

Function dan Procedure

Bagian 1



Pengantar (1)

Function adalah suatu **program terpisah** dalam blok sendiri yang **berfungsi sebagai sub-program** (modul program) yang merupakan **sebuah program kecil** untuk **memproses sebagian dari pekerjaan** program utama.

Pengantar (2)

- Function digunakan untuk **membuat blok program tersendiri**
- Untuk **memudahkan pembacaan** dari perintah program
- Membuat **program lebih terstruktur**, dll

Pengantar (3)

1. Program besar dapat di pisah-pisah menjadi program-program kecil melalui function.
2. Kemudahan dalam mencari kesalahan-kesalahan karena alur logika jelas dan kesalahan dapat dilokalisasi dalam suatu modul tertentu.
3. Memperbaiki atau memodifikasi program dapat dilakukan pada suatu modul tertentu saja tanpa mengganggu keseluruhan program.
4. Dapat digunakan kembali (Reusability) oleh program atau fungsi lain.
5. Meminimalkan penulisan perintah yang sama.

Pengantar (3)

1. Program besar dapat di pisah-pisah menjadi program-program kecil melalui function.
2. Kemudahan dalam mencari kesalahan-kesalahan karena alur logika jelas dan kesalahan dapat dilokalisasi dalam suatu modul tertentu.
3. Memperbaiki atau memodifikasi program dapat dilakukan pada suatu modul tertentu saja tanpa mengganggu keseluruhan program.
4. Dapat digunakan kembali (Reusability) oleh program atau fungsi lain.
5. Meminimalkan penulisan perintah yang sama.

Kategori Function (1)

1

Standard Library Function

fungsi-fungsi yang telah disediakan oleh Interpreter Python dalam file-file atau librarynya.

2

Programme-Defined Function

function yang dibuat oleh programmer sendiri. Function ini memiliki nama tertentu yang unik dalam program, letaknya terpisah dari program utama, dan bisa dijadikan satu ke dalam suatu library buatan programmer itu sendiri

Kategori Function (2)

Dalam python terdapat dua perintah yang dapat digunakan untuk membuat sebuah fungsi, yaitu **def** dan **lambda**.


def adalah perintah standar dalam python untuk mendefinisikan sebuah fungsi.

Sedangkan **lambda**, dalam python lebih dikenal dengan nama Anonymous Function (Fungsi yang tidak disebutkan namanya).

Function def (1)

Bentuk umum untuk mendeklarasikan fungsi adalah sebagai berikut :

python

 Copy code

```
def add(x, y):  
    print(f'nilai x: {x} dan y: {y}')
```

```
    return x + y
```


Function def (2)

Berikut adalah aturan sederhana untuk mendefinisikan sebuah fungsi dalam Python:

- Blok fungsi dimulai dengan kata kunci `def` diikuti oleh nama fungsi dan tanda kurung `()`.
- Semua parameter atau argumen masukan harus ditempatkan di dalam tanda kurung ini.
- Anda juga dapat mendefinisikan parameter di dalam tanda kurung ini.
- Pernyataan pertama dari sebuah fungsi dapat menjadi pernyataan opsional - string dokumentasi fungsi atau docstring.
- Blok kode di dalam setiap fungsi dimulai dengan titik dua `(:)` dan diindentasi. Pernyataan `return [expression]` keluar dari sebuah fungsi, secara opsional mengembalikan sebuah ekspresi kepada pemanggil.
- Pernyataan `return` tanpa argumen sama dengan `return None`.

