



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	71230971
Nama Lengkap	James Marvin Santoso
Minggu ke / Materi	07 / Pengolahan String

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

Pengantar String

String dalam Python adalah tipe data yang digunakan untuk menyimpan teks. String dapat berisi satu atau lebih karakter, seperti huruf, angka, simbol, atau spasi. String dapat dibuat dengan mengapit teks menggunakan tanda kutip tunggal (' '), tanda kutip ganda (" "), atau tanda kutip tiga (''' ' atau '''' ''') untuk string multibaris.

Berikut adalah contoh dari penggunaan string, sebagai berikut :

```
Contoh.py > ...
1  # String dengan kutip tunggal
2  string1 = 'Halo, ini adalah string dengan tanda kutip tunggal'
3
4  # String dengan kutip ganda
5  string2 = "Halo, ini adalah string dengan tanda kutip ganda"
6
7  # String dengan kutip tiga
8  string3 = '''Halo,Ini adalah string dengan tanda kutip tiga,
9  |         |         bisa digunakan untuk string multibaris'''
10
11 # Untuk string
12 print(string1)
13 print(string2)
14 print(string3)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/Contoh.py
Halo, ini adalah string dengan tanda kutip tunggal
Halo, ini adalah string dengan tanda kutip ganda
Halo,Ini adalah string dengan tanda kutip tiga,
        bisa digunakan untuk string multibaris
PS C:\Tugas> 
```

Kode tersebut mendefinisikan tiga variabel string yang berbeda, `string1`, `string2`, dan `string3`, yang masing-masing berisi teks yang diapit oleh tanda kutip tunggal, tanda kutip ganda, dan tanda kutip tiga; kemudian, ketiga string tersebut dicetak ke layar menggunakan fungsi `print()`, memunculkan teks dari masing-masing string.

Pengaksesan String dan Manipulasi String

Pengaksesan dan manipulasi string dalam Python adalah proses memanipulasi teks. Anda dapat mengakses karakter individual dalam string, memotong string menjadi potongan-potongan yang lebih kecil, menggabungkan string, mengubah huruf menjadi huruf besar atau kecil, dan melakukan berbagai operasi lainnya untuk mengubah atau memanipulasi string sesuai kebutuhan Anda.

Berikut adalah contoh-contoh Pengaksesan String dan Manipulasi String :

1. Pengaksesan Karakter: Dalam string "Hello, World!", karakter pertama adalah 'H' dan karakter terakhir adalah '!'.
2. Slicing: Dengan string "Hello, World!", jika Anda mengambil potongan dari indeks 0 hingga 5, Anda akan mendapatkan "Hello". Jika Anda mengambil potongan dari indeks 7 sampai akhir, Anda akan mendapatkan "World!".
3. Panjang String: Panjang string "Hello, World!" adalah 13 karakter.
4. Penggabungan String: Menggabungkan string "Hello" dengan string "World" akan menghasilkan "Hello World".
5. Pengulangan String: Mengulang string "Hello" tiga kali akan menghasilkan "HelloHelloHello".
6. Mengubah Huruf Besar/Kecil: Mengubah "Hello, World!" menjadi huruf besar akan menghasilkan "HELLO, WORLD!", sedangkan mengubahnya menjadi huruf kecil akan menghasilkan "hello, world!".
7. Menghapus Spasi Tambahan: Menghapus spasi tambahan di awal dan akhir string " Hello, World! " akan menghasilkan "Hello, World!".
8. Pemisahan String: Memisahkan string "Hello, World!" dengan separator ", " akan menghasilkan list ['Hello', 'World!'].

Operator dan Metode String

- OPERATOR in
Operator in dalam Python digunakan untuk memeriksa apakah sebuah nilai atau elemen terdapat dalam sebuah urutan atau koleksi data seperti string, list, tuple, atau set. Operator in mengembalikan nilai True jika nilai atau elemen tersebut terdapat dalam urutan atau koleksi data tersebut, dan False jika tidak.

Contoh penggunaan operator in pada String :

```
Contoh.py > ...
1  string = "Hello, World!"
2  print("Hello" in string)
3  print("Python" in string)

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/Contoh.py
True
False
```

Kode tersebut memeriksa apakah sub-string "Hello" dan "Python" terdapat dalam string "Hello, World!", mengembalikan `True` untuk "Hello" dan `False` untuk "Python".

- FUNGSI len

Fungsi len() dalam Python digunakan untuk mengembalikan panjang dari sebuah urutan atau koleksi data seperti string, list, tuple, atau set. Panjang ini adalah jumlah elemen dalam urutan atau koleksi data tersebut.

Contoh penggunaan fungsi len pada String :

```
Contoh.py > ...
1 string = "Hello, World!"
2 print(len(string))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/Contoh.py
13
PS C:\Tugas> |
```

Kode tersebut menggunakan fungsi 'len()' untuk menghitung panjang string "Hello, World!" dan kemudian mencetak hasilnya, yang dalam hal ini adalah 13.

