[COMPUTER PROGRAMING 2]

Sub- CPMK :
[Mahasiswa mampu memanipulasi string.]

PERTEMUAN 2

PART 1

A. String

Pada dunia pemrograman, ada satu tipe data yang berfungsi untuk menyimpan kumpulan dari karakter-karakter. Karakter-karakter tersebut tersusun menjadi satu-kesatuan membentuk sebuah kata, kalimat, atau paragraf yang bahkan bisa terbentuk dari digit dan juga numerik.

Pada python, String dibuat dengan kombinasi tanda petik tunggal (' ') atau tanda petik dua (" ").

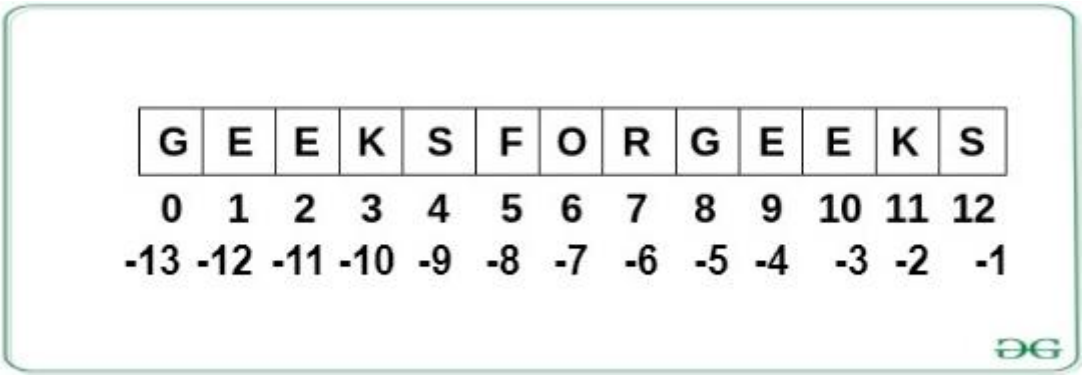
```
nama = ‘Ronaldo Wati’

asal = “Indonesia”
```

B. Mengakses Karakter dalam String

Di Python, karakter individual dari sebuah String dapat diakses dengan menggunakan metode Pengindeksan. Pengindeksan memungkinkan referensi alamat negatif untuk mengakses karakter dari belakang String, misalnya. -1 mengacu pada karakter terakhir, -2 mengacu pada karakter kedua terakhir, dan seterusnya.

Saat mengakses indeks di luar jangkauan akan menyebabkan IndexError. Hanya Integer yang boleh diteruskan sebagai indeks, float, atau tipe lain yang akan menyebabkan TypeError.



Dalam contoh ini, kita akan mendefinisikan string dengan Python dan mengakses karakternya menggunakan pengindeksan positif dan negatif. Elemen ke-0 akan menjadi karakter pertama dari string sedangkan elemen ke-1 adalah karakter terakhir dari string.

```
# Python Program to Access
# characters of String

String1 = "GeeksForGeeks"
print("Initial String: ")
print(String1)

# Printing First character
print("\nFirst character of String is: ")
print(String1[0])

# Printing Last character
print("\nLast character of String is: ")
print(String1[-1])
```

Output:

```
Initial String:
GeeksForGeeks
First character of String is:
G
Last cha racter of String is:
s
```

C. Escape Characters

Escape characters adalah karakter-karakter khusus yang digunakan dalam string untuk memberikan instruksi khusus kepada interpreter Python. Karakter-karakter ini dimulai dengan tanda \ (backslash) diikuti dengan karakter tertentu. Mereka memungkinkan Anda untuk menyisipkan karakter khusus, seperti tanda kutip atau karakter baris baru, ke dalam string. Di bawah ini adalah materi lengkap tentang escape characters beserta contoh-contohnya:

1. Karakter Baris Baru (\n)
Escape character \n digunakan untuk menyisipkan karakter baris baru. Ini berguna untuk membuat teks yang tampil dalam baris-baris terpisah.

Contoh:

```
teks = "Baris pertama\nBaris kedua"
print(teks)
```

Output:

```
Baris pertama
Baris kedua
```

2. Karakter Tab (\t)
Escape character \t digunakan untuk menyisipkan karakter tab horizontal.

Contoh:

```
teks = "Nama:\tJohn"
print(teks)
```

Output:

```
Nama:   John
```

3. Karakter Backslash (\\)
Untuk menyisipkan karakter backslash itu sendiri, Anda perlu menggunakan \ dua kali.

Contoh:

```
teks = "Ini adalah backslash: \\"
print(teks)
```

Output:

```
Ini adalah backslash: \
```

4. Karakter Tanda Kutip Tunggal (\') dan Tanda Kutip Ganda (\")
Anda dapat menggunakan escape characters \' atau \" untuk menyisipkan tanda kutip tunggal atau tanda kutip ganda dalam string yang diapit oleh tanda kutip yang sama.

