functionPython

August 26, 2022

Nama : Muhammad Reesa Rosyid

Program : Python for Data Science

1 Judul

1.1 Fungsi pada Python

Fungsi adalah blok kode terorganisir dan dapat digunakan kembali yang digunakan untuk melakukan sebuah tindakan/action. Fungsi memberikan modularitas yang lebih baik untuk aplikasi Anda dan tingkat penggunaan kode yang tinggi.

Fungsi pada python: * Fungsi blok dimulai dengan def kata kunci diikuti oleh nama fungsi dan tanda kurung (()). * Setiap parameter masukan atau argumen harus ditempatkan di dalam tanda kurung ini. Anda juga dapat menentukan parameter di dalam tanda kurung ini. * Pernyataan pertama dari sebuah fungsi dapat berupa pernyataan opsional - string dokumentasi fungsi atau docstring. * Blok kode dalam setiap fungsi dimulai dengan titik dua (:) dan indentasi. * Pernyataan kembali [ekspresi] keluar dari sebuah fungsi, secara opsional menyampaikan kembali ekspresi ke pemanggil. Pernyataan pengembalian tanpa argumen sama dengan return None.

Contoh Fungsi:

1.1.1 Fungsi tanpa parameter

[7]:
```
def sapa():
    "Fungsi untuk print string yang ditulis dibawah ini"
    print("Halo peserta kursus Python")
    print("dari program Kampus Merdeka")

#eksekusi fungsi
sapa()
```

```
Halo peserta kursus Python
dari program Kampus Merdeka
```

1.1.2 Fungsi dengan parameter

Fungsi dengan satu parameter:

[8]:
```
#fungsi
def printme( str ):
   "This prints a passed string into this function"
   print (str)

#panggil fungsi
printme("Reesa")
```

```
Reesa
```

[4]:
```
def sapa_nama(nama):
    "Fungsi untuk print string yang ditulis dibawah ini"
    print(f"Halo {nama}")
    print("Saya dari program Kampus Merdeka")

sapa_nama("Reesa")
```

```
Halo Reesa
Saya dari program Kampus Merdeka
```

Fungsi dengan lebih dari satu paramter:

[6]:
```
def penjumlahan(angka_1, angka_2):
    "Fungsi ini akan menjumlahkan 2 angka yang dimasukkan kedalam fungsi"
    print(f"Hasil dari {angka_1} + {angka_2} adalah {angka_1 + angka_2}")

penjumlahan(100, 123)
```

```
Hasil dari 100 + 123 adalah 223
```

1.1.3 Fungsi dengan mengembalikan nilai

[9]:
```
# fungsi yg mengembalikan nilai ?
def penjumlahan_2(angka_1, angka_2):
    "Fungsi ini akan menjumlahkan 2 angka yang dimasukkan kedalam fungsi"
    return angka_1 + angka_2
penjumlahan_2(10, 9)
```

[9]:
```
19
```

1.1.4 Fungsi dengan *argh

*args memungkinkan kita untuk meneruskan sejumlah variabel argumen non-kata kunci ke fungsi Python. Dalam fungsi tersebut, kita harus menggunakan tanda bintang ( ) sebelum nama parameter untuk meneruskan sejumlah variabel argumen.

[10]:
```
# *args params in function
def penjumlahan(*angka):
    hasil = 0
    for item in angka:
```

```
        hasil += item
    return hasil

penjumlahan(1,2,3,4,5,3,4,5,6,67,3,4,5,6,3)
```

[10]: 121

1.1.5 Fungsi dengan **kwargs

Sintaks khusus **kwargs dalam definisi fungsi dalam python digunakan untuk meneruskan daftar argumen panjang variabel dengan kata kunci. Kami menggunakan nama kwargs dengan bintang ganda. Alasannya adalah bahwa bintang ganda memungkinkan kita untuk melewati argumen kata kunci (dan berapa pun jumlahnya).

[11]:
```
# **kwargs params in function
def biodata(**data):
    print("Biodata Pegawai")
    for k, v in data.items():
        print(f"{k}\t: {v}")

biodata(Nama = "Reesa", Usia = 20)
```

```
Biodata Pegawai
Nama    : Reesa
Usia    : 20
```

1.2 Variable global & local

Variable global adalah variabel yang bisa dipanggil dari manapun dari satu file python. Sedangkan variable lokal adalah variabel yang hanya hidup di dalam satu blok kode tertentu.

Contoh variable global:

[12]:
```
my_var = 'Hacktiv8'

def fungsi_1():
    print(f"Jangan lupa subscribe channel {my_var}")

def fungsi_2():
    print(f"Memanggil var {my_var} dari fungsi 2")

fungsi_1()
fungsi_2()
```

```
Jangan lupa subscribe channel Hacktiv8
Memanggil var Hacktiv8 dari fungsi 2
```

Contoh variable local:

[15]:
```python
# local variable
def fungsi_3():
    nama = "Reesa"
    print(f"nama saya {nama}")
    print("String diatas diprint didalam fungsi")
fungsi_3()
```

```
nama saya Reesa
String diatas diprint didalam fungsi
```

1.3 Catching Error

Dalam Python, Exception dapat ditangani menggunakan pernyataan try. Operasi kritis yang dapat memunculkan eksepsi ditempatkan di dalam klausa try. Kode yang menangani Python Exception ditulis dalam klausa except.

Contoh:

[16]:
```python
try:
    print("Halo apa kabar")
    print(halo)
except Exception as e:
    print(str(e))
print(1+1)
```

```
Halo apa kabar
name 'halo' is not defined
2
```

1.4 Fungsi Rekrusif

Fungsi rekursif merupakan sebuah metode perulangan yang bersifat non-iterasi.

Contoh:

[19]:
```python
# fungsi rekursif
def faktorial(n):
    if n < 0:
        print("Error, tidak bisa melakukan faktorial untuk angka negatif")
    elif n == 0 or n == 1:
        return 1
    else:
        return n * faktorial(n-1)
faktorial(10)
```

[19]:
```
3628800
```