- TRAVERSING STRING

Traversing string dalam Python berarti mengunjungi setiap karakter dalam string satu per satu. Anda dapat melakukan traversing menggunakan loop for atau while untuk mengakses setiap karakter dalam string.

Contoh penggunaan traversing string untuk for dan while :

While:

```
Contoh.py > ...
1 string = "Hello, World!"
2 index = 0
3 while index < len(string):
4     print(string[index])
5     index += 1
```

For :

```
Contoh.py > ...
1 string = "Hello, World!"
2 for char in string:
3     print(char)
```

Hasil dari kedua kode tersebut sama yang dapat di lihat dibawah ini :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/Contoh.py
H
e
l
l
o
,

W
o
r
l
d
!
PS C:\Tugas> 
```

Contoh tersebut mencetak setiap karakter dalam string "Hello, World!" secara berurutan. Contoh pertama menggunakan `while` untuk menelusuri string dengan menginkremen `index` secara manual, sedangkan contoh kedua menggunakan `for` yang otomatis menelusuri setiap karakter dalam string tanpa mengelola indeks.

- **STRING SLICE**
String slice dalam Python digunakan untuk mengambil potongan atau sub-string dari sebuah string. Sintaksis umumnya adalah `string[start:stop:step]`, di mana start adalah indeks awal potongan, stop adalah indeks akhir (tidak termasuk), dan step adalah langkahnya (defaultnya 1)

Contoh penggunaan string slice, sebagai berikut :

```
Contoh.py > ...
1  string = "Hello, World!"
2  print(string[0:5])
3
4  print(string[7:])
5
6  print(string[::-2])

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

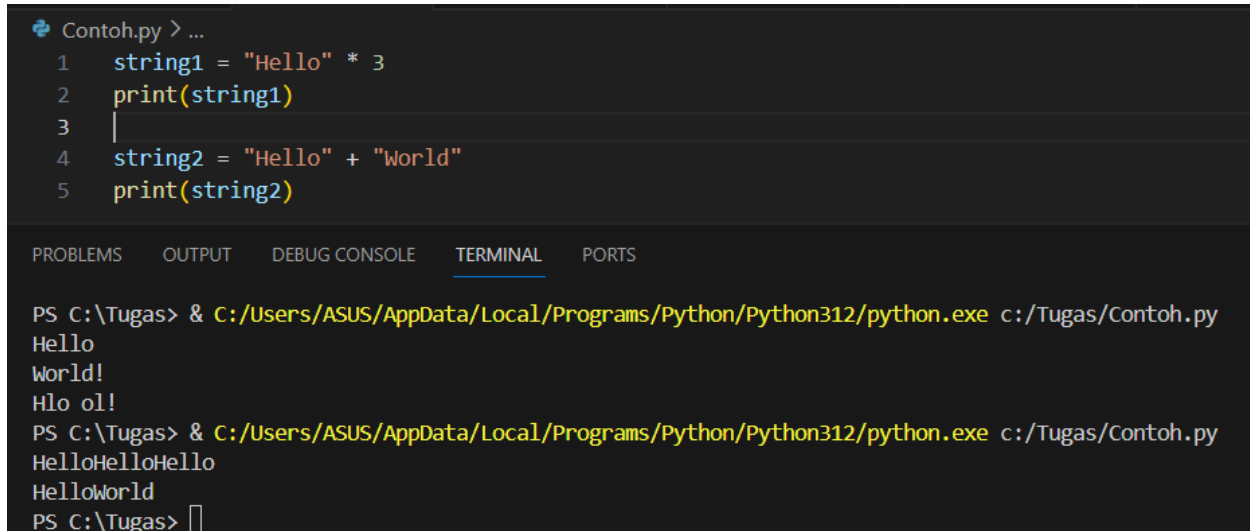
PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/Contoh.py
Hello
World!
Hlo ol!
PS C:\Tugas> 
```

Kode tersebut mengambil potongan-potongan string tertentu dari "Hello, World!" menggunakan string slice.

- Operator * dan + pada String

Pada Python, operator * digunakan untuk mengulang string sebanyak n kali, sedangkan operator + digunakan untuk menggabungkan atau menyambungkan dua string menjadi satu.

Contoh penggunaannya sebagai berikut :



```
Contoh.py > ...
1  string1 = "Hello" * 3
2  print(string1)
3
4  string2 = "Hello" + "World"
5  print(string2)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/Contoh.py
Hello
World!
Hlo ol!
PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe c:/Tugas/Contoh.py
HelloHelloHello
HelloWorld
PS C:\Tugas>
```

Dalam contoh di atas, operator * digunakan untuk mengulang string "Hello" sebanyak 3 kali, sedangkan operator + digunakan untuk menggabungkan string "Hello" dengan string "World".

Parsing String

Parsing string dalam Python adalah proses mengurai atau memecah string menjadi bagian-bagian yang lebih kecil atau lebih spesifik berdasarkan pola atau aturan tertentu. Hal ini sering digunakan dalam pengolahan teks dan analisis data untuk mengambil informasi tertentu dari string. Salah satu metode umum untuk parsing string adalah menggunakan metode `split()` untuk memecah string berdasarkan delimiter tertentu.

BAGIAN 2: LATIHAN MANDIRI

SOAL 7.1

- Source Code :