Contoh:

```
teks1 = "Dia berkata, \"Halo!\""
teks2 = 'It\'s a beautiful day.'
```

Output:

```
Dia berkata, "Halo!"
It's a beautiful day.
```

5. Karakter Unicode (\u dan \U)
Anda dapat menggunakan escape characters \u dan \U untuk menyisipkan karakter Unicode dalam string.

Contoh:

```
teks1 = "\u03A9"  # Omega (Greek letter)
teks2 = "\U0001F609"  # Emoji (Winking Face)
```

Output:

```
Ω
😉
```

6. Karakter Khusus (\b, \r, dll.)
Escape characters lainnya termasuk \b (backspace), \r (return), \f (form feed), dan lain-lain. Masing-masing memiliki penggunaan khusus tergantung pada kebutuhan Anda.

Contoh:

```
teks = "Hello\bWorld"  # Menggunakan backspace
```

Output:

```
HeloWorld
```

Escape characters adalah karakter-karakter khusus yang digunakan dalam string Python untuk menyisipkan karakter khusus atau memberikan instruksi khusus kepada interpreter. Mereka berguna dalam berbagai situasi, terutama ketika Anda perlu menyisipkan karakter baris baru, karakter tab, atau tanda kutip dalam string. Dengan memahami escape characters, Anda dapat lebih baik mengendalikan format dan tampilan teks dalam program Python Anda.

D. Operasi pada String?

Concatenation (Penggabungan): Anda dapat menggabungkan (menggabungkan) dua atau lebih string dengan operator +.

Contoh:

```
teks1 = "Hello"
teks2 = "World"
gabungan = teks1 + " " + teks2
```

Output:

```
Hello World
```

Repetition (Pengulangan): Anda dapat mengulangi string dengan menggunakan operator *.

Contoh:

```
teks = "Python "
ulang = teks * 3
```

Output:

```
Python Python Python
```

E. Memotong String

Tidak jauh berbeda dengan list, kita juga bisa melakukan indexing dan slicing pada string. Secara sintaks pun sama saja.

Indexing String

Kita bisa mengambil karakter pada index ke-i pada string seperti ini:

```
nama = 'Lendis Fabri'
print(nama[4]) # output: i
print(nama[7]) # output: F
print(nama[-1]) # output: i
print(nama[-3]) # output: b
```

Penjelasan:

- Indeks dimulai dari 0 (ujung kiri ke kanan).
- Indeks negatif dimulai dari -1 yang dihitung dari karakter paling kanan ke paling kiri.

Slicing String

Slicing adalah teknik yang berguna dalam pemrograman Python untuk mengambil bagian atau subset dari tipe data berurutan seperti string, daftar (list), dan tuple. Slicing memungkinkan Anda untuk mengakses sejumlah elemen berurutan dari tipe data tersebut dengan sangat fleksibel. Di bawah ini adalah beberapa konsep penting yang perlu Anda pahami tentang slicing Python:

a. Sintaks Slicing:
   Sintaks dasar slicing adalah [**start:stop:step**], di mana:

   - start: Indeks elemen pertama yang ingin Anda ambil (termasuk).
   - stop: Indeks elemen terakhir yang ingin Anda ambil (tidak termasuk).
   - step: Langkah (step) yang digunakan untuk melompati elemen saat slicing (opsional, default adalah 1).

b. Indeks Negatif:
   Python mengizinkan penggunaan indeks negatif untuk menghitung dari belakang. -1 merujuk pada elemen terakhir, -2 merujuk pada elemen sebelumnya, dan seterusnya.

c. Indeks Default:
   Jika Anda tidak mengisi start, stop, atau step, maka Python akan menggunakan nilai default sebagai berikut:

   - start default adalah 0.
   - stop default adalah panjang (length) dari tipe data tersebut.
   - step default adalah 1.

d. Hasil Slicing:
   Hasil dari operasi slicing adalah tipe data yang sama dengan tipe data yang di-slice. Misalnya, jika Anda slicing string, hasilnya adalah string. Jika Anda slicing daftar, hasilnya adalah daftar.

