

CII-1A3- PENGENALAN PEMROGRAMAN

PERTEMUAN KE-9

PENGOLAHAN STRING



Outline

- Mengolah string
 1. Membuat string
 2. Mengubah case string → Lower, Upper
 3. Menggabungkan string → Concatenation
 4. Memisahkan dan menggabungkan string dari list → Split, Join
 5. Mencari dan mengganti substring → Find, Replace
 6. String formatting
- Membaca file teks
- Menggunakan regular expression

Mengolah String

1. Membuat string

- Single-line

```
'string 1 line'
```

```
"string 1 line"
```

- Multi-line

```
'''string multi-line'''
```

```
"""string multi-line"""
```

- Single-line yang bisa ditampilkan multi-line → newline

```
'ini juga string bisa\nmulti-line'
```

Membuat string (cont'd)

- Jika single-line tapi panjang isinya maka gunakan \

```
s6 = 'string ini panjang sekali sehingga melebihi layar monitor, \
kita harus scroll horizontal untuk melihat sisa stringnya. \
Tentu saja akan lebih baik jika kita pindahkan ke baris berikutnya \
hanya sekedar untuk kebutuhan tampilan saja'
```

```
print(s6)
s6
```

- Apakah hasil yang ditampilkan?

Contoh membuat string

Input : Tidak Bisa Oke

Output : ada 2 spasi



Bagaimana entry nilai untuk
multi-line ?

Cat: Inputan pengguna

Salah satu solusi

```
print("Input multi-line. Tekan Ctrl-D atau Ctrl-Z ( windows ) untuk menyimpannya.")
isi = []
while True:
    try:
        baris = input()
    except EOFError:
        break
    isi.append(baris)

print(isi)
```

2. Mengubah case string

- Terdapat 4 tipe penggunaan uppercase maupun lower case

`<string>.upper()`: mengubah semua huruf menjadi **kapital**

`<string>.lower()`: mengubah semua huruf menjadi **kecil**

`<string>.capitalize()`: mengubah **huruf pertama string** menjadi kapital

`<string>.title()`: mengubah **huruf pertama setiap kata** menjadi kapital

mengembalikan **string baru** dengan case yang telah diubah, **bukan** mengubah string asli.

Contoh mengubah case string

```
s6 = "s4y4 p1nt4r ju9A p4nda1"

print(s6.upper())
print(s6.lower())
print(s6.capitalize())
print(s6.title())
print(s6)
```

- Apa yang akan ditampilkan program ini?

3. Menggabungkan string

- Beberapa string dapat digabungkan menjadi sebuah string baru dengan tanda + (plus).

```
s1 = 'Telkom'  
s2 = 'University'  
  
s3 = s1 + ' ' + s2  
print(s3)
```

Contoh menggabungkan string

```
s6 = "S4y4"

print(s6.upper()+s6.lower())
print(s6.capitalize()+"\n"+s6.title()+"\n\n")
print(s6)
```

- Apa yang akan ditampilkan program ini?
- 

4. Memisahkan dan menggabungkan string dari list

- Menggabungkan → join
- Sintaks

`<separator>.join(<list>)`

- Contoh

```
list_string = ['directory', 'subdirectory', 'subsubdirectory', 'filename']  
path = '/'.join(list_string) #separator adalah tanda /  
print(path)
```

```
directory/subdirectory/subsubdirectory/filename
```

Memisahkan dan menggabungkan string dari list(2)

- Memisahkan → split
- Sintaks → default nilai dari <separator> adalah spasi

`<string>.split(<separator>)`

- Contoh

```
kalimat = 'Setiap hari    aku belajar Python agar aku bisa jadi programmer handal'  
list_kata = kalimat.split()  
print(list_kata)
```

```
['Setiap', 'hari', 'aku', 'belajar', 'Python', 'agar', 'aku', 'bisa', 'jadi', 'programmer', 'handal']
```

5. Mencari dan mengganti substring

- Mencari
 - a. Find : akan mengembalikan index awal substring yang **pertama yang ditemukan**, jika tidak ditemukan akan mengembalikan -1

Sintaks

```
<string>.find(<substring>)
```

Contoh

```
text = 'seratus dua puluh ribu rupiah (Rp 125.000)'  
start = text.find('(Rp')  
end = text.find(')')  
print(start, end)
```

30 41

Mencari dan mengganti substring(2)

- Mencari
 - b. index : sama seperti find(), namun jika tidak ditemukan maka akan menimbulkan **exception**

Sintaks

```
<string.index(<substring>)
```

Contoh

```
text = '(Rp 125.000)'  
start=3  
end=11  
nominal = text[start+1:end]  
print(nominal)
```

```
125.000
```

Mencari dan mengganti substring(3)

- Mengganti
mengganti **semua** substring lama dengan substring baru
string **immutable** → tidak mengubah string awal
tapi **mengembalikan string baru**.

Sintaks `<string>.replace(<old>, <new>)`

Contoh

```
text = 'seratus dua puluh ribu rupiah (Rp 125.000), dan lima puluh rupiah (Rp 50)'  
newtext = text.replace('(', '[')  
print(newtext)
```

```
seratus dua puluh ribu rupiah [Rp 125.000], dan lima puluh rupiah [Rp 50]
```


Membaca file teks

Membuka file

- File adalah data yang tersimpan di dalam komputer.

