



# Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

<b>NIM</b>	<b>71220895</b>
<b>Nama Lengkap</b>	<b>Drestanta Dipta Jalu Prakasya</b>
<b>Minggu ke / Materi</b>	<b>10 / Tipe Data Dictionary</b>

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

Dictionary adalah struktur data yang lebih umum daripada daftar. Dalam Dictionary, indeks bisa berupa apa saja, tetapi dalam daftar, indeks hanya berupa bilangan bulat. Dictionary terdiri dari pasangan kunci:nilai, dengan nilai berbeda untuk setiap kunci. Hal ini dapat dianggap sebagai pemetaan antara kunci dan nilai, seperti membuat Dictionary bahasa Inggris-Spanyol yang mana kata dalam bahasa Inggris adalah kuncinya dan kata dalam bahasa Spanyol adalah nilainya. Untuk membuat Dictionary baru, gunakan fungsi dict() dan gunakan tanda kurung siku [ untuk menambahkan item].

Contoh:

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

PROBLEMS OUTPUT TERMINAL PORTS

```
{'one': 'uno'}
PS C:\Users\User\Downloads\Laprak 10> []
```

Kuncinya sendiri adalah cara untuk mendapatkan nilai di Dictionary. Misalnya, nilai "dos" akan dikembalikan oleh eng2sp["two"]. Namun, mencoba menggunakan kunci yang tidak ada akan menghasilkan pengecualian.

```
materi.py > ...
1  eng2sp = dict()
2  eng2sp['one'] = 'uno'
3  # print(eng2sp)
4  print(eng2sp['four'])
```

PROBLEMS OUTPUT TERMINAL PORTS

```
Traceback (most recent call last):
  File "c:\Users\User\Downloads\Laprak 10\materi.py", line 4, in <module>
    print(eng2sp['four'])
          ~~~~~^~~~~~
KeyError: 'four'
PS C:\Users\User\Downloads\Laprak 10> []
```

Kita dapat menggunakan fungsi len(), untuk mengetahui berapa banyak pasangan kunci:nilai yang terdapat dalam Dictionary, dan operator in, untuk mengetahui apakah terdapat kunci dalam Dictionary.

```
materi.py > ...
1  eng2sp = dict()
2  eng2sp['one'] = 'uno'
3  # print(eng2sp)
4  # print(eng2sp['four'])
5  print(len(eng2sp))
6  print('one' in eng2sp)
7  print('uno' in eng2sp)
```

PROBLEMS OUTPUT TERMINAL PORTS

```
1
True
False
PS C:\Users\User\Downloads\Laparak 10> 
```

Kita bisa menggunakan metode value(), yang mengembalikan nilai dalam bentuk daftar, untuk mendapatkan semua nilai dalam Dictionary.

```
materi.py > ...
1  eng2sp = dict()
2  eng2sp['one'] = 'uno'
3  vals = list(eng2sp.values())
4  print('uno' in vals)
```

PROBLEMS OUTPUT TERMINAL PORTS

```
True
PS C:\Users\User\Downloads\Laparak 10> 
```

Dengan struktur data Hash Tabel yang digunakan untuk menyimpan Dictionary dengan Python, waktu yang diperlukan untuk memproses operasi input tetap konstan.

## MATERI 1

Di sini, kita mempelajari beberapa metode untuk menghitung berapa banyak huruf yang ada dalam sebuah string:

1. Menggunakan variabel terpisah untuk setiap huruf alfabet, kemudian menghitung frekuensinya menggunakan kondisi berantai.
2. Simpan frekuensi huruf dalam daftar dengan 26 elemen.
3. Simpan frekuensi huruf dalam daftar dengan 26 elemen.

