III

String dan File

7. Pengolahan String

7.1 Tujuan Praktikum

Setelah mempelajari Bab ini, mahasiswa diharapkan dapat:

1. Dapat menjelaskan tentang String.
2. Dapat mengakses String dan memanipulasi String.
3. Dapat menggunakan operator metode String.
4. Dapat melakukan parsing String sederhana.

7.2 Alat dan Bahan

Praktikum ini membutuhkan perangkat komputer yang memiliki spesifikasi minimum sebagai berikut:

1. Terkoneksi ke Internet dan dapat mengunduh package-package Python.
2. Mampu menjalankan sistem operasi Windows 10 atau Ubuntu Linux.

Perangkat lunak yang diperlukan untuk mendukung praktikum ini adalah sebagai berikut:

1. Python 3.7 atau 3.8 yang terinstall menggunakan Anaconda atau Installer Python lainnya.
2. Web Browser (Mozilla Firefox, Microsoft Edge atau Google Chrome).
3. Command Prompt (jika menggunakan Windows).
4. Terminal (jika menggunakan Linux).
5. Editor Python (Visual Studio Code, PyCharm, Spyder atau editor-editor lainnya yang mendukung Python).

7.3 Materi

7.3.1 Pengantar String

String adalah untaian karakter-karakter yang menjadi satu kesatuan dan digunakan dalam program komputer untuk menyimpan kalimat, baik panjang ataupun pendek. String adalah suatu jenis data yang mampu menyimpan huruf / karakter dan disimpan dalam kode ASCII. Tidak semua bahasa pemrograman memiliki tipe data String, seperti misalnya bahasa C. Tipe data ini merupakan

jenis tipe data yang bukan tipe data dasar, karena tipe data ini pada dasarnya menyimpan lebih dari satu nilai tunggal sebagai satu kesatuan. Beberapa bahasa pemrograman ada yang menyebutkan bahwa String pada dasarnya adalah kumpulan tipe data karakter / array of character / list of character.

7.3.2 Pengaksesan String dan Manipulasi String

String dapat dibuat dengan sederhana menggunakan variabel:

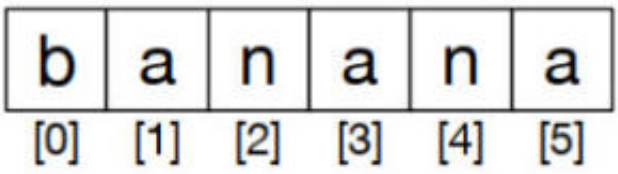
```
namasaya = "Antonius Rachmat C"
temansaya1 = "Yuan Lukito"
temansaya2 = 'Laurentius Kuncoro'
temansaya3 = "Matahari" + 'Bakti'

print(temansaya3)
print(namasaya[0]) #'A'
print(namasaya[9]) #'R'
print(temansaya1[1]) #'u'

huruf = temansaya2[0]
print(huruf) #'L'
```

Jadi string pertama kali dibuat dengan deklarasi variabel dan langsung diisi dengan data, string dapat diakses sebagai satu kesatuan dengan menyebut nama variabelnya, atau per huruf dengan menyebutkan indeksnya. Indeks pada string dimulai dari 0 seperti layaknya pada list. Indeks string haruslah berupa bilangan bulat, bukan pecahan.

Pada memory komputer, string disimpan secara urut menggunakan list yang berisi huruf-huruf dengan indeks yang dimulai dari nol. Gambar 7.1 menunjukkan string berisi "banana" yang disimpan pada memory komputer.



Gambar 7.1: Bentuk string di dalam memory

7.3.3 Operator dan Metode String

OPERATOR in

Pada String kita dapat memeriksa apakah suatu kalimat merupakan substring dari suatu kalimat lain dengan menggunakan operator in. Hasil dari operator ini adalah True / False.

```
kalimat = "saya mau makan"
data = "saya"
print(data in kalimat)  #True
print("mau" in kalimat) #True
print("dia" in kalimat) #False
```

Selain operator in, pada String juga dapat dilakukan perbandingan (comparison) yang juga menghasilkan True atau False.

```
if "saya" > "dia":
    print("Ya") #Ya
else:
    print("Tidak")

if "dua" == "dua":
    print("Sama")    #Sama
```

FUNGSI len

Cara untuk mengetahui berapa panjang (berapa jumlah karakter) dari sebuah string adalah dengan menggunakan operator len(<string>) Untuk menampilkan huruf terakhir dari sebuah string kita harus menggunakan indeks string yang ke- len(<string>-1), sebab indeks dimulai dari 0.

