



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	71230985
Nama Lengkap	TOMAS BECKET
Minggu ke / Materi	11 / 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 (40%)

MATERI 1

PENGENALAN DICTIONARY

Dictionary, sering kali dibandingkan dengan list, dan lebih bersifat umum. Dalam list, indeks harus berupa integer, sementara dalam dictionary, indeks bisa berupa apa pun.

Sebuah dictionary terdiri dari pasangan kunci:nilai. Kunci harus unik, tidak boleh ada dua kunci yang sama dalam satu dictionary. Dictionary juga dapat diinterpretasikan sebagai pemetaan antara sekumpulan kunci (yang unik) dan sekumpulan nilai. Setiap kunci memetakan suatu nilai. Keterkaitan antara kunci dan nilai disebut pasangan nilai kunci (key-value pair) atau item. Sebagai contoh, dictionary dapat digunakan untuk memetakan kata-kata dari bahasa Inggris ke bahasa Spanyol, di mana kunci dan nilai berupa data string. Asumsinya setiap kata dalam bahasa Inggris memiliki satu arti dalam bahasa Spanyol.

Misal bawah ini contoh source kode yang membuat dictionary baru dengan tiga item dan melakukan pencetakan hasil yang didapatkan yaitu berupa keseluruhan data dari dictionary

```
1 eng2sp = {'one':'uno', 'two':'dos', 'three':'tres'}
2 print(eng2sp)
3 print(eng2sp['one'])
```

Kode tersebut membuat dictionary eng2sp yang memetakan kata-kata dalam bahasa Inggris ke bahasa Spanyol. Dictionary tersebut memiliki pasangan kunci:nilai seperti {'one':'uno', 'two':'dos', 'three':'tres'}. Kemudian, kode mencetak dictionary tersebut dan nilai yang terkait dengan kunci 'one', yaitu 'uno'

Adapun fungsi dictionary yang dapat digunakan untuk method sebuah values, dan method ini akan mengembalikan nilai sesuai dengan tipe datanya dan dikonversi kedalam list dan digunakan dalam operator in

Contoh source code terdapat pada module :

```
>>> vals = list(eng2sp.values())
>>> 'uno' in vals
True
```

MATERI 2

DICTIONARY Sebagai set penghitung (Counters)

Adapun instruksi atau cara untuk memmbuat kita bisa menghitung jumlah kemunculan setiap huruf dalam sebuah string dengan tiga model perhitungan yang berbeda.

Yang dimana pada Model pertama melibatkan pembuatan 26 variabel untuk mewakili setiap huruf dalam alfabet, kemudian kita memproses string karakter per karakter, menambahkan jumlah masing-masing huruf ke variabel yang sesuai.

```
1 kata = 'makan nasi'
2
3 d = dict() # Dict kosong
4 for i in kata:
5     if i not in d:
6         d[i] = 1
7     else:
8         d[i] = d[i] + 1
9 print(d)
```

Dan Model kedua menggunakan sebuah list dengan 26 elemen, di mana setiap elemen akan mewakili satu huruf dalam alfabet. Setiap karakter dalam string dikonversi menjadi angka yang sesuai dengan posisi huruf dalam alfabet, dan angka tersebut digunakan sebagai indeks untuk menambah jumlah yang sesuai dalam list.

SOURCE CODE TERDAPAT pada Module:

```
>>> counts = { 'chuck' : 1 , 'annie' : 42, 'jan': 100}
>>> print(counts.get('jan', 0))
100
>>> print(counts.get('tim', 0))
0
```

Dan Model ketiga menggunakan dictionary di mana setiap karakter dalam string menjadi kunci, dan jumlah kemunculannya menjadi nilai. Kita menambahkan atau mengupdate nilai-nilai ini saat memproses string.

```
1 counts = {'chuck' : 1, 'annie' : 42, 'jan' : 100}
2 print(counts.get('annie'))
3
4 kata = 'makan nasi'
5 d = dict()
6 for i in kata:
7     d[i] = d.get(i,0) + 1
8 print(d)
```