Di antara ketiga pendekatan tersebut, penggunaan Dictionary adalah yang paling efisien karena tidak perlu mengetahui huruf mana yang akan muncul terlebih dahulu. Model implementasinya adalah sebagai berikut:

```
materi.py > ...
3   for c in word:
4       if c not in d:
5           d[c] = 1
6       else:
7           d[c] = d[c] + 1
8   print(d)
```

PROBLEMS OUTPUT TERMINAL PORTS

```
{'b': 1, 'r': 2, 'o': 2, 'n': 1, 't': 1, 's': 2, 'a': 1, 'u': 2}
PS C:\Users\User\Downloads\Laprak 10> 
```

Output program adalah histogram yang menunjukkan berapa kali setiap huruf muncul dalam string.

Untuk menyederhanakan perhitungan loop, kita juga dapat menggunakan metode get pada Dictionary:

```
materi.py > ...
1   word = 'brontosaurus'
2   d = dict()
3   for c in word:
4       d[c] = d.get(c, 0) + 1
5   print(d)
```

PROBLEMS OUTPUT TERMINAL PORTS

```
{'b': 1, 'r': 2, 'o': 2, 'n': 1, 't': 1, 's': 2, 'a': 1, 'u': 2}
PS C:\Users\User\Downloads\Laprak 10> 
```

Ini adalah "idiom" umum dalam Python untuk menghitung frekuensi dengan lebih cepat.

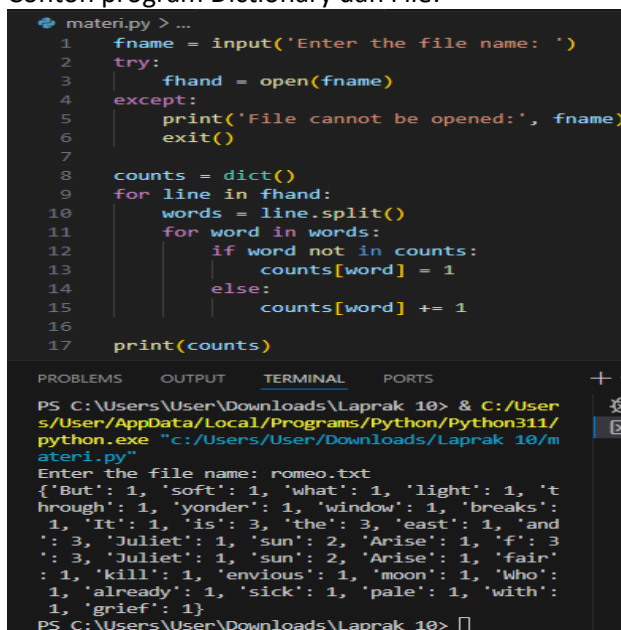
## MATERI 2

Dictionary dan File contoh sederhana penggunaan leksikon untuk menghitung kemunculan kata dalam file teks. Program ini membaca baris dari file teks, memecah setiap baris menjadi daftar kata, dan kemudian menggunakan leksikon untuk menghitung kemunculan setiap kata.

Berikut adalah penggunaan program penggunaan dari Dictionary dan File:

1. Masukkan nama file menggunakan input pengguna.
2. Buka file untuk dibaca.
3. Memberikan pesan error jika file tidak dapat dibuka.
4. Buat Dictionary kosong untuk menyimpan kemunculan kata.
5. Baca setiap baris dalam file menggunakan for loop.
6. Pisahkan setiap baris menjadi daftar kata menggunakan metode split().
7. Gunakan loop bersarang untuk mengulangi setiap kata di setiap baris.

Contoh program Dictionary dan File:



```
materi.py > ...
1  fname = input('Enter the file name: ')
2  try:
3      fhand = open(fname)
4  except:
5      print('File cannot be opened:', fname)
6      exit()
7
8  counts = dict()
9  for line in fhand:
10     words = line.split()
11     for word in words:
12         if word not in counts:
13             counts[word] = 1
14         else:
15             counts[word] += 1
16
17  print(counts)
```

