan ke variabel last dan first. Kemudian, kita mencetak nama dan nomor telepon yang sesuai dengan kunci tersebut.

Dengan menggunakan tuple sebagai kunci dalam dictionary, kita dapat membuat struktur data yang lebih kompleks dan fleksibel, seperti dalam contoh direktori telepon di atas.

- 11.4 Kegiatan Praktikum

- Kasus 11.1



```
Untitled-1.ipynb  tuple.py  X  Untitled-2.ipynb
C: > Users > asus > Documents > pratikum materi 12 > tuple.py > ...
1  kota = ("Jakarta", "Jogja", "Surabaya")
2  print("kota:", kota)
3  print("kota[0]:", kota[0])
4  print("kota[1]:", kota[1])
5  print("kota[2]:", kota[2])
6  print()
7
8  str1, str2, str3 = kota
9  print("str1:", str1)
10 print("str2:", str2)
11 print("str3:", str3)
12 print()
13
14 tpl = tuple("Belajar Python")

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  JUPYTER
Code +

PS C:\Users\asus\Documents\pratikum materi 12(jptr)> python -u "c:\Users\asus\Documents\pratikum materi 12\tuple.py"
kota: ('Jakarta', 'Jogja', 'Surabaya')
kota[0]: Jakarta
kota[1]: Jogja
kota[2]: Surabaya

str1: Jakarta
str2: Jogja
str3: Surabaya

tpl: ('B', 'e', 'l', 'a', 'j', 'a', 'r', ' ', 'P', 'y', 't', 'h', 'o', 'n')

tpl1: ('e', 'l', 'a', 'j', 'a', 'r', ' ', 'P', 'y', 't', 'h', 'o', 'n')
urutan: ('Jaka', 'Joko', 'Jono')
```

berikut adalah ringkasan dari pernyataan-pernyataan dalam program tersebut:

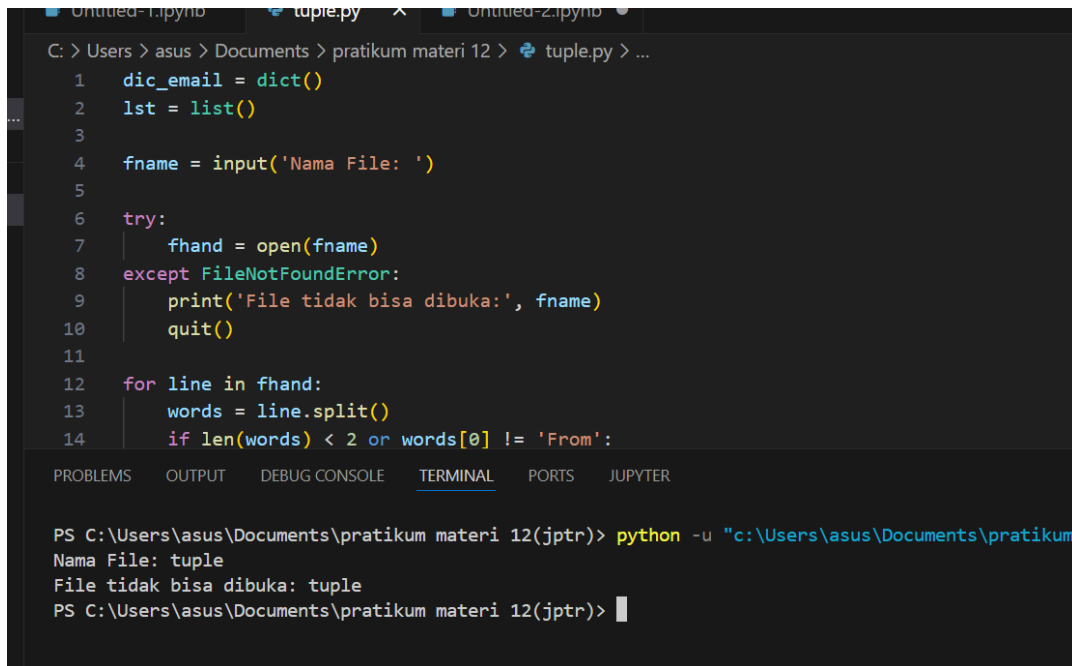
Membuat dan mencetak tuple:

- Menunjukkan cara membuat dan mengakses elemen-elemen tuple.Membagi tuple kedalam string:
- Memecah tuple menjadi beberapa variabel string terpisah.Membuat tuple dari sebuah kata:

- Membuat tuple dari sebuah string dengan mengonversi setiap karakter menjadi elemen tuple. Membuat tuple tanpa huruf pertama dari sebuah string:
- Membuat tuple dari sebuah string dengan menghilangkan karakter pertama. Mencetak tuple secara terbalik:
- Mencetak isi sebuah tuple secara terbalik menggunakan slicing.

