Pertemuan 2

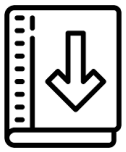
Fungsi/Sub Program Python

CAPAIAN PEMBELAJARAN

1. Memahami fungsi dan cara membuat fungsi
2. Menerapkan pada kasus di lapangan

KEBUTUHAN ALAT/BAHAN/SOFTWARE

1. Sistem Operasi Linux/Window 10
2. Python 3.x



DASAR TEORI

Fungsi

Fungsi adalah aspek terpenting dari sebuah aplikasi. Fungsi dapat didefinisikan sebagai blok terorganisir dari kode yang dapat digunakan kembali, yang dapat dipanggil kapan pun diperlukan.

Python memungkinkan untuk membagi program besar ke dalam blok bangunan dasar yang dikenal sebagai fungsi. Fungsi ini berisi kumpulan pernyataan pemrograman. Sebuah fungsi dapat dipanggil beberapa kali untuk memberikan kegunaan kembali dan modularitas ke program Python.

Fungsi membantu programmer untuk memecah program menjadi bagian-bagian yang lebih kecil serta mengatur kode dengan sangat efektif dan menghindari pengulangan kode.

Seiring berkembangnya program, fungsi membuat program lebih terorganisir.

Keuntungan Fungsi:

- Dengan menggunakan fungsi, dapat menghindari penulisan ulang logika/kode yang sama berulang kali dalam sebuah program.

- Dapat memanggil fungsi Python beberapa kali dalam suatu program dan di mana saja dalam suatu program.
- Dapat melacak program Python besar dengan mudah ketika dibagi menjadi beberapa fungsi.
- Dapat digunakan kembali adalah pencapaian utama fungsi Python.

Fungsi Bawaan

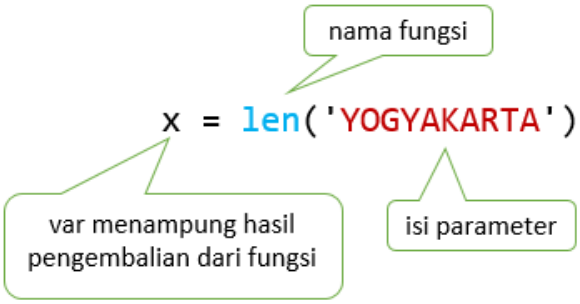
Interpreter Python memiliki sejumlah fungsi dan tipe bawaan di dalamnya yang selalu tersedia.

Fungsi Bawann Python				
abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	slice()
ascii()	divmod()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	

Contoh:

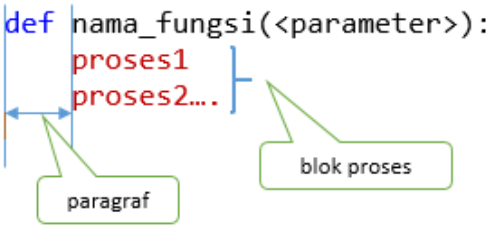
```
x = len('YOGYAKARTA')
print(x)
```

Fungsi len() akan menghitun panjang karakter “YOGYAKARTA” sehingga hasil keluarannya x = 10



Mebuat Fungsi Sendiri

Untuk membuat fungsi Python menggunakan perintah:



PRAKTIK

1. Buatlah nama fungsi sebaikan diawal huruf kecil, penamaan seperti pada penamaan variabel. contoh:

```
#======================
# /fungsi/pratik31.py
# fungsi tanpa parameter
# nim : …….
# nama : …….
#----------------------

def cetak_alamat():
    print("Jalan Wonosari KM 7 Yogyakarta")
cetak_alamat()
```

Penjelasan:

2. Melewati Objek yang Dapat Berubah (String) contoh:

```
#=======================
# /fungsi/pratik32.py
# fungsi mengubah string
# nim : …….
# nama : …….
# definisi function

def ubah_string (str):
    str = str + " makan rumput "
    print("Di dalam function :",str)


hewan = "Sapi"

#memanggil function
ubah_string(hewan)
print("Di luar function :",hewan)
```