A. Membuka dan menampilkan isi file


Sintaks

`<var> = open (<nama_file>)`

Contoh

```
fname = 'file1.txt'  
fh = open(fname)  
for line in fh:  
    print(line, end="")
```

```
ini file baru  
coba
```

 file1.txt - Notepad

File Edit Format View Help
ini file baru
coba|

Bad filename

```
Traceback (most recent call last):  
  File "D:/PY/scratch_1.py", line 2, in <module>  
    fh = open(fname)  
FileNotFoundError: [Errno 2] No such file or directory: 'file11.txt'
```

- Untuk menghindarinya maka

```
fname = 'file11.txt'  
try:  
    fh = open(fname)  
except:  
    print("nama file salah")  
    quit()  
for line in fh:  
    print(line, end="")
```

nama file salah

Mode pada files

No	Mode	Description
1	'r'	Membuka file untuk dibaca (mode default)
2	'w'	Membuka file untuk ditulis. Jika file tidak ada maka membuat file baru. Jika file sudah ada maka isi file lama ditumpuk (replace)
3	'x'	Membuat file baru. Jika file sudah ada maka akan muncul pesan error
4	'a'	File dibuka untuk ditambahkan. Jika file tidak ada maka membuat file baru.
5	't'	Membuka file dalam bentuk teks (mode default)
6	'b'	Membuka file dalam bentuk binary
7	'+'	Membuka file untuk dibaca dan ditulis

Sintaks

`<var> = open (<nama_file>,<Mode>)`

Contoh:

- Menambahkan isi File kemudian mencetak / menampilkannya

```
f = open("file1.txt", "r")
print(f.read())

f = open("file1.txt", "a")
f.write("Tambahan isi")
f.close()

f = open("file1.txt", "r")
print(f.read())
```

```
ISI FILE LAMA
ISI FILE LAMATambahan isi
```

```
f = open("file1.txt", "r")
print(f.read())

f = open("file1.txt", "w")
f.write("Tambahan isi")
f.close()

f = open("file1.txt", "r")
print(f.read())
```

```
ISI FILE LAMA
Tambahan isi
```

Menggunakan regular expression

Regular Expression

- Bertujuan untuk mencari / mencocokkan sebuah pola pada sebuah string
- Contoh string = “**s**aya suka **s**api”, jika ada sebuah pertanyaan ingin mencari apakah ada pola “**sa**” pada string tersebut maka jawabannya adalah benar ada dan terletak di dua posisi yaitu (0,2) dan (10,12)
- Pada python regular expression (RegEx) memiliki library khusus yang dapat digunakan untuk mencari / mencocokkan pola yaitu dengan memanfaatkan import library dibawah ini

```
import re
```

Fungsi RegEx

No	Fungsi	Deskripsi
1	match	Mengecek kecocokan pola hanya di bagian awal dari string yang sedang dicari.
2	findall	Menampilkan semua pola yang ditemukan pada string
3	search	Menampilkan objek pola pertama yang cocok beserta posisi
4	split	Menampilkan string yang telah dipisahkan dari pola yang cocok
5	sub	Mengganti satu atau lebih pola yang cocok / sesuai

```
import re
```

```
txt = "saya suka python"
x = re.findall("[a,i,u,e,o]", txt)
x1 = re.search("py", txt)
x2 = re.split("py", txt)
x3 = re.sub("py","PY", txt)
print(x,"\n",x1,"\n",x2,"\n",x3)
```

```
['a', 'a', 'u', 'a', 'o']
<re.Match object; span=(10, 12), match='py'>
['saya suka ', 'thon']
saya suka PYthon
```


Contoh lain

```
import re
teks = "amazing training with love"
match = re.match(r'ama', teks)
if match:
    print (match.group(0))
    print (match.start(), match.end())
else:
    print ("no match")
```

raw (teksny dibaca utuh tanpa *special character*, misal \t akan dianggap apa adanya, bukan sebagai tab)

```
ama
0 3
```

Arti khusus *metacharacter*

No	Sintaks	Deksripsi
1	[]	Kumpulan karakter
2	\	Tanda urutan khusus / special karakter
3	.	Satu karakter apapun kecuali newline
4	^	Berawal dengan
5	\$	Berakhir dengan
6	*	Kemunculan dari kosong atau lebih
7	+	Kemunculan dari satu atau lebih

```
import re
```

```
txt = """Python itu mudah  
saya suka python"""
```

```
x = re.findall("[a-e]", txt)
```

```
['d', 'a', 'a', 'a', 'a']
```

```
x = re.findall("\n", txt)
```

```
['\n']
```

```
x = re.findall(".ython", txt)
```

```
['Python', 'python']
```

```
x = re.findall("^s", txt)
```

```
[]
```

```
x = re.findall("thon$", txt)
```

```
['thon']
```

```
x = re.findall("thon*", txt)
```

```
['thon ', 'thon']
```

```
x = re.findall("thon+", txt)
```

```
['thon ']
```

Apakah ada spasi yang mengikuti string thon?



Fakultas Informatika
School of Computing
Telkom University



THANK YOU