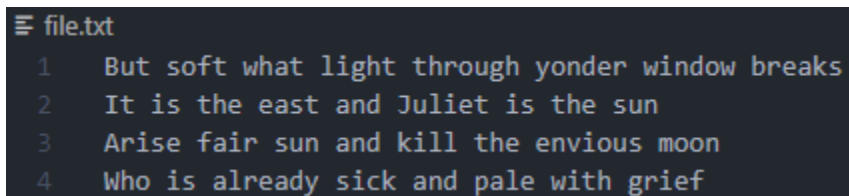
Dari ketiga model tersebut, model menggunakan dictionary lebih praktis karena kita tidak perlu mengetahui sebelumnya huruf mana yang akan muncul dalam string. Kita hanya perlu memberikan ruang untuk huruf-huruf yang mungkin muncul, dan model ini akan secara otomatis menyesuaikan diri dengan huruf-huruf yang ada dalam string.

MATERI 3

DICTIONARY AND FILE

Salah satu kegunaan umum dictionary adalah untuk menghitung kemunculan kata-kata dalam sebuah file teks.

Pertama, kita akan menggunakan versi teks yang disingkat dan disederhanakan tanpa tanda baca. Kemudian, kita akan menggunakan teks adegan yang lebih lengkap dengan tanda baca yang disertakan.



```
file.txt
1 But soft what light through yonder window breaks
2 It is the east and Juliet is the sun
3 Arise fair sun and kill the envious moon
4 Who is already sick and pale with grief
```

Kita akan menulis sebuah program Python untuk membaca setiap baris dari file teks, memecah setiap baris menjadi daftar kata, dan kemudian melakukan iterasi melalui setiap kata dalam setiap baris untuk menghitung kemunculan setiap kata menggunakan dictionary.

Asumsikan kita menggunakan dua perulangan for. Perulangan luar digunakan untuk membaca setiap baris dari file, sedangkan perulangan dalam digunakan untuk melakukan iterasi melalui setiap kata dalam baris tersebut.

Ini adalah contoh pola yang disebut nested loop karena satu perulangan adalah bagian dalam yang terletak di dalam perulangan luar. Perulangan dalam akan dieksekusi untuk setiap iterasi dari perulangan luar.

Dan yang Perlu diperhatikan yaitu bahwa perulangan dalam cenderung berjalan lebih cepat karena dieksekusi lebih sering dibandingkan perulangan luar, yang mungkin memiliki jumlah iterasi yang lebih besar.

penjelasan Source Code :

```
1  with open('file.txt', 'r') as file:
2      try:
3          fhand = file
4      except:
5          print('file cannot be opened', file)
6          exit()
7
8      counts = dict()
9      for line in fhand:
10         words = line.split()
11         for kata in words:
12             if kata not in counts:
13                 counts[kata] = 1
14             else:
15                 counts[kata] += 1
16     print(counts)
```

Kode tersebut membuka sebuah file teks bernama 'file.txt' dalam mode baca ('r'). Kemudian, kode mencoba membaca file tersebut dan menyimpannya dalam variabel fhand.

```
with open('file.txt', 'r') as file:
    try:
        fhand = file
```

Jika file tidak dapat dibuka, kode akan menampilkan pesan 'file cannot be opened' dan menghentikan eksekusi program.

```
counts = dict()
for line in fhand:
    words = line.split()
```

Setelah file berhasil dibuka, program menggunakan sebuah dictionary bernama counts untuk menghitung kemunculan setiap kata dalam file. Program melakukan iterasi melalui setiap baris dalam file menggunakan loop for line in fhand. Pada setiap baris, kata-kata dipisahkan menggunakan metode split() sehingga membentuk sebuah list words.

```
for kata in words:
    if kata not in counts:
        counts[kata] = 1
    else:
        counts[kata] += 1
```

Selanjutnya, program melakukan iterasi melalui setiap kata dalam list words menggunakan loop for kata in words. Untuk setiap kata, program memeriksa apakah kata tersebut sudah ada dalam dictionary counts. Jika kata tersebut belum ada dalam dictionary, maka program

menambahkan kata tersebut ke dalam dictionary dan memberikan nilai 1. Jika kata tersebut sudah ada dalam dictionary, maka program menambahkan 1 pada nilai yang sudah ada.

