
# Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

| | |
|---|---|
| **NIM** | **71230982** |
| **Nama Lengkap** | **Rainie Fanita Chrisabel Hadisantoso** |
| **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**

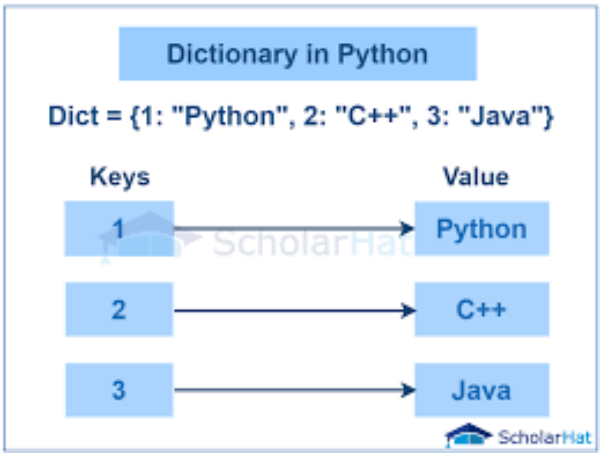
# MATERI : Tipe Data Dictionary

Pada pertemuan Tipe Data Dictionary, berikut beberapa tujuan dari pelajaran prakttikum:

1. Menjelaskan tentang dictionary
2. Menggunakan dictionary sebagai model penghitung
3. Membangun dictionary dari file
4. Menggunakan perulangan dari dictionary
5. Melakukan text parsing tingkat lanjut.

## Pengertian Dictionary

Dictionary sebetulnya mirip dengan list, namun bersigat lebih umum. Dalam list, indeks harus berupa interger dimulai dari 0, dalam dictionary indeks dapat berupa apapun. Divtionaty dapat disimpulkan adalah pemetaan data-data berisi anggota bersifat berpasangan atau 1 set= key:value. Hubungan antara key dan value dpat pula disebut sebagai pasangan nilai-kunci (key-value pair). Key adalah yang apa disebut dengan indeks dan bersifat unik atau tunggal, sebab key yang menjadi pegangan dalam dictionary. Sedangkan value adalah nilai data dari indeks yang telah ditentukan. Gambar dibawah merupakan gambaran pemetaan antara key dan value dalam sebuah dictionary.




Untuk membuat dictionary kita dapat mulai dengan mendifiniskan sebuah variabel baru yang akan menjadi nama dari dictionary. Kemudian untuk menginisiasi dictionary baru pada variabel tersebut digunakan fungsi dict(). Kata dict merupakan built-in function dalam python itu sendiri, jadi penggunaan skata dict sebagai nama variabel perlu dijauhi. Berikut contoh inisiasi dictionary untuk menyimpatn data berupa nama angka dalam bahasa indonesia dan terjemahannya dalam bahasa inggris. Untuk dictionarynya digunakan nama ariabel “idtoen”, kemudian diberikan fungsi dict(). Untuk sekarang jika diprint “idtoen” masih kosong.

```
idtoen = dict()
print(idtoen)
```

Kemudian kita mulai memasukkan data-data key dan value yang ingin dimasukkan. Caranya dengan memanggil nama variabel, mendefiniskan key, dan memberikan isi valuenya. Formatnya: nama[“key”] = “value”

```
idtoen = dict()
idtoen["satu"] = "one"
idtoen["dua"] = "two"
idtoen["tiga"] = "three"
print(idtoen)
#{"satu":"one", "dua":"two", "tiga":"three"}
```

Hasil output dari sebuah dictionary sesuai dengan format inputnya, jadi urutan isi output sesuai dengan urutan data dictionary dimasukkan. Namun hal tersebut tidak menjadi masalah karena elemen dalam dictionary tidak memiliki indeks berupa interger seperti pada list. Pencarian data pada dictionary bergantung sepenuhnya pada key-key yang telah diberikan.

```
idtoen = dict()
idtoen["satu"] = "one"
idtoen["tiga"] = "three"
idtoen["dua"] = "two"
print(idtoen)
#{"satu":"one", "tiga":"three", "dua":"two"}
```

Untuk pencarian data dan pemanggilan harus selalu menggunakan key. Apabila digunakan value, maka akan error. Hasil yang dikeluarkan dari pemanggilan key adalah value dari key tersebut.

```
print(idtoen["satu"])
#output : "one"
print(idtoen["one"])
#output = KeyError: "one"
```