Percobaan: Contoh Penggunaan Slicing
Mari kita lakukan beberapa percobaan untuk memahami penggunaan slicing dalam Python dengan berbagai tipe data:

1. Slicing pada String:

```
text = "Hello, World!"
# Ambil karakter dari indeks 0 hingga 5 (indeks 5 tidak
termasuk)
substring = text[0:5]
print(substring)  # Output: "Hello"

# Gunakan indeks negatif untuk mengambil karakter dari
belakang
last_chars = text[-6:-1]
print(last_chars)  # Output: "World"
```

2. Slicing pada List:

```
my_list = [1, 2, 3, 4, 5]
# Ambil elemen dari indeks 1 hingga 3 (indeks 3 tidak
termasuk)
sublist = my_list[1:3]
print(sublist)  # Output: [2, 3]

# Gunakan indeks negatif untuk mengambil elemen dari
belakang
last_elements = my_list[-3:]
print(last_elements)  # Output: [3, 4, 5]
```

3. Slicing pada Tuple:

```
my_tuple = (10, 20, 30, 40, 50)
# Ambil elemen dari indeks 1 hingga 4 (indeks 4 tidak
termasuk)
subtuple = my_tuple[1:4]
print(subtuple)  # Output: (20, 30, 40)

# Gunakan indeks negatif untuk mengambil elemen dari
belakang
last_items = my_tuple[-3:]
print(last_items)  # Output: (30, 40, 50)
```

Dengan memahami konsep slicing ini, Anda dapat dengan mudah mengambil bagian yang diperlukan dari tipe data berurutan dalam Python. Slicing sangat berguna dalam pengolahan data dan manipulasi string, daftar, atau tuple dalam program Anda.

F. Menghitung Panjang String

Panjang string adalah jumlah karakter dalam string tersebut. Dalam Python, Anda dapat dengan mudah menghitung panjang string menggunakan beberapa cara yang berbeda. Di bawah ini adalah materi lengkap tentang cara menghitung panjang string di Python beserta contoh-contohnya:

1. Fungsi len()
Fungsi len() adalah cara paling umum dan mudah untuk menghitung panjang sebuah string. Fungsi ini mengambil string sebagai argumen dan mengembalikan jumlah karakter dalam string tersebut.

Contoh:

```
teks = "Hello, World!"
panjang = len(teks)
print(panjang)  # Output: 13
```

2. Iterasi Menggunakan Loop
Anda juga dapat menghitung panjang string dengan mengiterasinya menggunakan loop dan menginkrementasi hitungan pada setiap iterasi.

Contoh:

```
teks = "Hello, World!"
panjang = 0
for karakter in teks:
    panjang += 1
print(panjang)  # Output: 13
```

3. Metode count()
Metode count() digunakan untuk menghitung berapa kali substring tertentu muncul dalam string. Dengan menghitung jumlah substring itu sendiri, Anda juga dapat mengetahui panjang string.

Contoh:

```
teks = "Hello, World!"
panjang = teks.count('') - 1  # Menghitung jumlah karakter
tanpa menghitung karakter terakhir ('\n')
print(panjang)  # Output: 13
```

4. Menggunakan while Loop
Anda juga dapat menghitung panjang string dengan menggunakan loop while yang akan berhenti ketika mencapai akhir string.

Contoh:

```
teks = "Hello, World!"
panjang = 0
indeks = 0
while teks[indeks:]:
    indeks += 1
    panjang += 1
print(panjang)  # Output: 13
```

Panjang string adalah jumlah karakter dalam string tersebut. Anda dapat menghitung panjang string di Python dengan mudah menggunakan fungsi len(), metode iterasi, metode count(), library textwrap, atau dengan mengggunakan

loop while. Fungsi len() adalah cara paling umum dan direkomendasikan untuk menghitung panjang string dalam Python. Dengan memahami cara menghitung panjang string, Anda dapat dengan mudah melakukan manipulasi dan analisis terhadap teks dalam program Anda.

G. Penggabungan String

Penggabungan string berarti membuat string baru dengan menggabungkan dua atau lebih nilai string. Banyak metode bawaan dan '+' operator digunakan untuk menggabungkan nilai string dalam banyak bahasa pemrograman. '+' operator juga digunakan dalam python untuk menggabungkan nilai string tetapi bekerja secara berbeda dari bahasa scripting lainnya. Dalam JavaScript, ketika sebuah nilai string digabungkan dengan nilai angka maka nilai angka tersebut akan dikonversi secara otomatis menjadi string dan digabungkan dengan nilai string lainnya. Tetapi jika Anda melakukan tugas yang sama di Python maka itu akan menghasilkan error karena Python tidak dapat mengubah angka menjadi string secara otomatis. Banyak cara lain yang ada di Python untuk menggabungkan nilai string. Artikel ini menunjukkan bagaimana Anda dapat melakukan penggabungan string dengan Python dengan cara yang berbeda.