```
Latihan 7,1.py > ...
1  def is_anagram(word1, word2):
2      |   return sorted(word1.lower().replace(" ", "")) == sorted(word2.lower().replace(" ", ""))
3
4  kata1 = input("Masukkan kata pertama: ")
5  kata2 = input("Masukkan kata kedua: ")
6
7  if is_anagram(kata1, kata2):
8      |   print(f"{kata1} dan {kata2} adalah anagram.")
9  else:
10     |   print(f"{kata1} dan {kata2} bukan anagram.")
```

- Output :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe "c:/Tugas/Latihan 6,1.py"
Masukkan kata pertama: atma
Masukkan kata kedua: maat
atma dan maat adalah anagram.
PS C:\Tugas> |
```

- Penjelasan :

Fungsi `is_anagram` memeriksa apakah dua kata adalah anagram, yaitu memiliki huruf yang sama meskipun urutannya berbeda, dengan mengubah huruf menjadi huruf kecil dan menghapus spasi, lalu membandingkan hasil pengurutan kedua kata tersebut. Selanjutnya, program meminta input dua kata dari pengguna, memanggil fungsi `is_anagram` untuk memeriksanya, dan mencetak hasilnya.

SOAL 7.2

- Source Code :

```
Latihan 7,2.py > ...
1  def hitung_kemunculan_kata(kalimat, kata):
2      kata_kalimat = kalimat.split()
3      count = kata_kalimat.count(kata)
4      return count
5
6  kalimat = "Saya mau makan. Makan itu wajib. Mau siang atau malam saya wajib makan"
7  kata_dicari = "makan"
8
9  jumlah_kemunculan = hitung_kemunculan_kata(kalimat, kata_dicari)
10 print(f"{kata_dicari} ada {jumlah_kemunculan} buah")
```

- Output :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe "c:/Tugas/Latihan 7,2.py"
makan ada 1 buah
PS C:\Tugas> █
```

- Penjelasan :

Fungsi `hitung_kemunculan_kata` menghitung jumlah kemunculan sebuah kata dalam sebuah kalimat dengan memecah kalimat menjadi kata-kata, menghitung jumlah kemunculan kata yang dicari, dan mengembalikan hasilnya. Program kemudian menentukan kalimat dan kata yang akan diperiksa kemunculannya, memanggil fungsi `hitung_kemunculan_kata`, dan mencetak hasilnya.

SOAL 7.3

- Source Code :

```
Latihan 7,3.py > ...
1  def hapus_spasi_berlebih(kalimat):
2      kata_kalimat = kalimat.split()
3
4      kalimat_baru = " ".join(kata_kalimat)
5
6      return kalimat_baru
7
8  kalimat = "saya tidak suka memancing ikan "
9  kalimat_tanpa_spasi_berlebih = hapus_spasi_berlebih(kalimat)
10 print(kalimat_tanpa_spasi_berlebih)
```

- Output :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe "c:/Tugas/Latihan 7,3.py"
saya tidak suka memancing ikan
PS C:\Tugas> 
```

- Penjelasan :

Fungsi `hapus_spasi_berlebih` menghapus spasi berlebih dalam sebuah kalimat dengan memecah kalimat menjadi kata-kata, menggabungkan kembali kata-kata tersebut tanpa spasi berlebih, dan mengembalikan kalimat baru. Kalimat yang akan diubah dimasukkan, fungsi dipanggil, dan hasilnya langsung dicetak.

SOAL 7.4

- Source Code :

```
Latihan 7,4.py > ...
1 def kata_terpendek_terpanjang(kalimat):
2     kata_kalimat = kalimat.split()
3
4     kata_terpendek = min(kata_kalimat, key=len)
5     kata_terpanjang = max(kata_kalimat, key=len)
6
7     return kata_terpendek, kata_terpanjang
8
9 kalimat = "red snakes and a black frog in the pool"
10 terpendek, terpanjang = kata_terpendek_terpanjang(kalimat)
11 print(f"terpendek: {terpendek}, terpanjang: {terpanjang}")
```

- Output :

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Tugas> & C:/Users/ASUS/AppData/Local/Programs/Python/Python312/python.exe "c:/Tugas/Latihan 7,4.py"
terpendek: a, terpanjang: snakes
PS C:\Tugas> █
```

- Penjelasan :

Fungsi `kata_terpendek_terpanjang` mengembalikan kata terpendek dan terpanjang dalam sebuah kalimat dengan memecah kalimat menjadi kata-kata, kemudian mencari kata terpendek dan terpanjang menggunakan fungsi `min` dan `max` dengan parameter `key=len`. Selanjutnya, kalimat yang akan diperiksa dimasukkan, fungsi `kata_terpendek_terpanjang` dipanggil, dan hasilnya dicetak.

- Link GitHub :

<https://github.com/Jamesmarvins/Tugas-Alpro-7.git>