Contoh program Python adalah:

```
kalimat = "universitas kristen duta wacana yogyakarta"
print(len(kalimat)) #output 42

terakhir = kalimat[len(kalimat)-1]
print(terakhir) #output 'a'

#bisa juga menggunakan indeks -1
terakhir_versi2 = kalimat[-1]
print(terakhir_versi2)  #output 'a'
#atau menggunakan indeks -2 untuk huruf terakhir kedua
terakhir2 = kalimat[-2]
print(terakhir2)    #output 't'
```

TRAVERSING STRING

Untuk dapat menampilkan string dengan cara ditampilkan huruf demi huruf adalah dengan menggunakan loop yang dilakukan per huruf dengan 2 cara:

- Dilakukan dengan akses terhadap indeks

```
    kalimat = "indonesia jaya"
    i = 0
    while i < len(kalimat):
        print(kalimat[i],end='')
        i += 1
```

- Dilakukan tanpa akses terhadap indeks secara otomatis

```
    kalimat = "indonesia jaya"
    for kal in kalimat:
        print(kal,end='')
```

STRING SLICE

String slice adalah menampilkan substring pada sebuah string dengan menggunakan indeks dari awal tertentu sampai akhir-1 tertentu. Sintaksnya menggunakan <string>[awal:akhir]. Bagian awal atau akhir boleh dikosongkan. Bagian awal dimulai dari 0.

```
kalimat = "cerita rakyat"
awal = 0
akhir = 6
print(kalimat[awal:akhir])  #cerita
print(kalimat[7:len(kalimat)])  #rakyat
print(kalimat[:5])  #cerit
print(kalimat[5:])  #a rakyat
print(kalimat[:])  #cerita rakyat
```

String merupakan data yang bersifat immutable! Immutable adalah bahwa data tersebut tidak bisa diubah saat program berjalan, hanya bisa diinisialisasi saja. Contoh:

```
kalimat = "satu"
kalimat[0] = "batu" #TypeError: 'str' object does not support item assignment
```

Agar bisa diubah, maka harus disimpan dalam variabel yang berbeda.

```
kalimat = "satu"
kalimat_baru = kalimat[0] + "alah"  #salah
```

Berikut adalah beberapa method String yang sering digunakan:

Nama Method	Kegunaan	Penggunaan
capitalize()	untuk mengubah string menjadi huruf besar	string.capitalize()
count()	menghitung jumlah substring yang muncul dari sebuah string	string.count()
endswith()	mengetahui apakah suatu string diakhiri dengan string yang dinputkan	string.endswith()
startswith()	mengetahui apakah suatu string diawali dengan string yang dinputkan	string.startswith()
find()	mengembalikan indeks pertama string jika ditemukan string yang dicari	string.find()
islower() dan isupper()	mengembalikan True jika string adalah huruf kecil / huruf besar	string.islower() dan string.isupper()
isdigit()	mengembalikan True jika string adalah digit (angka)	string.isdigit()
strip()	menghapus semua whitespace yang ada di depan dan di akhir string	string.strip()
split()	memecah string menjadi token-token berdasarkan pemisah, misalnya berdasarkan spasi	string.split()

Dan masih banyak lainnya. Ingat semua fungsi/method di atas yang mengembalikan string, mengembalikan string baru, tidak mengubah yang aslinya, karena string bersifat *immutable*. Daftar lengkap dapat dilihat di `https://docs.python.org/library/stdtypes.html#string-methods`

Operator * dan + pada String

Pada Python operator + dan * memiliki kemampuan khusus. Operator + yang biasanya digunakan untuk menjumlahkan bilangan bisa digunakan untuk menggabungkan dua buah string. Sedangkan operator * yang bisa digunakan untuk mengkalikan bilangan bisa digunakan untuk menampilkan string sejumlah perkaliannya. Perhatikan kode berikut:

```
kata1 = "saya"
kata2 = "makan"
kata3 = kata1 + " " + kata2
print(kata3)    #hasil adalah penggabungan: saya makan
kata4 = "ulang"
print(kata4 * 4)    #hasil adalah ulangulangulangulang
kata4 = "ulang "
print(kata4 * 2)    #hasil adalah ulang ulang
```

7.3.4 Parsing String

Parsing string adalah cara menelusuri string bagian demi bagian untuk mendapatkan / menemukan / mengubah bagian string yang diinginkan. Perhatikan contoh berikut:

"Saudara-saudara, pada tanggal 17-08-1945 Indonesia merdeka".

Dilihat dari string di atas, diingikan untuk mengambil tanggal bulan tahun dari kalimat di atas dan disusun ulang menjadi format 08/17/1945. Kita dapat menggunakan cara parsing string misalnya:

- Spit string berdasarkan spasi sehingga menjadi token: "Saudara-saudara", "pada", "tanggal", "17-08-1945", "Indonesia", dan "merdeka".
- Dari sini, lanjutkan dengan mencari token yang diawali dengan angka, kemudian lanjutkan dengan split angka tersebut dengan pemisah '-'.
- Kemudian susun ulang token-token dari langkah sebelumnya sehingga berbentuk seperti yang diinginkan.