Setelah selesai menghitung kemunculan setiap kata dalam file, program mencetak dictionary counts yang berisi jumlah kemunculan setiap kata.

HASIL OUTPUT

```
{'and': 3, 'envious': 1, 'already': 1, 'fair': 1,
'is': 3, 'through': 1, 'pale': 1, 'yonder': 1,
'what': 1, 'sun': 2, 'Who': 1, 'But': 1, 'moon': 1,
'window': 1, 'sick': 1, 'east': 1, 'breaks': 1,
'grief': 1, 'with': 1, 'light': 1, 'It': 1, 'Arise': 1,
'kill': 1, 'the': 3, 'soft': 1, 'Juliet': 1}
```

MATERI 4

LOOPING AND DICTIONARY

Dalam statement for, dictionary akan bekerja dengan cara menelusuri kunci yang ada didalamnya. Looping ini akan melakukan pencetakan setiap kunci sesuai dengan hubungan nilainya.

```
1 counts = {'chuk' : 1, 'annie' : 42, 'jan' : 100}
2 for key in counts:
3     print(key, counts[key])
```

Ataupun contoh lain yang terdapat pada module

```
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])
```

Pada looping for yang melalui kunci dict perlu menambahkan operator index untuk mengambil nilai yang sesuai untuk setiap kunci.

MATERI 5

ADVANCE TEXT PARSING

Sebelumnya terdapat kode yang membahas tentang menghitung berapa kata yang terdapat pada sebuah file text dan sekarang bagaimana cara jika terdapat tanda koma, titik, tanda tanya, dsbg. Maka masalah ini bisa diselesaikan dengan fungsi split dan mengimport string yang ingin dihilangkan.

```
≡ file2.txt
1 But, soft! what light through yonder window breaks?
2 It is the east, and Juliet is the sun.
3 Arise, fair sun, and kill the envious moon,
4 Who is already sick and pale with grief,
```

PENJELASAN Source Code:

```
1 import string
2 string.punctuation
3 '!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
4
5
6 fname = input('Enter the file name: ')
7 try:
8     fhand = open(fname)
9 except:
10     print('File cannot be opened:', fname)
11     exit()
12
13 counts = dict()
14 for line in fhand:
15     line = line.rstrip()
16     line = line.translate(line.maketrans('', '', string.punctuation))
17     line = line.lower()
18     words = line.split()
19     for word in words:
20         if word not in counts:
21             counts[word] = 1
22         else:
23             counts[word] += 1
24
25 print(counts)
```

```
line = line.rstrip()
```

Metode `rstrip()` digunakan untuk menghapus karakter spasi tambahan di ujung baris.

```
line = line.translate(line.maketrans('', '', string.punctuation))
```

Metode `translate()` digunakan untuk menghapus tanda baca dari setiap baris. Fungsi `string.punctuation` dari modul `string` menyediakan string yang berisi semua tanda baca yang umum digunakan. Metode `maketrans()` digunakan untuk membuat tabel transisi yang akan mengonversi setiap karakter tanda baca menjadi karakter kosong (`' '`). Tabel transisi ini kemudian diterapkan ke setiap baris menggunakan metode `translate()`, sehingga menghapus semua tanda baca dari baris tersebut.

```
line = line.lower()
```

Metode `lower()` digunakan untuk mengubah semua huruf dalam baris menjadi huruf kecil, sehingga kata yang sama dengan huruf besar dan huruf kecil dianggap sama.

```
words = line.split()
```

Baris yang sudah diproses kemudian dipisahkan menjadi kata-kata menggunakan metode `split()`.

```
for word in words:
    if word not in counts:
        counts[word] = 1
    else:
        counts[word] += 1
```

Program melakukan iterasi melalui setiap kata dalam list `words` dan menghitung kemunculan setiap kata dalam dictionary `counts`.