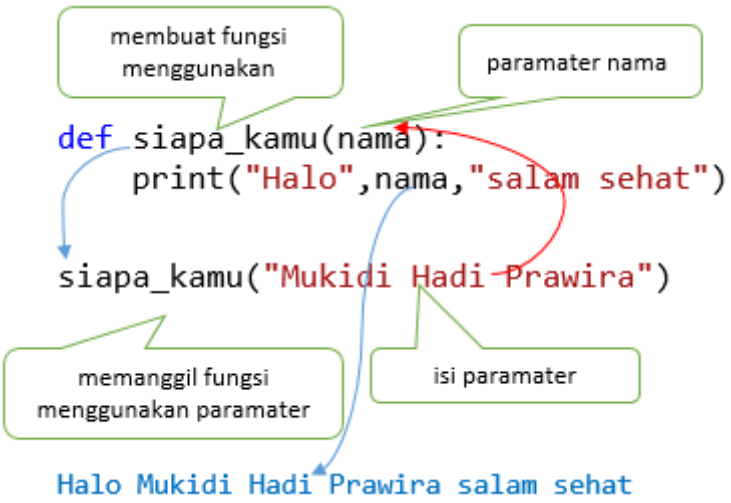
Hasil keluaran:

```
Di dalam function : Sapi makan rumput
Di luar function : Sapi
```

3. Fungsi menggunakan argumen/paramater, perhatikan contoh berikut:

```
#=======================
# /fungsi/pratik33.py
# fungsi mengubah string
# nim : …….
# nama : …….
# definisi function
def siapa_kamu(nama):
    print("Halo",nama,"salam sehat")
siapa_kamu("Mukidi Hadi Prawira")
```

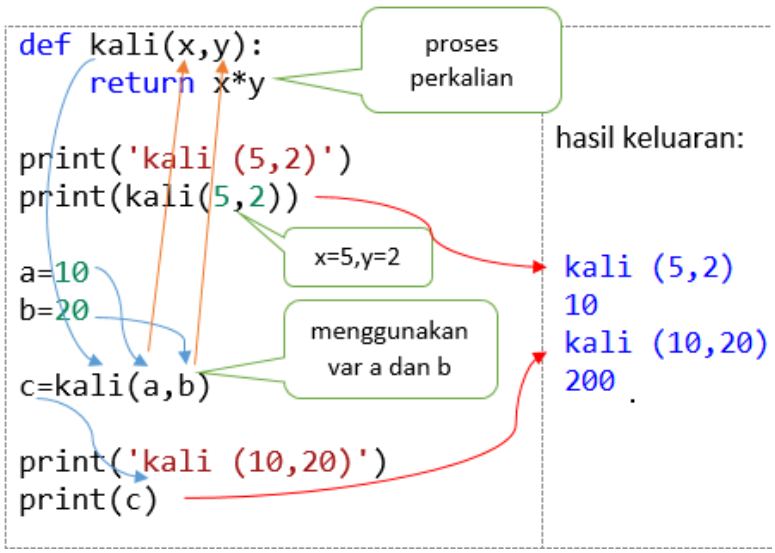
penjelasan:



4. Fungsi menggunakan argumen/parameter dan mengembalikan hasil, untuk mencoba buatlah seperti program berikut:

```
#======================
# /fungsi/pratik34.py
# fungsi mengubah string
# nim : …….
# nama : …….
# definisi function
def kali(x,y):
    return x*y
```

penjelasan:



5. Default Argument/parameter

Python memungkinkan untuk menginisialisasi argumen pada definisi fungsi. Jika nilai salah satu argumen tidak diberikan pada saat pemanggilan fungsi, maka argumen tersebut dapat diinisialisasi

dengan nilai yang diberikan dalam definisi meskipun argumen tidak ditentukan pada pemanggilan fungsi.

