



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	71230971
Nama Lengkap	James Marvin Santoso
Minggu ke / Materi	10 / Tipe Data Dictionary

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

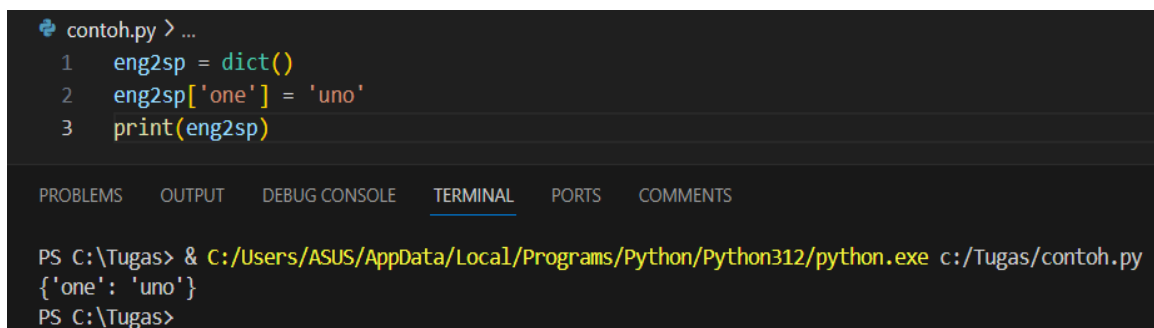
MATERI

Tipe data dictionary dalam Python adalah struktur data yang memungkinkan Anda untuk menyimpan kumpulan data dalam bentuk pasangan kunci-nilai yang bersifat tidak berurutan. Ini berarti bahwa setiap elemen dalam dictionary terdiri dari dua bagian: kunci yang unik dan nilai yang terkait dengan kunci tersebut. Dictionary digunakan untuk memetakan kunci ke nilai-nilai mereka yang sesuai, mirip dengan konsep kamus di kehidupan nyata di mana kata-kata (kunci) memiliki definisi atau makna (nilai).

Salah satu fitur utama dari dictionary dalam Python adalah kemampuannya untuk menyimpan dan mengakses nilai-nilai dengan cepat berdasarkan kunci. Karena kunci harus unik, Anda dapat dengan mudah mencari nilai tertentu dalam dictionary dengan menggunakan kunci sebagai referensi, tanpa perlu melakukan pencarian melalui seluruh struktur data.

Berikut adalah contoh penggunaan dictionary:

- Fungsi `dict` digunakan untuk membuat dictionary baru yang kosong. Karena `dict` merupakan fungsi bawaan (built-in function) dari Python, sebaiknya menghindari menggunakan `dict` sebagai nama variabel.

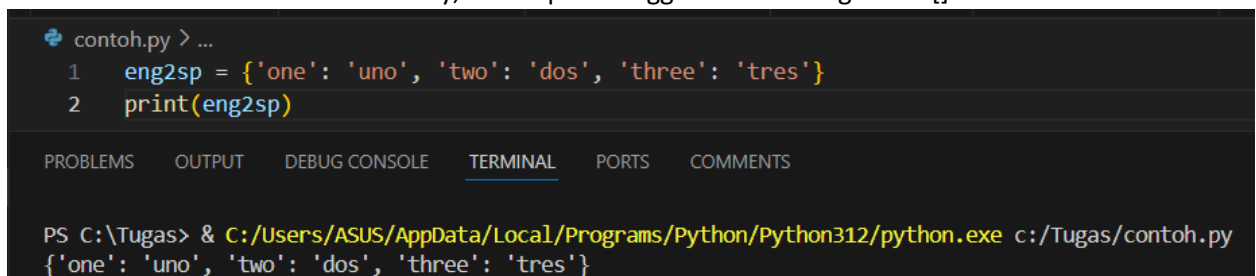


```
contoh.py > ...
1  eng2sp = dict()
2  eng2sp['one'] = 'uno'
3  print(eng2sp)

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  COMMENTS

PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/contoh.py
{'one': 'uno'}
PS C:\Tugas>
```

- Tanda kurung kurawal `{}` digunakan untuk merepresentasikan dictionary kosong. Untuk menambahkan item dalam dictionary, kita dapat menggunakan kurung kotak `[]`.



```
contoh.py > ...
1  eng2sp = {'one': 'uno', 'two': 'dos', 'three': 'tres'}
2  print(eng2sp)

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  COMMENTS

PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/contoh.py
{'one': 'uno', 'two': 'dos', 'three': 'tres'}
```