BAGIAN 2: LATIHAN MANDIRI (60%)

SOAL 1

1.1 SOURCE CODE

```
Latihan1.py > ...
1  # LATIHAN 10.1
2  dictionary = {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
3
4  print("key".center(5), "value".center(7), "item".center(5))
5
6  for key, value in dictionary.items():
7      print(str(key).center(4), str(value).center(5), str(key).center(9))
```

1.2 PENJELASAN

Source code diatas menciptakan sebuah dictionary dengan nama dictionary yang memiliki beberapa pasangan key-value. Dan dictionary ini memiliki pasangan key dari 1 hingga 6, dengan nilai-nilai yang sesuai adalah 10, 20, 30, 40, 50, dan 60.

Kemudian, program tersebut akan mencetak header untuk tabel yang akan ditampilkan. Header ini berisi tiga kolom, yaitu "key", "value", dan "item".

Setelah header dicetak, program melakukan iterasi melalui setiap pasangan key-value dalam dictionary menggunakan loop `for key, value in dictionary.items()`. Pada setiap iterasi, nilai key, value, dan key tersebut di cetak dengan menggunakan metode `center()` yang menyelaraskan nilai-nilai tersebut ke tengah kolom yang ditentukan. Ini memberikan tampilan tabel yang rapi dan terstruktur.

1.3 HASIL OUTPUT

```
$ python -u "c:\Users\tomas\OneDrive\Dokumen\KULIAH\SEMESTER 2\(\PraAlpro) Praktikum Algoritma Pemro
● graman\PraAlpro Tugas 10\Latihan1.py"
key  value  item
1    10    1
2    20    2
3    30    3
4    40    4
5    50    5
6    60    6
```

SOAL 2

2.1 SOURCE CODE

```
Latihan2.py > ...  
1  # LATIHAN 10.2  
2  lista = ['red', 'green', 'blue']  
3  listb = ['#FF0000', '#008000', '#0000FF']  
4  
5  hasil = dict(zip(lista, listb))  
6  
7  print(hasil)
```

2.2 PENJELASAN

Terdapat List `lista` berisi nama warna ('red', 'green', 'blue') dan List `listb` berisi kode warna dalam format heksadesimal ('#FF0000', '#008000', '#0000FF').

Fungsi `zip()` digunakan untuk menggabungkan kedua list tersebut menjadi sebuah objek zip yang berisi pasangan nilai dari kedua list. Setiap elemen pada posisi yang sama dari kedua list akan menjadi satu pasangan nilai dalam objek zip tersebut.

Fungsi `dict()` kemudian digunakan untuk mengonversi objek zip tersebut menjadi sebuah dictionary. Objek zip tersebut menjadi pasangan key-value di dalam dictionary. Setiap elemen dari `lista` menjadi key dan setiap elemen dari `listb` menjadi value dalam dictionary hasil.

Hasilnya kemudian dicetak menggunakan `print()`, yang akan menampilkan dictionary yang terbentuk. Dalam contoh ini, dictionary tersebut akan berisi hubungan antara nama warna dan kode warna dalam format heksadesimal.

2.3 HASIL OUTPUT

```
$ python -u "c:\Users\tomas\OneDrive\Dokumen\KULIAH\SEMESTER 2\(\PraAlpro) Praktikum Algoritma Pemrograman\Pra  
Alpro Tugas 10\Latihan2.py"  
{'red': '#FF0000', 'green': '#008000', 'blue': '#0000FF'}
```

SOAL 3

3.1 SOURCE CODE

```
Latihan3.py > ...
1  # LATIHAN 10.3
2  nama_file = input("Masukkan nama file: ")
3
4  try:
5      file = open(nama_file, 'r')
6  except:
7      print("File tidak dapat dibuka:", nama_file)
8      exit()
9
10 pesan_masuk = dict()
11
12 for line in file:
13     if line.startswith("From "):
14         words = line.split()
15         email = words[1]
16         pesan_masuk[email] = pesan_masuk.get(email, 0) + 1
17
18 file.close()
19
20 print(pesan_masuk)
```