Setiap pernyataan memiliki tujuan khusus untuk menunjukkan konsep dasar dalam penggunaan tuple dan operasi-operasi yang dapat dilakukan pada tuple dalam Python.

➤ Kasus 11.2



```

C: > Users > asus > Documents > pratikum materi 12 > tuple.py > ...
1  dic_email = dict()
2  lst = list()
3
4  fname = input('Nama File: ')
5
6  try:
7      fhand = open(fname)
8  except FileNotFoundError:
9      print('File tidak bisa dibuka:', fname)
10     quit()
11
12  for line in fhand:
13      words = line.split()
14      if len(words) < 2 or words[0] != 'From':

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

```

PS C:\Users\asus\Documents\pratikum materi 12(jptr)> python -u "c:\Users\asus\Documents\pratikum
Nama File: tuple
File tidak bisa dibuka: tuple
PS C:\Users\asus\Documents\pratikum materi 12(jptr)>

```

Pada program ini, langkah-langkahnya adalah: Membaca nama file dari pengguna. Membuka file dan mengiterasi setiap barisnya. Memisahkan kata-kata dari setiap baris. Jika baris tersebut merupakan baris "From", maka email pengirim akan ditambahkan ke dalam dictionary dic_email dengan nilai awal 1 atau diincrement jika sudah ada. Setelah selesai membaca file, kita membuat sebuah list dari tuple (jumlah_commit, email) untuk setiap email. List tersebut diurutkan berdasarkan jumlah commit dari yang terbanyak.

Terakhir, kita mencetak email dan jumlah commit terbanyak (hanya satu email yang paling banyak melakukan commit). Program ini akan menghasilkan output berupa email dan jumlah commit terbanyak dari file yang diberikan.

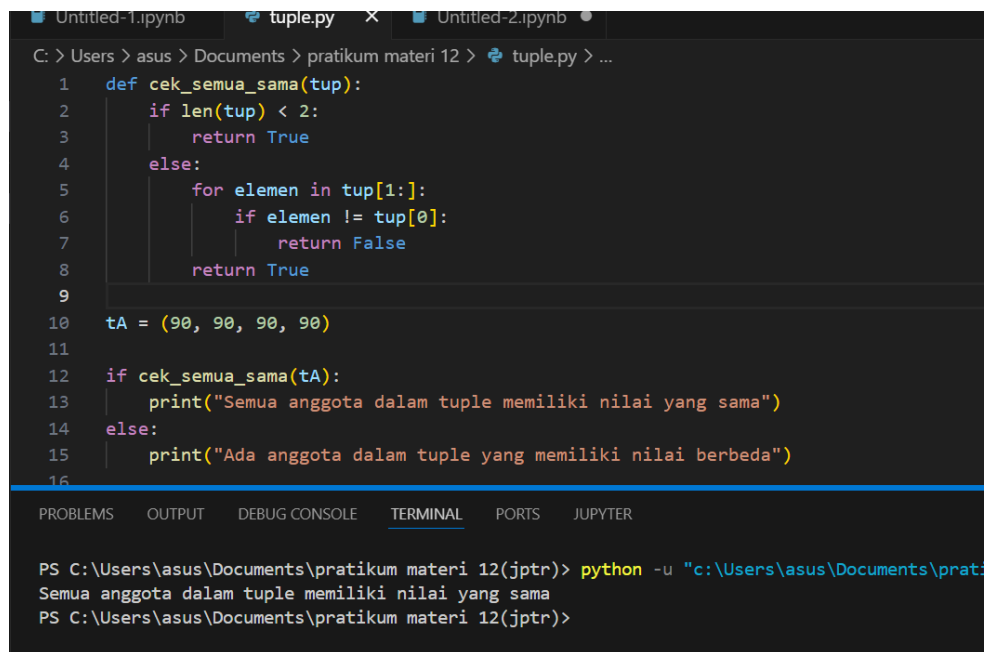
MATERI 2

Penjelasan materi 2, dst... sesuai format ini.

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