- Pengurutan pasangan kunci-nilai tidaklah sama. Secara umum, urutan item dalam dictionary tidak dapat diprediksi. Hal ini tidak menjadi masalah karena elemen dalam dictionary tidak diberikan indeks dengan indeks integer, melainkan menggunakan kunci untuk mencari nilai yang sesuai.

```
print(eng2sp['two']) # Output: 'dos'
```

- Jika menggunakan kunci yang tidak ada dalam dictionary, akan muncul pengecualian KeyError.

```
print(eng2sp['four']) # KeyError: 'four'
```

- Fungsi len pada dictionary digunakan untuk mengembalikan jumlah pasangan nilai kunci.

```
print(len(eng2sp)) # Output: 3
```

- Operator in dalam dictionary mengembalikan nilai benar (True) atau salah (False) sesuai dengan kunci yang ada dalam dictionary.

```
print('one' in eng2sp) # Output: True
print('uno' in eng2sp) # Output: False
```

- Untuk mengetahui nilai yang ada dalam dictionary, kita dapat menggunakan metode values, yang mengembalikan nilai dalam bentuk list.

```
vals = list(eng2sp.values())
print('uno' in vals) # Output: True
```

Python menggunakan algoritma Hash Table untuk dictionary, sehingga operator in membutuhkan waktu yang sama untuk memprosesnya, tidak memperdulikan jumlah item yang ada dalam dictionary.

DICTIONARY SEBAGAI SET PENGHITUNG (COUNTERS)

Dictionary dalam Python dapat digunakan sebagai set penghitung (counters) untuk menghitung kemunculan elemen-elemen dalam suatu list atau string. Dengan menggunakan dictionary sebagai set penghitung, kita dapat dengan mudah menghitung jumlah kemunculan setiap elemen tanpa perlu menggunakan loop secara manual.

Berikut adalah contoh penggunaan dictionary sebagai set penghitung untuk menghitung jumlah kemunculan setiap karakter dalam sebuah string:

```
# Contoh string
string = "hello"

# Menginisialisasi dictionary kosong sebagai set penghitung
counter = {}

# Menghitung kemunculan setiap karakter dalam string
for char in string:
    if char in counter:
        counter[char] += 1
    else:
        counter[char] = 1

# Menampilkan hasil set penghitung
print(counter)
```

Outputnya akan menjadi:

```
{'h': 1, 'e': 1, 'l': 2, 'o': 1}
```

Dalam contoh ini, setiap karakter dalam string diiterasi, dan kemudian diperiksa apakah karakter tersebut sudah ada dalam set penghitung (counter). Jika sudah ada, nilai untuk karakter tersebut ditambah 1. Jika belum ada, karakter tersebut ditambahkan ke set penghitung dengan nilai 1.

Dengan menggunakan dictionary sebagai set penghitung, kita dapat menghitung kemunculan setiap elemen dalam list atau string dengan cara yang efisien dan mudah dibaca.

DICTIONARY DAN FILE

Dalam Python, dictionary dapat digunakan untuk memanipulasi data dari file. File dalam Python bisa berupa file teks yang berisi data yang terstruktur atau tidak terstruktur. Berikut adalah beberapa cara umum menggunakan dictionary dengan file dalam Python:

1. Membaca Data dari File ke Dictionary:

```
data = {}
with open('data.txt', 'r') as file:
    for line in file:
        key, value = line.strip().split(':')
        data[key] = value
```

2. Menulis Data dari Dictionary ke File:

```
data = {'key1': 'value1', 'key2': 'value2', 'key3': 'value3'}
with open('data.txt', 'w') as file:
    for key, value in data.items():
        file.write(f'{key}:{value}\n')
```

3. Menambahkan Data dari Dictionary ke File yang Sudah Ada:

```
data = {'key4': 'value4', 'key5': 'value5'}
with open('data.txt', 'a') as file:
    for key, value in data.items():
        file.write(f'{key}:{value}\n')
```

4. Mengubah Data dalam File Menggunakan Dictionary:

```
data = {}
with open('data.txt', 'r') as file:
    for line in file:
        key, value = line.strip().split(':')
        data[key] = int(value) # Mengubah nilai menjadi integer (contoh)

# Mengubah nilai dalam dictionary
data['key1'] = 100
data['key2'] = 200

# Menulis kembali data yang telah diubah ke dalam file
with open('data.txt', 'w') as file:
    for key, value in data.items():
        file.write(f'{key}:{value}\n')
```