3.2 PENJELASAN

Pada soal diminta untuk menghitung berapa banyak pesan yang masuk dari email dan disajikan dalam bentuk dictionary sehingga penjelasan dari source code diatas :

1. Program meminta pengguna untuk memasukkan nama file teks.
2. File tersebut dibuka dalam mode baca ``r``. Jika file tidak dapat dibuka, program mencetak pesan kesalahan dan keluar.
3. Program membuat dictionary kosong `pesan_masuk` yang akan digunakan untuk menyimpan jumlah pesan dari setiap alamat email pengirim.
4. Program membaca setiap baris dari file satu per satu.
5. Untuk setiap baris yang dimulai dengan "From ", program memecah baris tersebut menjadi kata-kata menggunakan metode `split()`.
6. Alamat email pengirim berada pada indeks ke-1 setelah kata "From ". Program mengambil alamat email tersebut.

7. Program kemudian menambahkan alamat email pengirim ke dalam dictionary `pesan_masuk`. Jika alamat email tersebut sudah ada dalam dictionary, program akan menambahkan 1 pada nilai yang sudah ada. Jika belum ada, program akan menetapkan nilai 1 untuk alamat email tersebut.

8. Setelah selesai membaca semua baris dalam file, file ditutup.

9. Program mencetak dictionary `pesan_masuk`, yang berisi jumlah pesan yang dikirim dari setiap alamat email pengirim.

3.3 HASIL OUTPUT

```
$ python -u "c:\Users\tomas\OneDrive\Dokumen\KULIAH\SEMESTER 2\(\PraAlpro) Praktikum Algoritma Pemrograman\PraAlpro Tugas 10\Latihan3.py"
Masukkan nama file: mbox-short.txt
{'stephen.marquard@uct.ac.za': 2, 'louis@media.berkeley.edu': 3, 'zqian@umich.edu': 4, 'rjlowe@iupui.edu': 2, 'cwen@iupui.edu': 5, 'gsilver@umich.edu': 3, 'wagnermr@iupui.edu': 1, 'antranig@caret.cam.ac.uk': 1, 'gopal.ramasammycook@gmail.com': 1, 'david.horwitz@uct.ac.za': 4, 'ray@media.berkeley.edu': 1}
```

SOAL 4

4.1 SOURCE CODE

```
Latihan4.py > ...
1  # LATIHAN 10.4
2  nama_file = input("Masukkan nama file: ")
3
4  try:
5      file = open(nama_file, 'r')
6  except:
7      print("File tidak dapat dibuka:", nama_file)
8      exit()
9
10 pesan_per_domain = dict()
11
12 for line in file:
13     if line.startswith("From "):
14         words = line.split()
15         email = words[1]
16         domain = email.split('@')[1]
17         pesan_per_domain[domain] = pesan_per_domain.get(domain, 0) + 1
18
19 file.close()
20
21 print(pesan_per_domain)
```

4.2 PENJELASAN

Sama seperti soal dan penjelasan sebelumnya hanya saja pada kasus ini, kita diminta untuk menghitung jumlah pesan yang dikirim masing-masing domain yang disajikan dalam bentuk dictionary sehingga hanya perlu menambahkan dan merubah fungsi akhir pada iterasi line in file :

SOURCE CODE SOAL 10.3

```
pesan_masuk = dict()

for line in file:
    if line.startswith("From "):
        words = line.split()
        email = words[1]
        pesan_masuk[email] = pesan_masuk.get(email, 0) + 1
```

SOURCE CODE SOAL 10.4

```
pesan_per_domain = dict()

for line in file:
    if line.startswith("From "):
        words = line.split()
        email = words[1]
        domain = email.split('@')[1]
        pesan_per_domain[domain] = pesan_per_domain.get(domain, 0) + 1
```

4.3 HASIL OUTPUT

```
$ python -u "c:\Users\tomas\OneDrive\Dokumen\KULIAH\SEMESTER 2\{PraAlpro} Praktikum Algoritma Pemrograman\PraAlpro Tugas 10\Latihan4.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}
```