Bautlah seperti pada program berikut:

```
#====================================
# fungsi/pratik35.py
# fungsi menggunakan argumen default
#------------------------------------
# membuat fungsi
def tampilkanNama(nama,umur=22):
    print("Nama:",nama," dan umur:",umur)

# menggunakan argumen nama, umur=default
tampilkanNama(nama = "Ana")

# mengunakan dua nilai argumen
tampilkanNama("Ani",20)
```

Hasil keluaran:

```
Nama: Ana  dan umur: 22
Nama: Ani  dan umur: 20
>>>
```

6. Argumen Panjang Variabel (*args)

Dalam proyek besar, terkadang tidak mengetahui jumlah argumen yang harus dilewati terlebih dahulu. Dalam kasus ini, Python memberi fleksibilitas untuk memberikan nilai yang dipisahkan koma yang secara internal diperlakukan sebagai tupel pada pemanggilan fungsi. Dengan menggunakan argumen panjang variabel, dapat melewatkan sejumlah argumen.

Caranya dengan mendefinisikan argumen panjang variabel menggunakan *args (bintang) sebagai *<nama variabel >.

Perhatikan contoh berikut.

```
#====================================
# fungsi/praktik36.py
# fungsi menggunakan argumen
# variabel panjang
#------------------------------------
# membuat fungsi
def cetakHewan(*hewan):
    print("tipe argumen:",type(hewan)
    print("menambahkan argumen")
    for h in hewan:
        print(h)
# memanggil fungsi
cetakHewan("gajah","sapi","kerbau")
```

Hasil keluaran:

```
tipe argumen: <class 'tuple'>
menambahkan argumen
gajah
sapi
kerbau
```

7. Argumen kata kunci(kwargs)**

Python menyediakan fasilitas untuk meneruskan beberapa argumen kata kunci yang dapat direpresentasikan sebagai **kwargs. Ini mirip dengan *args tetapi menyimpan argumen dalam format kamus.

Perhatikan contoh berikut.

```
#====================================
# fungsi/praktik37.py
# fungsi menggunakan banyak argumen
#------------------------------------
def hewan(**kunci):
    print(kunci)

hewan(hewan="python")
hewan(hewan="sapi", pemakan="tumbuhan")
hewan(a=["python"])
```

Hasil keluaran:

```
{'hewan': 'python'}
{'hewan': 'sapi', 'pemakan': 'tumbuhan'}
{'a': ['python']}
>>>
```

8. buatlah pungsi untuk menhitung pembayaran dengan nilai pembayaran, dan diskon, dengan menggunakan masukan nilai balanja dan strok pembayaran yang sudah dikurangi diskon:

```
#==================================
# fungsi/praktik38.py
# menghitung diskon
#----------------------------------
def hitDiskon(pemb, disk):
    return pemb*disk/100

bayar  = int(input("Jumlah Bayar :"))
diskon = hitDiskon(bayar,10)
jBayar = bayar-diskon
print("Bayar      :",bayar)
print("Diskon 10% :",diskon)
print("Total      :",jBayar)
```

ujian hasil keluaranya:

9. Menggunakan Import
   Buatlah kumpulan fungsi tambah, kurang, kali, bagi, dan simpan ke fiel kalkulator.py seperti pada program berikut:

```
#-------------
# kalkulator
```

```
# nim : …….
# nama : …….
#--------------
def tambah(x,y):
    return x+y

def kurang(x,y):
    return x-y

def kali(x,y):
    return x*y

def bagi(x,y):
    return x/y
```

Buatlah program untuk mangil fungsi diatas menggunakan import:

```
 import kalkulator as k

a = int(input("Masukkan nilai a="))
b = int(input("Masukkan nilai b="))
print("a+b =",k.tambah(a,b))
print("a-b =",k.kurang(a,b))
print("a*b =",k.kali(a,b))
print("a/b =",k.bagi(a,b))
```

Eksekusi amati hasilnya

LATIHAN

1. Buatlah sub program fungsi untuk menghitung jumlah karakter, dengan mamasukan suatu teks.
2. Mengacu pada praktik nomor 9, tambahkan beberapa fungsi, pangkat, penambahan tipe data string.

TUGAS

REFERENSI

Python Function https://www.javatpoint.com/python-functions