LOOPING DAN DICTIONARY

Looping digunakan untuk mengakses dan melakukan operasi pada setiap elemen dalam dictionary. Terdapat beberapa cara untuk melakukan looping pada dictionary dalam Python:

1) Looping melalui Kunci (Keys):

```
contoh.py > ...
1 data = {'a': 1, 'b': 2, 'c': 3}
2 for key in data:
3     print(key, data[key])
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/contoh.py
a 1
b 2
c 3
```

2) Looping melalui Item (Key-Value Pairs):

```
contoh.py > ...
1 data = {'a': 1, 'b': 2, 'c': 3}
2 for key, value in data.items():
3     print(key, value)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/contoh.py
a 1
b 2
c 3
```

3) Looping hanya melalui Nilai (Values):

```
contoh.py > ...
1 data = {'a': 1, 'b': 2, 'c': 3}
2 for value in data.values():
3     print(value)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/contoh.py
1
2
3
```

Dalam contoh di atas, data adalah dictionary yang akan di-looping. Dalam looping pertama, for key in data secara otomatis mengambil kunci-kunci dari dictionary. Dalam looping kedua, for key, value in data.items() mengambil pasangan kunci-nilai dari dictionary. Sedangkan, dalam looping ketiga, for value in data.values() hanya mengambil nilainya saja.

ADVANCED TEXT PARSING

Advanced text parsing dalam Python melibatkan penggunaan modul-modul seperti `re` (regular expressions) untuk menangani pola-pola teks kompleks, `NLTK` untuk pemrosesan bahasa alami, `Beautiful Soup` untuk parsing HTML, `Pandas` untuk parsing data tabular seperti CSV, dan `Spacy` untuk pemrosesan bahasa alami yang lebih canggih. Dengan menggunakan modul-modul ini, Anda dapat melakukan parsing teks dengan berbagai cara yang lebih efisien dan kuat.

Contoh singkat advanced text parsing menggunakan regular expressions (`re`) untuk mencari email dalam teks:

```

1  import re
2
3  text = "Hello, my email is example@example.com"
4  pattern = r'\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b'
5
6  emails = re.findall(pattern, text)
7  print(emails)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```

PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/contoh.py
['example@example.com']

```

Dalam contoh ini, regular expression pattern digunakan untuk mencari pola email dalam teks. Kemudian, `re.findall` digunakan untuk menemukan semua kemunculan pola email dalam teks.

BAGIAN 2: LATIHAN MANDIRI

SOAL 10.1

- Source Code :

```
Latihan 10,1.py > [?] dict
1 dict = {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
2
3 print(f"{'key':<4}{'value':<6}{'item':<5}")
4
5 for key, value in dict.items():
6     print(f"{'key':<4}{'value':<6}{'key':<5}")
```

- Output :

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe "c:/Tugas/Latihan 10,1.py"
key value item
1 10 1
2 20 2
3 30 3
4 40 4
5 50 5
6 60 6
```

- Penjelasan :

Dalam kode tersebut, sebuah kamus dibuat dengan pasangan kunci-nilai yang merepresentasikan hubungan antara angka-angka. Kemudian, menggunakan f-string, judul kolom ('key', 'value', 'item') dicetak dengan mengatur alignment agar output terlihat rapih.

Setelah itu, dilakukan iterasi melalui kamus menggunakan loop for dan metode items(). Pada setiap iterasi, nilai key dan value diambil dari pasangan kunci-nilai tersebut, dan dicetak dengan menggunakan f-string yang telah diformat sebelumnya agar sesuai dengan judul kolom yang telah ditentukan sebelumnya. Hasilnya adalah tampilan berformat tabel yang memperlihatkan kunci, nilai, dan item dari kamus tersebut.

SOAL 10.2

- Source Code :

```
Latihan 10,2.py > ...
1  lista = ['red', 'green', 'blue']
2  listb = ['#FF0000', '#008000', '#0000FF']
3
4  result_dict = dict(zip(lista, listb))
5
6  print(result_dict)
```

- Output :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  COMMENTS

PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe "c:/Tugas/Latihan 10,2.py"
{'red': '#FF0000', 'green': '#008000', 'blue': '#0000FF'}
```

- Penjelasan :

Dalam kode tersebut, dua list `lista` dan `listb` digunakan untuk menyimpan nama-nama warna dan kode warna RGB dalam format heksadesimal. Fungsi `zip` digunakan untuk menggabungkan kedua list tersebut menjadi sebuah list tuple yang berisi pasangan nilai yang sesuai berdasarkan urutan elemen di setiap list. Selanjutnya, fungsi `dict` digunakan untuk mengonversi list tuple tersebut menjadi sebuah kamus, di mana setiap elemen pertama dalam tuple menjadi kunci dan elemen kedua menjadi nilai. Hasilnya adalah kamus `result_dict` yang memetakan setiap warna dalam `lista` ke kode warna RGB yang sesuai dalam `listb`.