PROBLEMS OUTPUT TERMINAL PORTS

```
PS C:\Users\User\Downloads\Laprak 10> & C:/Users/User/AppData/Local/Programs/Python/Python311/python.exe "c:/Users/User/Downloads/Laprak 10/materi.py"
Enter the file name: romeo.txt
{'But': 1, 'soft': 1, 'what': 1, 'light': 1, 'through': 1, 'yonder': 1, 'window': 1, 'breaks': 1, 'It': 1, 'is': 3, 'the': 3, 'east': 1, 'and': 3, 'Juliet': 1, 'sun': 2, 'Arise': 1, 'f': 3, 'fair': 3, 'kill': 1, 'envious': 1, 'moon': 1, 'Who': 1, 'already': 1, 'sick': 1, 'pale': 1, 'with': 1, 'grief': 1}
PS C:\Users\User\Downloads\Laprak 10>
```

Program ini sangat berguna untuk menganalisis teks dalam file dan menemukan kata-kata yang paling sering digunakan. Dengan menggunakan program ini, pengguna dapat dengan mudah melihat bagaimana kata-kata didistribusikan dalam teks.

### MATERI 3

Dictionary akan mencetak semua kunci dalam pernyataan for beserta nilainya. Outputnya tidak mengikuti pola sekuensial tertentu. Untuk mencetak kunci dalam urutan abjad, Anda harus terlebih dahulu membuat daftar kunci dari Dictionary menggunakan metode.keys(), lalu mengurutkan daftar tersebut. Kemudian, Anda menelusuri daftar yang diurutkan dan mencetak pasangan nilai kunci yang diurutkan. Ini kode dan outputnya:

```
materi.py > ...
1 counts = {'chuck': 1, 'annie': 42, 'jan': 100}
2 lst = list(counts.keys())
3 print(lst)
4 lst.sort()
5 for key in lst:
6     print(key, counts[key])
```

PROBLEMS OUTPUT TERMINAL PORTS + v ...

```
['chuck', 'annie', 'jan']
annie 42
chuck 1
jan 100
PS C:\Users\User\Downloads\Laprak 10> []
```

Kode yang disebutkan di atas mencetak kunci dalam urutan abjad yang memiliki pasangan nilai yang sesuai.

## MATERI 4

Pada bagian sebelumnya, kita menggunakan contoh file yang kalimat-kalimat di dalamnya telah dihapus tanda bacanya. Di bagian ini, kita akan menggunakan file yang sama, romeo.txt, dengan tanda baca lengkap. Fungsi split Python dapat mencari spasi dan mengubah kata menjadi token yang dipisahkan spasi. Misalnya kata "lambung!" dan "lunak" dianggap dua kata yang berbeda dan dipisahkan dalam Dictionary. Jika menggunakan huruf kapital, hal yang sama juga berlaku. Misalnya, kata "siapa" dan "siapa" dianggap kata yang berbeda dan harus dihitung secara terpisah. Untuk model penyelesaian lainnya, metode string lain dapat digunakan, seperti pengurangan, tanda baca, dan terjemahan. Terjemahan adalah metode string yang paling halus (hampir tidak terlihat). Metode ini menghasilkan string kosong untuk fromstr dan tostr serta menghapus semua karakter yang ada dalam deletestr. Untuk fromstr dan tostr, string kosong dapat dihasilkan, dan parameter deletestr dapat dihilangkan. Meskipun kita tidak akan menentukan tostr, kita akan menggunakan parameter deletestr untuk menghapus semua tanda baca. Python akan secara otomatis mengidentifikasi bagian mana yang dianggap sebagai "tanda baca".

