



STIKOM PGRI BANYUWANGI

Dasar Pemrograman Backend



Pengenalan Python

Rudi Hartono, M.Kom

Fungsi (Function) dalam Bahasa Python

Berdasarkan siapa yang membuat, fungsi bisa dibedakan ke dalam 2 kelompok:

1. **Built-In Function**
2. **User Defined Function**

Built-In Function adalah sebutan untuk fungsi yang sudah ada secara bawaan dari dalam bahasa pemrograman. Sedangkan **User Defined Function** adalah fungsi yang kita (sebagai programmer) membuatnya sendiri.

❖ Membuat Fungsi dalam Bahasa Python

Untuk membuat fungsi, awali dengan keyword `def` lalu diikuti dengan nama fungsinya. Penulisan namaFunction sebenarnya boleh bebas, selama mengikuti aturan penulisan identifier yakni tidak boleh diawali angka dan tidak boleh mengandung spasi.

➤ Contoh

```
1 def nama_function():
2     # Isi function disini...
3     # Isi function disini...
4     return nilai
```

```
1 def sapa_lisa():
2     print("Hai Lisa");
3
4 def sapa_sari():
5     print("Morning, Sari");
6
7 def sapa_rudi():
8     print("Halo bro,..");
9
10 sapa_lisa()
11 sapa_sari()
12 sapa_rudi()
```

```
1 def hitung_luas_segitiga():
2     alas = 5
3     tinggi = 7
4     luas = (alas * tinggi) / 2
5     print("Luas segitiga adalah: ",luas)
6
7 hitung_luas_segitiga()
```

❖ Membuat Fungsi Dengan Parameter

Parameter adalah sebutan untuk nilai inputan fungsi pada saat fungsi itu di definisikan, sedangkan **argumen** adalah sebutan untuk nilai inputan fungsi pada saat fungsi itu dipanggil.

```
1 def nama_function(param1, param2):  
2     # Isi function disini...  
3     # Isi function disini...  
4     return nilai  
5  
6 nama_function(arg1, arg2)
```

➤ Contoh

```
1 def sapa_teman(nama):  
2     print("Hai", nama);  
3  
4 sapa_teman("Lisa")
```

```
1 def sapa_teman(nama1, nama2, nama3):  
2     print("Hai", nama1, nama2, nama3);  
3  
4 sapa_teman("Lisa", "Nova", "Putri")
```

```
1 def hitung_luas_segitiga(alas, tinggi):  
2     luas = (alas * tinggi) / 2  
3     print("Luas segitiga adalah: ", luas)  
4  
5 hitung_luas_segitiga(5, 7)  
6 hitung_luas_segitiga(2, 10)  
7 hitung_luas_segitiga(191, 357)
```

❖ Perintah Return Dalam Fungsi

Di dalam function, perintah **return** berfungsi mirip seperti **break** dalam perulangan. Jika ditemukan perintah **return**, pemrosesan function akan berhenti dan tidak akan mengeksekusi kode dibawahnya:

➤ Contoh

```
1  def harga_setelah_pajak(harga_dasar):  
2      return harga_dasar + (harga_dasar * 10/100)  
3  
4  harga_cilok = 5000  
5  harga_final_cilok = harga_setelah_pajak(harga_cilok)  
6  print("Harga cilok 1 porsi Rp.", harga_final_cilok)
```

❖ Default Parameter dalam Python

Default Parameter adalah istilah untuk parameter yang memiliki nilai awal, atau nilai default. Kadang fitur ini disebut juga sebagai **Default Argument**.

- Dengan **default parameter**, kita bisa memanggil fungsi **tambah()** hanya dengan 1 inputan angka, atau bahkan tidak perlu sama sekali.



```
1 def tambah(var1 = 5, var2 = 2):  
2     return var1 + var2  
3  
4 print( tambah() )  
5 print( tambah(1) )  
6 print( tambah(1,2) )  
7 print( tambah(5,4) )
```

- Function **pangkat()** memiliki 2 buah parameter: **angka** dan **pangkat**. Untuk parameter **pangkat** memiliki nilai default 2, sehingga jika pada saat pemanggilan tidak diisi, nilai 2 ini yang menjadi nilai untuk parameter.



```
1 def pangkat(angka, pangkat = 2):  
2     hasil = 1  
3     for i in range(0,pangkat):  
4         hasil = hasil * angka  
5     return hasil;  
6  
7 print( pangkat(3) )      # 9  
8 print( pangkat(5) )      # 25  
9 print( pangkat(10) )     # 100  
10 print( pangkat(3,3) )    # 27  
11 print( pangkat(5,4) )    # 625  
12 print( pangkat(6,6) )    # 46656
```


❖ Arbitrary Arguments (*args) dalam Python

Arbitrary arguments adalah istilah untuk menyebut jumlah argumen yang tidak bisa ditentukan atau berubah-ubah.