SOAL 10.3

- Source Code :

```
Latihan 10,3.py > ...
1 def count_emails_custom(filename_custom):
2     email_counts_custom = {}
3     with open(filename_custom, 'r') as file_custom:
4         for line_custom in file_custom:
5             if line_custom.startswith('From:'):
6                 email_custom = line_custom.split()[1]
7                 email_counts_custom[email_custom] = email_counts_custom.get(email_custom, 0) + 1
8     return email_counts_custom
9
10 filename_input = input("Masukkan nama file: ")
11 email_counts_result = count_emails_custom(filename_input)
12 print(email_counts_result)
```

- Output :

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
PS C:\Tugas> & C:/Users/ASIS/AppData/Local/Programs/Python/Python312/python.exe "c:/Tugas/Latihan 10,3.py"
Masukkan nama file: mbox-short.txt
{'stephen.marquard@uct.ac.za': 2, 'louis@media.berkeley.edu': 3, 'zqiang@umich.edu': 4, 'rjlowe@iupui.edu': 2, 'cwen@iupui.edu': 5, 'gsilver@umich.edu': 3, 'wagnerw@iupui.edu': 1, 'antranig@caret.cam.ac.uk': 1, 'gopal.ramasammycook@gmail.com': 1, 'david.horvitz@uct.ac.za': 4, 'ray@media.berkeley.edu': 1}
PS C:\Tugas>
```

- Penjelasan :

Kode ini adalah sebuah fungsi Python yang menghitung jumlah kemunculan setiap alamat email dalam sebuah file. Fungsi `count_emails` membaca file, mencari baris yang dimulai dengan 'From:', mengambil alamat email, dan menghitung jumlah kemunculan masing-masing alamat email. Hasilnya dicetak untuk ditampilkan kepada pengguna.

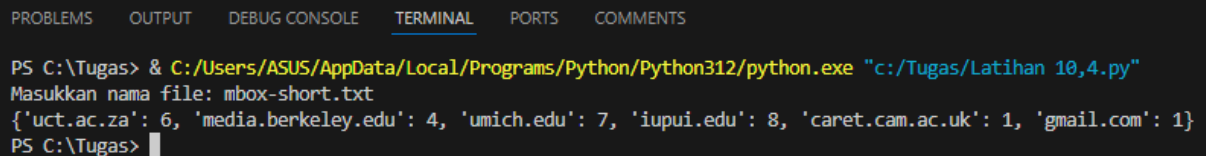
SOAL 10.4

- Source Code :

```
def count_email_domains_custom(filename_custom):
    email_domains_custom = {}
    with open(filename_custom, 'r') as file_custom:
        for line_custom in file_custom:
            if line_custom.startswith('From '):
                email_custom = line_custom.split()[1]
                domain_custom = email_custom.split('@')[1]
                email_domains_custom[domain_custom] =
email_domains_custom.get(domain_custom, 0) + 1
    return email_domains_custom

filename_input = input("Masukkan nama file: ")
result_custom = count_email_domains_custom(filename_input)
print(result_custom)
```

- Output :



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  COMMENTS
PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe "c:/Tugas/Latihan 10,4.py"
Masukkan nama file: mbox-short.txt
{'uct.ac.za': 6, 'media.berkeley.edu': 4, 'umich.edu': 7, 'iupui.edu': 8, 'caret.cam.ac.uk': 1, 'gmail.com': 1}
PS C:\Tugas> █
```

- Penjelasan :

Fungsi `count_email_domains` menghitung jumlah kemunculan setiap domain email dalam sebuah file. Variabel `domains` digunakan untuk menyimpan domain email dan jumlah kemunculannya. File dibaca baris per baris, dan jika baris dimulai dengan 'From ', domain email diekstraksi dan dihitung jumlah kemunculannya. Hasilnya dicetak untuk ditampilkan kepada pengguna.

- Link GitHub :

<https://github.com/Jamesmarvins/Tugas-Alpro-10.git>