Berikut beberapa contoh penggunaannya dalam program:

```
materi.py > ...
1  import string
2  fname = input('Masukkan nama file: ')
3  try:
4      fhand = open(fname)
5  except:
6      print('File tidak dapat dibuka:', fname)
7      exit()
8
9  counts = dict()
10 for line in fhand:
11     line = line.rstrip()
12     line = line.translate(line.maketrans('', '', string.punctuation))
13     line = line.lower()
14     words = line.split()
15     for word in words:
16         if word not in counts:
17             counts[word] = 1
18         else:
19             counts[word] += 1
20
21 print(counts)
22
```

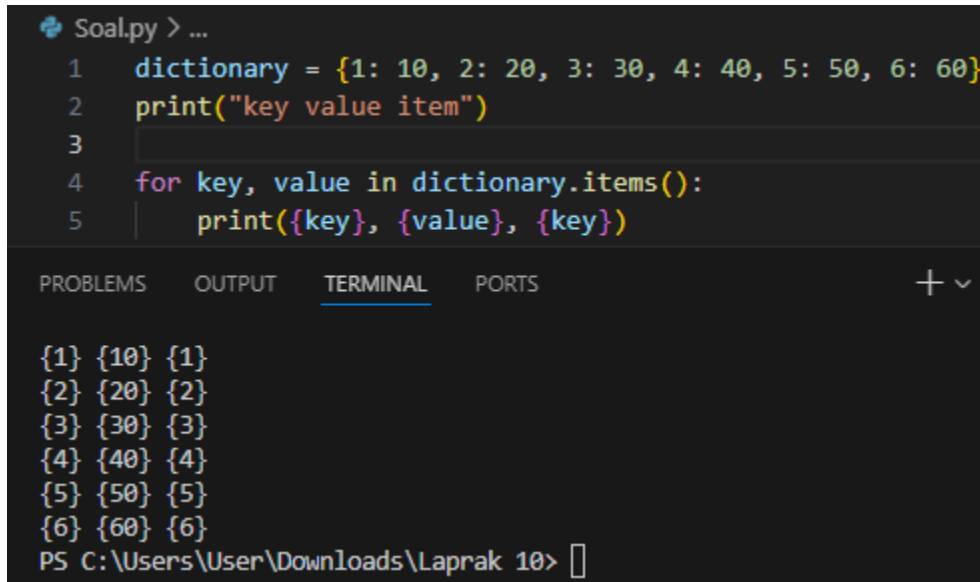
PROBLEMS   OUTPUT   TERMINAL   PORTS

```
Masukkan nama file: romeo.txt
{'but': 1, 'soft': 1, 'what': 1, 'light': 1, 'through': 1, 'yonder': 1, 'window': 1, 'breaks': 1, 'it': 1, 'is': 3, 'the': 3, 'east': 1, 'and': 3, 'juliet': 1, 'sun': 2, 'arise': 1, 'fair': 1, 'kill': 1, 'envious': 1, 'moon': 1, 'who': 1, 'already': 1, 'sick': 1, 'pale': 1, 'with': 1, 'grief': 1}
PS C:\Users\User\Downloads\Laprak 10> 
```

## 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

A screenshot of a Python IDE window titled 'Soal.py > ...'. The code editor shows five lines of Python code: 1. dictionary = {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}, 2. print("key value item"), 3. (empty line), 4. for key, value in dictionary.items():, 5. print({key}, {value}, {key}). Below the code editor is a terminal window with tabs for PROBLEMS, OUTPUT, TERMINAL, and PORTS. The TERMINAL tab is active and shows the output of the program: {1} {10} {1}, {2} {20} {2}, {3} {30} {3}, {4} {40} {4}, {5} {50} {5}, {6} {60} {6}. The prompt at the bottom of the terminal is PS C:\Users\User\Downloads\Laprak 10>.

```
Soal.py > ...
1 dictionary = {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
2 print("key value item")
3
4 for key, value in dictionary.items():
5     print({key}, {value}, {key})
```

PROBLEMS OUTPUT TERMINAL PORTS + v

```
{1} {10} {1}
{2} {20} {2}
{3} {30} {3}
{4} {40} {4}
{5} {50} {5}
{6} {60} {6}
PS C:\Users\User\Downloads\Laprak 10> 
```

Untuk mendapatkan pasangan nilai kunci dari kamus dalam program ini, kami menggunakan fungsi items(). Kemudian kami mengulangi setiap pasangan dan mencetak kunci, nilai, dan item (kunci pengembalian).