Berikut adalah contoh penyelesaiannya:

```
    kalimat = "Saudara-saudara, pada tanggal 17-08-1945 Indonesia merdeka"

    hasil = kalimat.split(" ")

    for kal in hasil:
        if kal[0].isdigit():
            hasil2 = kal.split("-")
            print(hasil2[1]+"/"+hasil2[0]+"/"+hasil2[2])
```

Hasil dari keluaran program adalah **08/17/1945**

7.4 Kegiatan Praktikum

Kasus 7.1 Buatlah program untuk menghitung berapa huruf hidup pada sebuah kalimat per kata

Kasus 7.2 Buatlah program untuk membuat mengubah susunan format tanggal dari YYYY-MM-DD menjadi DD-MM-YYYY

Kasus 7.3 Buatlah program untuk mengetahui apakah suatau kalimat adalah palindrom atau bukan! Palindrom adalah kalimat yang jika dibalik sama saja. Misalnya: Step on no pets, Pull up If I pull up, Some men interpret nine memos, dan Madam, In Eden I'm Adam.

Kasus 7.4 Buatlah program untuk mengambil beberapa kata dari suatu kalimat yang diinputkan. Misal kalimat: "saya akan pergi sekolah". Akan diambil 1 kata, maka hasilnya: "saya", "akan", "pergi", "sekolah". Sedangkan jika 2 kata maka hasilnya: "saya akan", "akan pergi", "pergi sekolah".

7.4.1 Pembahasan Soal 1

Untuk menghitung huruf hidup digunakan tahapan sebagai berikut:

- minta input kalimat yang akan dicari
- ubah string menjadi huruf kecil semua agar tidak membedakan huruf besar dan kecil
- siapkan variabel total jumlah huruf hidup aiueo dan set nilai awal = 0
- carilah ada berapa huruf 'a','i','u','e','o' pada kalimat yang diinputkan dengan method count()

Hasil:

```
a_string = "AnTonIus"
lowercase = a_string.lower()
total = 0
for x in "aiueo":
    jml = lowercase.count(x)
    total += jml
print(total)    #hasil = 4
```

7.4.2 Pembahasan Soal 2

Langkah yang harus dilakukan bisa 2 macam, yaitu dengan memecah / split string dan menyusunnya kembali sesuai yang diinginkan.

```
tgl = "2020-12-01"
hasil = tgl.split("-")
tgl2 = hasil[2]+"-"+hasil[1]+"-"+hasil[0]
print(tgl2)
```

7.4.3 Pembahasan Soal 3

Palindrom dapat diperiksa dengan membalik kata tersebut dan kemudian mencocokannya dengan string aslinya. Jika sama maka palindrom, jika tidak maka bukan palindrom.

```
satu = "Step! on, no.. pets??"
satu = satu.lower()

satu = ''.join([i for i in satu if i.isalpha()])    #buang semua yang bukan alfabet

dua = satu[::-1]
if dua == satu:
    print("palindrom")
else:
    print("bukan")
```

7.4.4 Pembahasan Soal 4

Untuk mengambil beberapa kata dari kalimat, digunakan logika sebagai berikut:

- Ubah kalimat menjadi huruf kecil semua (lowercase)
- Pecah (split), kalimat dipisah berdasarkan spasi
- Untuk setiap kata yang sudah dipisah tersebut, maka ambil indeks kata per n buah, misal 1 buah jika n=1, 2 jika n=2. Indeks yang diambil adalah dimulai dari 0, jika n=2 maka ambil 0 dan 1, lalu ambil 1 dan 2, 2 dan 3, 3 dan 4, dst...dilakukan dalam perulangan.

```
#Kalimat
text = 'A quick brown fox jumps over the lazy dog.'

def ambil_kata_kalimat(kalimat, n):
    kalimat = kalimat.lower()   #ubah huruf kecil
    hasil_akhir = []    #siapkan tempat hasil
    hasil = kalimat.split() #pecah berdasarkan spasi
    for i in range (0,len(hasil)):  #loop per hasil pecah
        tmp = ' '.join(hasil[i:i + n])  #ambil per indeks
        hasil_akhir.append(tmp) #tambahkan ke list

    return hasil_akhir  #return

print(ambil_kata_kalimat(text, 2))
```

Hasil

```
['a quick', 'quick brown', 'brown fox', 'fox jumps',
'jumps over', 'over the', 'the lazy',
'lazy dog.', 'dog.']
```

7.5 Latihan Mandiri

Latihan 7.1 Buatlah sebuah program yang dapat mendeteksi apakah suatu kata adalah anagram dari kata lainnya atau bukan! Anagram adalah kata yang dibolak-balik susunan hurufnya sama. Misal: mata anagram dengan atma, maat, taam, tama, dsb. ■

Latihan 7.2 Buatlah suatu program yang dapat menghitung frekuensi kemunculan suatu kata yang ada pada String. Misal terdapat kalimat "Saya mau makan. Makan itu wajib. Mau siang atau malam saya wajib makan". Ditanyakan kata "makan". Output: makan ada 3 buah ■

Latihan 7.3 Buatlah suatu program yang dapat menghapus semua spasi yang berlebih pada sebuah string, dan menjadikannya satu spasi normal! Misal: "saya tidak suka memancing ikan " Output: "saya tidak suka memancing ikan" ■

Latihan 7.4 Buatlah suatu program mengetahui kata terpendek dan terpanjang dari suatu kalimat yang diinputkan! Misal: "red snakes and a black frog in the pool" Output: terpendek: a, terpanjang: snakes ■