```
Untitled-1.ipynb  tuple.py  X  Untitled-2.ipynb
C: > Users > asus > Documents > pratikum materi 12 > tuple.py > ...
1  def cek_semua_sama(tup):
2      if len(tup) < 2:
3          return True
4      else:
5          for elemen in tup[1:]:
6              if elemen != tup[0]:
7                  return False
8          return True
9
10 tA = (90, 90, 90, 90)
11
12 if cek_semua_sama(tA):
13     print("Semua anggota dalam tuple memiliki nilai yang sama")
14 else:
15     print("Ada anggota dalam tuple yang memiliki nilai berbeda")
16

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  JUPYTER

PS C:\Users\asus\Documents\pratikum materi 12(jptr)> python -u "c:\Users\asus\Documents\pratikum materi 12(jptr)\tuple.py"
Semua anggota dalam tuple memiliki nilai yang sama
PS C:\Users\asus\Documents\pratikum materi 12(jptr)>
```

Tentu, berikut adalah alasan mengapa kode di atas dapat melakukan pengecekan apakah semua anggota dalam tuple memiliki nilai yang sama:

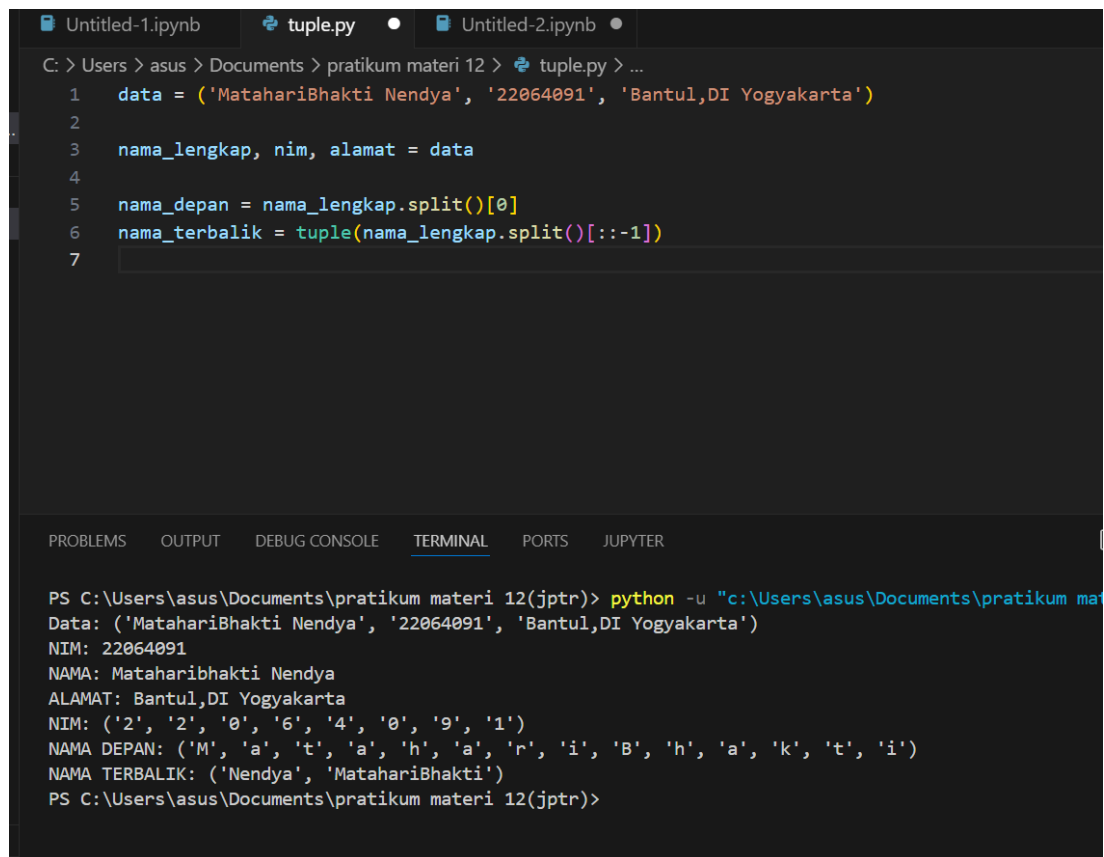
Fungsi `cek_semua_sama(tup)`:

- Fungsi ini menerima sebuah tuple sebagai argumen.
- Pada awalnya, fungsi ini memeriksa apakah panjang tuple kurang dari 2. Jika demikian, artinya tuple tersebut pasti memiliki semua anggotanya sama (baik 1 atau 0 anggota), sehingga akan mengembalikan `True`.
- Jika panjang tuple lebih dari 1, fungsi akan membandingkan setiap elemen dalam tuple dengan elemen pertama.
- Jika ditemukan elemen yang berbeda dengan elemen pertama, fungsi akan langsung mengembalikan `False`, menunjukkan bahwa tidak semua anggota memiliki nilai yang sama.
- Jika semua elemen sama dengan elemen pertama, fungsi akan mengembalikan `True`, menunjukkan bahwa semua anggota memiliki nilai yang sama.

Penggunaan contoh tuple dan pemanggilan fungsi:

- Pada contoh di bawah fungsi, kita menggunakan tuple `tA = (90, 90, 90, 90)` yang memiliki semua anggotanya sama. Kemudian, kita memanggil fungsi `cek_semua_sama()` dengan argumen tuple tersebut. Jika fungsi mengembalikan `True`, maka kita mencetak bahwa semua anggota dalam tuple memiliki nilai yang sama. Jika fungsi mengembalikan `False`, maka kita mencetak bahwa ada anggota dalam tuple yang memiliki nilai berbeda. Dengan demikian, kode tersebut secara efisien dapat mengecek apakah semua anggota dalam tuple memiliki nilai yang sama atau tidak.

Soal 2