Perlu diingat juga bahwa key dan value harus selalu berpasangan. Apabila salah satu tidak ada, maka program akan error. Untuk value kita dapat memberikan nilai kosong dengan “”, namn  tidak bisa kosong, None, tidak ada, hampa dan tak bermakna seperti hidupku.

```
idtoen["empat"] =
#error
idtoen["empat"] = ""
#ya bisa sih
idtoen["empat"] = "four"
#gitu dong
```

Kita dapat menggunakan fungsi le untuk dictionary. Namun len tidak akna menghitung panjang data dalam dictionary, en akan menghitung jumlah pasangan key-values yang da dalam sebuah dictionary.

```
len(idtoen)  #4
```

Operator in juga dapat diaplikasikan di dictionary dan mengembalikan true or false, sesuai dengan ketepatan kunci yang ada dalam dictionary. Namun in hanya dapat mengecek key dan bukan value. Lagi-lagi semuanya bergantung pada key.

```
"satu" in idtoen    #True
"one" in idtoen     #False
```

Untuk mengetahui nilai-nilai value dalam dictionary kita dapat menggunakan method values(). Metode values akan mengembalikan value-value dan menjadikannya dalam sebuah list.

```
value = list(idtoen.values())
"one" in value  #True
```

Pengaplikasian alggoritma operator in dalam list dan dictionary berbeda. List menggunakan algoritma linear. Ketika isi list bertambah, waktu mencari pun bertambah. Sedangkan dalam dictionary digunakan algoritms Hash Table, aku nggak tau itu kerjanya gimana tapi intinya seberapa banyak pasangan key-values dalam ictionary waktu yang diperlukan untuk memprosesnya tetap dama.

## Dictionary sebagai set penghitung (counters)

Contoh misalnya, ada string dan kita harus menghitung banyaknya jumlah tiap huruf yang muncul dalam string tersebut. Ada 3 cara yang dapat digunakan:

- Menggunakan variabel untuk tiap huruf pada alfabet, kemudian menambah perhitungan sesuai dengan karakter yang muncul dalam string menggunakan kondisional berantai.
- Menggunakan list dengan 26 elemen, mengkonversi setiap karakter menjadi angka, dan menggunakan angka sebagai indeks dalam list untuk menambah perhitungan.
- Menggunakan dictionary dengan karakter sebagai kunci dan perhitungan sebagai nilai, di mana karakter ditambahkan sebagai item dalam dictionary dan nilai diincrement.

Tentu saja meskipun ketiga cara diatas menerapkan proses yang berbeda-beda, mereka tetap menghasilkan output yang sama.

Implementasi menggunakan dictionary lebih praktis karena tidak memerlukan pengetahuan tentang huruf mana yang akan muncul dalam string. Modelnya adalah memberikan ruang untuk huruf yang akan muncul dan menambah nilai setiap kali huruf tersebut muncul dalam string.

```
word = "brontosaurus"
d = dict()
for c in word:
    if c not in d:
        d[c] = 1
    else:
        d[c] = d[c] + 1
```

```
print(d)
```

Program diatas menggunakan model komputasi histogram. Histogram adalah istilah statistika mengenai perhitungan frekuensi. Perulangan for akan mengecek setiap huruf dalam string. Tiap huruf akan dimasukkan dalam dictionary sebagai key. Jika belum ada key huruf dalam dictionary maka value adalah 1, jika sudah ada key huruf yang sama dalam dictionary maka nilai akan bertambah 1. Begitu terus hingga huruf-huruf dalam string habis.

Namun juga ada proses cara lain yang lebih sederhana dan efektif untuk melakukan program diatas, yaitu menggunakan metode get() pada dictionary. Metode get() berfungsi untuk emngambil vlaue yang dimiliki oleh sebuah key. Kita dapat memberikan value default dengan metode get. Formatnya adalah = nama.get( key, default value). Dapat dilihat dibawah, dengan menggunakan metode get, program menjadi lebih sederhana.

```
word = 'brontosaurus'
d = dict()
for c in word:
    d[c] = d.get(c,0) + 1
print(d)
```

Kedua metode menghasilkan output yang sama sebagai berikut:

```
{'b': 1, 'r': 2, 'o': 2, 'n': 1, 't': 1, 's': 2, 'a': 1, 'u': 2}
```

## Dictionary dan File

Dictionary memiliki banyak kegunaan, seperti beberapa diantaranya adalah untuk menghitung kemunculan kata dalam sebuah file. Berikut contoh program mencari dan menghitung kata-kata dalam sebuah file text.

```
file  = input("Masukkan nama file: ")
daftar = dict()
try:
    f = open(file)
except:
    print('File tidak dapat dibuka')
    exit()

for baris in f:
    isi = baris.split()
    for kata in isi:
        if word not in daftar:
            daftar[kata] = 1
        else:
            daftar[kata] += 1
```

```
print(daftar)
```