1. Penggabungan String menggunakan operator '+'
   Buat file python dengan skrip berikut untuk memeriksa caranya '+'operator bekerja dengan Python untuk penggabungan string. Dua nilai string ditugaskan ke variabel bernamastr1 dan str2. Kedua nilai ini digabungkan dengan benar dan dicetak. Selanjutnya, satu nilai string dan nilai numerik ditugaskan ke variabel bernamatext dan price. Jika Anda ingin menggabungkan nilai-nilai ini maka itu akan menghasilkan error dengan menyebutkan bahwaint nilai dapat diubah menjadi str. Jadi, nilai numerik diubah menjadi nilai string dengan menggunakanstr() metode sebelum menggabungkan data.

```
# Define to string values
str1 = "I like "
str2 = "Programming"
# Combining a string value with another string value
combineText1 = str1+str2
# Print the combined output
print("Combining string with string:n",combineText1)
# Define a string value
text = "price of the book is "
# Define a number value
price = 50
# Combining a string value with a number value
combineText2 = text + "$" + str(price)
# Print the combined output
print("nCombining string with number:n",combineText2)
```

2. Penggabungan String menggunakan operator ‘%’
   Buat file python dengan skrip berikut untuk memeriksa caranya the ‘%’simbol berfungsi untuk penggabungan string dengan Python. Ini berfungsi seperti pemformatan string bahasa C. Di sini, dua nilai string ditetapkan dalam variabel, str1 dan str2. Anda dapat menggabungkan dua atau lebih nilai string dengan membuat grup yang dipisahkan koma dan menggunakan simbol ‘%’ di depan grup. Di sini, tanda kurung pertama, () digunakan untuk mengelompokkan nilai string dan ‘%s’ digunakan dalamprint() metode untuk mendefinisikan itu, nilai pencetakan adalah string.

```
#!/usr/bin/env python3
 # Define two string values
 str1 = "Python"
 str2 = "is a popular scripting language"

 # Combine the string values using '%' operator
 print("output after combining strings:nn%s %s" % (str1,
str2))
```

3. Penggabungan String menggunakan format() metode
   Jika Anda ingin menggabungkan nilai string lebih spesifik maka Anda harus menggunakan format()metode Python. Dengan menggunakan metode ini, Anda dapat menggabungkan nilai string berdasarkan posisinya. Posisi string dihitung sebagai 0,1,2 dan seterusnya. Dua nilai string diambil dari user dan ditugaskan ke variabel bernamastr1 dan str2. Selanjutnya, variabel-variabel ini digunakan dalamformat()metode sebagai argumen. Posisi variabel tidak disebutkan dalam skrip. Jadi, posisi variabel default adalah 0 dan 1.

```
#!/usr/bin/env python3
 # Define two string values
 str1 = input("Enter the first string valuen")
 str2 = input("Enter the second string valuen")

 # Combine the string values using format() operator
 combineText = "{} {}".format(str1, str2)

 # Print the combined text
 print("output after combining strings:nn",combineText)
```

4. Penggabungan String menggunakan join() metode
   join() adalah metode lain yang berguna dari Python untuk menggabungkan string. Jika Anda ingin menambahkan nilai string tertentu pada saat menggabungkan string maka Anda harus menggunakanjoin()metode untuk penggabungan. Buat file python dengan skrip berikut untuk memeriksa useranthe join()metode. Tiga nilai string ditetapkan dalam variabel

bernamastr1, str2, dan str3. Pertamajoin()digunakan untuk menggabungkan string tanpa string tertentu. Keduajoin()digunakan untuk menggabungkan nilai string dengan koma(). Ketigajoin() digunakan untuk menggabungkan nilai string dengan baris baru(n).

```
#!/usr/bin/env python3
 # Define two string values

 str1 = "Python Programmming"
 str2 = "Bash Programming"
 str3 = "Java Programming"

 # Using join() method to combine the strings
 combineText = "".join([str1, str2, str3])

 # Print the output
 print("nOutput:n%s" % combineText)

 # Using join() method with comma to combine the strings
 combineText = ",".join([str1, str2, str3])

 # Print the output
 print("nOutput:n%s" % combineText)

 # Using join() method with newline to combine the strings
 combineText = "n".join([str1, str2, str3])

 # Print the output
 print("nOutput:n%s" % combineText)
```

5. Menggabungkan string Tuple menggunakan metode join()
   Buat file python dengan skrip berikut. Di sini, metode join() akan menggabungkan nilai string dari Tuple dengan baris baru(n).

```
#!/usr/bin/env python3
 # Define a tuple of string values
 tupleString = ("Ubuntu","Windows", "MacOS",
"Fedora","Android","RedHat")

 # Combine the string values of the tuple using join()
method
 combineText = "n".join(tupleString)

 # Print the output
 print("nlist of operating systems are:nn%s" % combineText)
```

H. Perkalian String

Selain melakukan string concatenation menggunakan operator tambah (+), kita juga bisa menggunakan operator kali ('*').

Operator perkalian ini akan mengulang-ulang string yang dikalikan.

```
print('----------')
print('-' * 10)
```

=== To be Continue ===