```
Untitled-1.ipynb  tuple.py  Untitled-2.ipynb
C: > Users > asus > Documents > pratikum materi 12 > tuple.py > ...
1  data = ('MatahariBhakti Nendya', '22064091', 'Bantul,DI Yogyakarta')
2
3  nama_lengkap, nim, alamat = data
4
5  nama_depan = nama_lengkap.split()[0]
6  nama_terbalik = tuple(nama_lengkap.split()[::-1])
7

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  JUPYTER

PS C:\Users\asus\Documents\pratikum materi 12(jptr)> python -u "c:\Users\asus\Documents\pratikum materi 12\tuple.py"
Data: ('MatahariBhakti Nendya', '22064091', 'Bantul,DI Yogyakarta')
NIM: 22064091
NAMA: Mataharibhakti Nendya
ALAMAT: Bantul,DI Yogyakarta
NIM: ('2', '2', '0', '6', '4', '0', '9', '1')
NAMA DEPAN: ('M', 'a', 't', 'a', 'h', 'a', 'r', 'i', 'B', 'h', 'a', 'k', 't', 'i')
NAMA TERBALIK: ('Nendya', 'MatahariBhakti')
PS C:\Users\asus\Documents\pratikum materi 12(jptr)>
```

Data awal disimpan dalam sebuah tuple data.

Data tersebut dipisahkan menjadi tiga variabel terpisah: nama_lengkap, nim, dan alamat.

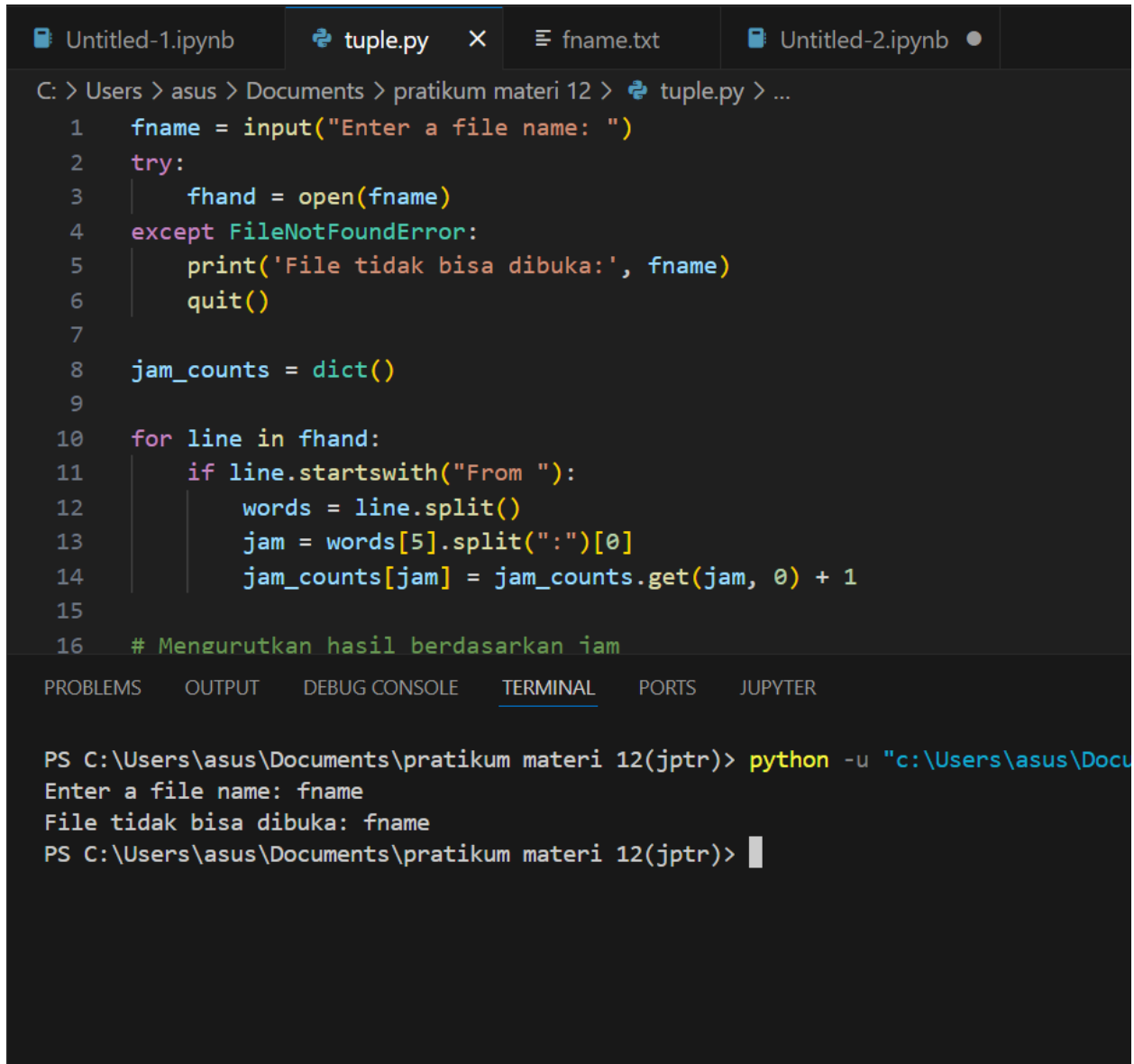
Nama lengkap kemudian dipisahkan menjadi nama depan dan nama terbalik.

Menggunakan slicing untuk mendapatkan nama depan dan nama terbalik.

Output dicetak sesuai dengan format yang diminta.

Output dari program ini akan sesuai dengan contoh yang diberikan.

Soal 3



The screenshot shows a Jupyter Notebook with four tabs: 'Untitled-1.ipynb', 'tuple.py', 'fname.txt', and 'Untitled-2.ipynb'. The 'tuple.py' tab is active, displaying a Python script. The script prompts the user for a file name, attempts to open it, and handles a 'FileNotFoundError' by printing a message and quitting. It then reads the file line by line, counts the occurrences of each hour mentioned in the 'From' field of email headers, and sorts the results by hour.