## SOAL 2

```
soal2.py > ...
1 list_a = ['red', 'green', 'blue']
2 list_b = ['#FF0000', '#008000', '#0000FF']
3
4 hasil = {key: value for key, value in zip(list_a, list_b)}
5
6 print(hasil)
```

PROBLEMS OUTPUT TERMINAL PORTS

```
{'red': '#FF0000', 'green': '#008000', 'blue': '#0000FF'}
PS C:\Users\User\Downloads\Laparak 10> 
```

Metode ini memungkinkan kita membuat kamus secara langsung menggunakan pemahaman kamus dengan mengulangi pasangan nilai yang dibuat oleh fungsi `zip()` dari dua daftar.

### SOAL 3

```
soal3.py > ...
1  from collections import Counter
2  file_name = input("Masukkan nama file: ")
3
4  try:
5      with open(file_name, 'r') as file:
6          email_count = Counter(line.split()[1] for line in file if line.startswith("From "))
7          print(email_count)
8
9  except FileNotFoundError:
10     print("File tidak ditemukan.")
```

PROBLEMS OUTPUT TERMINAL PORTS + v

Masukkan nama file: mbox-short.txt  
Counter({'cwen@iupui.edu': 5, 'zqian@umich.edu': 4, 'david.horwitz@uct.ac.za': 4, 'louis@media.berkeley.edu': 3, 'gsilver@umich.edu': 3, 'stephen.marquard@uct.ac.za': 2, 'rjlowe@iupui.edu': 2, 'wagnermr@iupui.edu': 1, 'ntranig@caret.cam.ac.uk': 1, 'gopal.ramasammycook@gmail.com': 1, 'ray@media.berkeley.edu': 1})  
PS C:\Users\User\Downloads\Laprak 10>

Code dijalankan dengan menggunakan 'defaultdict' untuk menghindari penggunaan '.get()' dan penanganan kasus dimana kunci belum ada dalam dictionary. Kita tidak perlu khawatir menangani kasus kunci yang belum ada karena kita membuat kamus\_count email menggunakan defaultdict(int). Jika kuncinya belum ada, Python secara otomatis menginisialisasi nilainya dengan int(), yaitu 0).

#### SOAL 4

Program Python berikut dapat digunakan untuk mencatat nama domain pengirim pesan dari file mbox-short.txt dan menghitung jumlah pesan yang dikirim untuk setiap domain:

```
soal4.py > ...
1  file_name = input("Masukkan nama file: ")
2
3  try:
4      with open(file_name, 'r') as file:
5          domain_count = {}
6          for line in file:
7              if line.startswith("From "):
8                  domain = line.split()[1].split('@')[1]
9                  domain_count[domain] = domain_count.get(domain, 0) + 1
10         print(domain_count)
11
12     except FileNotFoundError:
13         print("File tidak ditemukan.")
```

PROBLEMS   OUTPUT   TERMINAL   PORTS   + v

```
Masukkan nama file: mbox-short.txt
{'uct.ac.za': 6, 'media.berkeley.edu': 4, 'umich.edu': 7, 'iupui.edu': 8, 'caret.ca
m.ac.uk': 1, 'gmail.com': 1}
PS C:\Users\User\Downloads\Laparak 10> 
```

Program ini membaca setiap baris dalam file. Jika baris dimulai dengan "Dari", kami menggunakan split() untuk mengambil bagian domain dari alamat email pengirim dan mengambil elemen kedua setelah "@" (split('@')[1]). Kemudian, dengan menggunakan domain sebagai kunci, kami menggunakan kamus domain\_count untuk menyimpan jumlah pesan dari setiap domain. Jika domain sudah ada di kamus, kita tambah jumlahnya, dan jika belum, kita tambahkan nilai 1. Terakhir, kita cetak kamus yang berisi jumlah pesan dari setiap domain.