Function def (3)

Syntax

```
1 def printme(str):  
2     print(str);  
3     return;  
4  
5 printme("Hari ini belajar Function");
```

Function Argument

Function arguments (argumen fungsi) merujuk pada **nilai-nilai yang dapat diterima oleh sebuah fungsi** saat fungsi tersebut dipanggil. Dalam Python, terdapat beberapa jenis argumen formal yang dapat digunakan dalam definisi fungsi, yaitu:

- Required arguments
- Keyword arguments
- Default arguments
- Variable-length arguments

Required Arguments

Argumen yang **harus disertakan** saat memanggil fungsi. Jumlah required arguments harus sesuai dengan yang didefinisikan dalam fungsi

python

```
def greet(name):  
    print("Hello,", name)  
  
greet("Alice") # Output: Hello, Alice
```

Keyword Arguments

Argumen yang disertakan dalam pemanggilan fungsi dengan

menyebutkan nama parameter dan nilainya

python

```
def greet(name, message):  
    print(message, name)
```

```
greet(message="Welcome", name="Bob") # Output: Welcom
```

Default Arguments

Argumen yang **memiliki nilai default** dan dapat diabaikan saat pemanggilan fungsi.

python

```
def greet(name, message="Hello"):
    print(message, name)
```

```
greet("Alice") # Output: Hello Alice
```

```
greet("Bob", "Welcome") # Output: Welcome Bob
```

Variabel-Length Arguments

Argumen yang memungkinkan fungsi menerima jumlah argumen yang tidak terbatas.

python

```
def sum_all(*args):  
    total = 0  
    for num in args:  
        total += num  
    return total  
  
print(sum_all(1, 2, 3)) # Output: 6
```

Keyword Return

- Fungsi yang tidak mengembalikan nilai biasanya disebut dengan **prosedur**
- Cara mengembalikan nilai adalah dengan memakai kata kunci return lalu diikuti nilai atau variabel yang akan dikembalikan.
- Pernyataan return **mengakhiri eksekusi fungsi** saat itu juga, sehingga pernyataan apapun setelah return tidak akan dieksekusi

Latihan

Algoritma:

1. Inisialisasi sebuah variabel `total` dengan nilai 0.
2. Iterasi melalui setiap elemen dalam list.
3. Untuk setiap elemen, periksa apakah itu genap.
4. Jika iya, tambahkan elemen tersebut ke variabel `total`.
5. Setelah iterasi selesai, kembalikan nilai dari variabel `total`.

Contoh Input:

```
python
```

[Copy code](#)

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

Contoh Output:

```
python
```

[Copy code](#)

```
30 # (2 + 4 + 6 + 8 + 10)
```

Latihan

Algoritma:

1. Buat sebuah set kosong untuk menyimpan angka-angka unik.
2. Iterasi melalui setiap elemen dalam list.
3. Untuk setiap elemen, periksa apakah itu sudah ada dalam set.
4. Jika belum, tambahkan elemen tersebut ke set.
5. Setelah iterasi selesai, buat sebuah list baru dari set tersebut dan balik urutannya.
6. Kembalikan list baru tersebut.

Contoh Input:

```
python
```

[Copy code](#)

```
[1, 2, 3, 3, 4, 5, 6, 5, 7, 8, 9, 10]
```

Contoh Output:

```
python
```

[Copy code](#)

```
[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
```

Scope Variable

- Scope variabel atau cakupan variabel merupakan suatu keadaan dimana pendeklarasian sebuah variabel di tentukan.
- Dalam scope variabel dikenal dua istilah yaitu **local** dan **global**.

Local Variables

- Variabel yang **dideklarasikan di dalam sebuah fungsi** dan hanya dapat diakses di dalam fungsi tersebut disebut variabel lokal.
- Variabel lokal hanya memiliki **ruang lingkup (scope) di dalam fungsi** tempat ia dideklarasikan.
- Variabel lokal digunakan ketika nilainya hanya diperlukan di dalam fungsi dan **tidak perlu diakses dari luar fungsi**.

Global Variables

- Variabel yang **dideklarasikan di luar fungsi** dan dapat diakses dari seluruh bagian program disebut variabel global.
- Variabel global **memiliki ruang lingkup di seluruh program**, artinya variabel ini dapat diakses dari dalam maupun luar fungsi.
- Variabel global digunakan ketika nilai yang disimpan diperlukan di berbagai bagian program dan **harus dapat diakses secara global**.

python

 Copy code

```
def my_function():  
    x = 10  
    print(x)  
  
x = 5  
my_function()  
print(x)
```

TERIMA KASIH 