Dari kode diatas, ada beberapa bagian yang dituliskan:try-except, perulangan baris.perunagan kata. Jadi pertama, meminta pengguna untuk memasukkan nama file. Mendefiniskan variabel dengan fungsi dict() sebagai wadah data yang diminta. Kemudian try-except bersifat optional namun digunakan agar program tetap berjalan dan mengeluarkan output meskipun error karena kesalahan nama file. Jika lolos dari try-except maka kode masuk ke perulangan untuk membaca tiap baris dalam file. Dlam tiap baris file, kata-katanya dipisahkan dengan split(). Kemudian untuk setiap kata terpisah dalam baris dimasukkan dalam dictionary. Jika belum ada di dictionary maka jumlah 1, jika adda maka jumlah data dalam dictionary diambil dengan gel() kemudian ditambahkan 1. Terakhir dictionary diprint.

## Looping dan Dictionary

Statement for digunakan untuk melihat keys yang ada didalam dictionary, bukan value. Conoth perulagan berikut untuk menampilkan value tiap keynya.

```
idtoen = {"satu":"one", "tiga":"three", "dua":"two"}
for angka in idtoen:
    print(f"{angka} = {idtoen[angka]}"\n)
#output =
# satu = one
# tiga = three
# dua = two
```

Sebetulnya key tidak memiliki urutna tertentu dan dapat bersifat random. Untuk menampilkan key secara urutan alfabet, key dapat dimasukkan dalam list, kemudian di sort(), baru dilakukan perulangan untuk tiap key dalam list dan digunakan keynya untuk memanggil dictionary sesuai urutan key urut.

## Advanced Text Parsing

Python memiliki fungsi `split()` yang membagi teks menjadi kata-kata berdasarkan spasi. Ini memperlakukan kata-kata seperti "soft!" dan "soft" sebagai dua kata terpisah dalam kamus Penggunaan huruf kapital juga mempengaruhi pembagian kata, seperti "who" dan "Who" dianggap berbeda.

Metode translate() mengganti karakter dari satu set ke set lain dan menghapus karakter yang ada dalam set tertentu. Dalam kasus ini, karakter yang akan dihapus adalah tanda baca. Jika parameter tostr dikosongkan, Python secara otomatis

Metode alternatif seperti `lower()`, `punctuation`, dan `translate()` dapat digunakan untuk memproses teks dengan menghapus tanda baca dan mengubah huruf menjadi huruf kecil.Untuk text parsing yang lebih lannjut,kita dapat menggunakan library tambahan dengan import string. Dengan import string kita dapat menggunakan string.punctuation.String.punctuation berisi tanda-tanda tulisan selain alfabet. Dengan itu kita dapat memisahkan huruf dengan tanada baca dengan lebih mudah.

```
string.punctuation = '!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
```

Berikut contoh program yang menggunakan string.punctuation.Program berfungsi untuk memasukkan dan mendata jumlah kata dalam sebuah file text.

```
import string

fname = input('Enter the file name: ')
try:
    fhand = open(fname)
except:
    print('File cannot be opened:', fname)
    exit()

counts = dict()
for line in fhand:
    line = line.rstrip()
    line = line.translate(line.maketrans('', '', string.punctuation))
    line = line.lower()
    words = line.split()
    for word in words:
        if word not in counts:
            counts[word] = 1
        else:
            counts[word] += 1

print(counts)
```

# LATIHAN MANDIRI

## SOAL 10.1

Bualah sebuah program untuk mendapatkan nilai key, value, dan item dari sebuah dictionary.

a. Source code

```
d = {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}

print("Key","Value","Item",sep="\t")

i=1
for key in d:
  print(key,d[key],1,sep="\t")
  i+=1
```

b. Output

```
PS C:\Users\asus\Downloads\File> python -u "c:\Users\asus\Downloads\File\
1.py"
Key     Value   Item
1       10      1
2       20      1
3       30      1
4       40      1
5       50      1
6       60      1
```

c. Penjelasan

- Variabel dictionary “d” berisi data kunci-nilai.
- Menampilkan judul kolom "Key", "Value", dan "Item" dengan print() .digunakan sep=”\t” untuk memisahkan tiap kata dengan spasi tab.
- Digunakan loop untuk mengambil setiap data kunci-nilai dari “d” beserta nomor urutnya(i didefinisikan sebelum loop bernilai 1, dan tiap terjadi loop nilai i bertambah 1).
- Setiap terjadi perulangan diprint kunci, nilai, dan nomor urutnya.