```
C: > Users > asus > Documents > pratikum materi 12 > tuple.py > ...
1  fname = input("Enter a file name: ")
2  try:
3      fhand = open(fname)
4  except FileNotFoundError:
5      print('File tidak bisa dibuka:', fname)
6      quit()
7
8  jam_counts = dict()
9
10 for line in fhand:
11     if line.startswith("From "):
12         words = line.split()
13         jam = words[5].split(":")[0]
14         jam_counts[jam] = jam_counts.get(jam, 0) + 1
15
16 # Mengurutkan hasil berdasarkan jam
```

Below the code editor, the 'TERMINAL' tab is selected, showing the execution of the script. The user enters 'fname' at the prompt, and the program outputs 'File tidak bisa dibuka: fname' before returning to the command prompt.

```
PS C:\Users\asus\Documents\pratikum materi 12(jptr)> python -u "c:\Users\asus\Docu
Enter a file name: fname
File tidak bisa dibuka: fname
PS C:\Users\asus\Documents\pratikum materi 12(jptr)> 
```

Program ini akan membaca file yang dimasukkan oleh pengguna, kemudian akan menghitung distribusi jam dalam satu hari dimana ada pesan yang diterima dari setiap email yang masuk. Hasilnya akan dicetak dalam format yang sesuai dengan contoh yang diberikan. Dengan penanganan kesalahan yang diperbarui ini, program akan memberikan pesan kesalahan yang lebih spesifik tergantung pada jenis kesalahan yang ditemui (file tidak ditemukan, izin akses ditolak, atau kesalahan lain yang tidak terduga). Ini akan membantu pengguna memahami mengapa file tidak dapat dibuka.