- Fungsi **sapa_teman()** di definisikan dengan 3 parameter, yakni nama1, nama2 dan nama3. Pada saat pemanggilan fungsi di baris 6, ketiga parameter ini harus diisi. Jika kita memanggil fungsi **sapa_teman()** dengan 4 parameter



```
1 def sapa_teman(nama1, nama2, nama3):  
2     print("Halo", nama1)  
3     print("Halo", nama2)  
4     print("Halo", nama3)  
5  
6 sapa_teman("Alex", "Nisa", "Sari", "Risa")
```



Error..

- Dengan menggunakan teknik *arbitrary arguments*, kita bisa membuat sebuah fungsi untuk menerima 3, 4 atau 100 argumen sekaligus. Caranya, buat satu parameter dengan tanda bintang di awal seperti berikut:



```
1 def sapa_teman(*args):  
2     print(args)  
3     print(type(args))  
4  
5 sapa_teman("Alex", "Nisa", "Sari", "Risa")
```

❖ Arbitrary Keyword Arguments (**kwargs) dalam Python

- ✓ **Arbitrary keyword arguments** adalah istilah untuk menyebut jumlah named argumen fungsi yang tidak bisa ditentukan atau berubah-ubah.
- ✓ Jika dalam arbitrary arguments (***args**) argumen fungsi ditulis langsung dengan nilai saja, maka dalam arbitrary keyword arguments (****kwargs**), argumen fungsi tersebut ditulis dalam bentuk pasangan nama dan value

- Isi dari fungsi **sambung_kata()** langsung menampilkan parameter **kwargs** dengan perintah **print(kwargs)**, serta memeriksa tipe data kwargs dengan perintah **print(type(kwargs))**.



```
1 def sambung_kata(**kwargs):
2     print(kwargs)
3     print(type(kwargs))
4
5 sambung_kata(a="Belajar", b="Python", c="di", d="STIKOM" )
```

- Di dalam fungsi **sambung_kata()** terdapat perulangan for untuk menampilkan semua nilai argumen yang dikirim. Akan tetapi karena **kata** adalah sebuah dictionary, maka yang tampil adalah **key** dari dictionary tersebut.



```
1 def sambung_kata(**kata):
2     for i in kata.values():
3         print(i)
4
5 sambung_kata(a="Belajar", b="Python", c="di", d="STIKOM" )
```

Perintah “INPUT” dalam Bahasa Python

- ❖ Salah satu fitur dalam Python adalah fungsi `input()`. Fungsi `input()` memungkinkan kita menerima masukan dari pengguna melalui *keyboard*.
- ❖ Fungsi ini sangat berguna dalam berbagai situasi, terutama saat kita perlu mengumpulkan data atau informasi dari pengguna.
- ❖ Fungsi `input()` Python bekerja dengan cara menerima satu parameter opsional (*string*) yang akan ditampilkan pada layar sebelum pengguna memasukkan *input*.

```
input([prompt])
```

- **`input()`** adalah nama fungsi. Dalam Python, kita memanggil fungsi dengan mengetikkan nama fungsinya diikuti oleh tanda kurung.
- **`[prompt]`** adalah parameter opsional yang dapat kamu masukkan dalam tanda kurung. Parameter biasanya berupa *string* yang akan ditampilkan sebelum memasukkan `input`. *String* bertindak sebagai petunjuk atau '**prompt**' bagi pengguna tentang jenis data yang perlu mereka masukkan.

```
1  # Meminta pengguna memasukkan suatu angka
2  angka = input("Masukkan suatu angka:")
3
4  print ("Anda telah memasukkan angka:", angka)
5
```


Tugas :

❖ Soal Menghitung Gaji Karyawan

Buatlah program dalam bahasa Python untuk menentukan gaji karyawan mingguan dengan ketentuan sebagai berikut:

- Golongan = A maka upah per jam 5000
- Golongan = B maka upah per jam 7000
- Golongan = C maka upah per jam 8000
- Golongan = D maka upah per jam 10000

Ketentuan tambahan:

- Jika jam kerja karyawan lebih dari 48 jam per minggu maka akan mendapat uang lembur dengan perhitungan uang lembur = $(\text{jam kerja} - 48) \times 4000$.
- Jika jam kerja kurang dari 48 jam maka pegawai tidak akan mendapat uang lembur.
- Perhitungan gaji pegawai adalah **upah + uang lembur**.
- Input berupa nama karyawan, golongan dan jam kerja.
- Outputnya adalah nama karyawan dan gaji yang diterima.

❖ Contoh Hasil

```
## Program Python Menghitung Gaji Karyawan ##  
=====
```

```
Nama Karyawan: Sari Lidya
```

```
Golongan: D
```

```
Jumlah jam kerja: 62
```

```
Sari Lidya menerima upah Rp. 676000 per minggu
```

The background features a light green field with faint, curved, wavy lines. Two large, dark green curved shapes, resembling stylized waves or hills, are positioned in the top-right and bottom-left corners. Small clusters of light green dots are located near the intersections of these dark green shapes with the main background.

Terima kasih.