## SOAL 10.2

Buatlah sebuah program untuk memetakan dua list mejadi satu dictionary.

a. Source code

```
lsa = ['red', 'green', 'blue']
lsb = ['#FF0000','#008000', '#0000FF']

d = dict()
for i in range (len(lsa)):
  d [lsa[i]] = lsb[i]
```

```
print(d)
```

b. Output

```
PS C:\Users\asus\Downloads\File> python -u "c:\Users\asus\Downloads\File
2.py"
{'red': '#FF0000', 'green': '#008000', 'blue': '#0000FF'}
```

c. Penjelasan

- Pertama, didefiisikan dua list: lsa yang berisi nama warna dan lsb yang berisi kode warna dalam format heksadesimal.
- Selanjutnya, didefinisikan dictionary kosong bernama d.
- Dilakukan loop for sebanyak panjang list lsa. Sehingga jumlah data yang dimasukkan sesuai dengan data dalam lsa
- Pada setiap perulangan diambil elemen dari lsa dan lsb pada indeks yang sama menggunakan lsa[i] dan lsb[i], kemudian keduanya dipasangkan sebagai kunci dan nilai di dalam dictionary d.
- Setelah loop selesai, dictionary d akan berisi pasangan kunci-nilai yang telah terbentuk dari list lsa dan lsb.
- Dicetak dictionary d, berisi nama warna dan kode warna.

## SOAL 10.3

Dengan menggunakan file mbox-short.txt, buatlah program yang dapat membaca log email dan sajikan dalam histogram menggunakan dictionary. Kemudian hitung berapa banyak pesan yang masuk dari email dan sajikan dalam bentuk dictionary.

a. Source code

```
file  = input("Masukkan nama file: ")
email = dict()
try:
    f = open(file)
except:
    print('File tidak dapat dibuka')
    exit()

for baris in f:
    isi = baris.split()
    if len(isi) < 2 or isi[0] != "From":
        continue
    else:
        email[isi[1]] = 1 +  email.get(isi[1],0)

print(email)
```

b. Output

```
PS C:\Users\asus\Downloads\File> python -u "c:\Users\asus\Downloads\File\Latihan_10_71230982\Latihan-10.
3.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}
```

c. Penjelasan

- Input nama file dari pengguna.
- Try except untuk mengect apakah file bisa dibuka atu tidak Jika tidak bisa dibuka, akan ditampilkan “File tidak dapat dibuka” dan menghentikan program. Jika dapat dibuka maka akan melanjutka program
- Digunakan perulangan untuk membaca setiap baris dalam file, dan setiap baris akan di pisahkan dengan split()
- Jika baris dimulai dengan "From", alamat email pada indeks ke 1 akan diambil dan dimasukkan ke dalam dictionary email dan jumlahnya. Jumlah email dihitung menggunakan get untuk menghitung jumlah yang sudah ada dalam dictionary dan menambanya dengan 1.
- Dictionary diprint

## SOAL 10.4

Dengan menggunakan file mbox-short.txt, buat program untuk mencatat data nama domain pengirim pesan. Hitunglah jumlah pesan yang dikirim masing-masing domain. sajikan dalam bentuk dictionary..

a. Source code

```
domain = dict()
file = input("Masukkan nama file: ")
try:
    f = open(file)
except:
    print("File tidak dapat dibuka")
    exit()

for baris in f:
    isi = baris.split()
    if len(isi) > 0  and isi[0] == "From:" :
        domain[isi[1][isi[1].find("@")+1:]]
=  domain.get(isi[1][isi[1].find("@")+1:],0) + 1

print(domain)
```

b. Output

```
PS C:\Users\asus\Downloads\File> python -u "c:\Users\asus\Downloads\File\Latihan_10_71230982\tempCodeRun
nerFile.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.c
om': 1}
```

c. Penjelasan

- Input nama file dari pengguna.

- Try except untuk membuka file. Jika file tidak dapat dibuka, akan ada pesan error dan keluar dari program.
- Perulangan untuk membaca tiap baris dalam file.
- Jika baris dimulai dengan "From:" dicari domain dari alamat email dari "@". Kemudian diambil substring dari alamat email dari karakter setelah "@" sebagai domain. Domain dimasukkan dalam dictionary dengan value jumlah domain dalam dict ditambah 1.
- Akhirnya, dictionary domain dicetak.

Link github:

https://github.com/rainiefch/Praktikum-Algoritma-Pemrograman_71230982/tree/eabf3d3bad2aed2d70358fa23e5cbc4c5b8a1bbc/Latihan%2010