PEMBELAJARAN DASAR PYTHON

BAB 8 : FUNCTION

1.1 Tujuan & Manfaat

1. Mahasiswa mengerti konsep fungsi pada bahasa pemrograman python

1.2 Ulasan Materi

FUNCTION

1. Pengertian Function

Function/Fungsi adalah kode program yang terisolasi dalam satu blok kode dan hanya berjalan ketika dipanggil. Programmer dapat meneruskan data, yang dikenal sebagai parameter, ke dalam suatu fungsi. Sebuah fungsi dapat mengembalikan data sebagai hasilnya. Fungsi memiliki beberapa atribut seperti nama fungsi, isi fungsi, parameter/argument, dan nilai balik. Pembuatan fungsi dilakukan dengan keyword def diikuti dengan nama fungsi, lalu di bawahnya ditulis body/isi fungsi.

Contoh:

```
def my_function():
  print("Hello from a function")
```

Output :

```
Hello from a function
```

Setelah di deklarasikan, fungsi bisa dipanggil berkali-kali.

Contoh:

```
def say_hello():
    print("Hello")
say_hello()
say_hello()
say_hello())
```

Output :

```
Hello
Hello
Hello
```

Pada contoh di atas, selain say_hello() sebenarnya ada satu buah fungsi lagi yang digunakan pada contoh, yaitu print(). Fungsi print() dideklarasikan dalam Python Standard Library (stdlib). Sewaktu program dijalankan fungsi-fungsi dalam stdlib otomatis ter-import dan bisa digunakan.

Contoh:

```
ef print_something():
    print("Hello")
    today = "Thursday"
    print(f"Happy {today}")
    for i in range(5):
        print(f"i: {i}")
print_something()
```

Output :

```
Hello
Happy Thursday
i: 0
i: 1
i: 2
i: 3
i: 4
```

2. Parameter Fungsi

Fungsi bisa memiliki parameter. Dengan adanya parameter, suatu nilai bisa di-sisipkan ke dalam fungsi secara dinamis saat pemanggilannya.

Parameter sendiri merupakan istilah untuk variabel yang menempel pada fungsi, yang mengharuskan kita untuk menyisipkan nilai pada parameter tersebut saat pemanggilan fungsi.

Contoh:

```
def calculate_circle_area(r):
    area = 3.14 * (r ** 2)
print("area of circle:", area)
```

Output :

```
area of circle: 1949764.1600000001
```

Penjelasan :

- Fungsi calculate_circle_area() dideklarasikan memiliki parameter bernama r.
- Notasi penulisan parameter fungsi ada diantara penulisan kurung () milik blok deklarasi fungsi.
- Tugas fungsi calculate_circle_area() adalah menghitung luas lingkaran dengan nilai jari-jari didapat dari parameter r. Nilai luas lingkaran kemudian di-print.
- Setelah blok deklarasi fungsi, ada statement pemanggilan fungsi calculate_circle_area(). Nilai numerik 788 digunakan sebagai argument parameter r pemanggilan fungsi tersebut.

3. Argumen

Informasi dapat diteruskan ke fungsi sebagai argumen. Argumen ditentukan etelah nama fungsi, di dalam tanda kurung. Anda dapat menambahkan argumen sebanyak yang Anda inginkan, cukup pisahkan dengan koma. Contoh berikut memiliki fungsi dengan satu argumen (fname). Saat fungsi dipanggil, memberikan nama belakang, yang digunakan di dalam fungsi untuk mencetak nama lengkap:

Contoh:

```
def my_function(fname):
  print(fname + " Refsnes")
my_function("Budi")
my_function("Hanif")
my_function("Rudi")
```

Output :

```
Budi Refsnes
Hanif Refsnes
Rudi Refsnes
```

Argumen sering disingkat menjadi args dalam dokumentasi Python

Parameter atau Argumen?

Istilah parameter dan argumen dapat digunakan untuk hal yang sama: informasi yang diteruskan ke suatu fungsi.

Dari perspektif fungsi:

Parameter adalah variabel yang terdaftar di dalam tanda kurung dalam definisi fungsi. Argumen adalah nilai yang dikirim ke fungsi saat dipanggil

1. **Jumlah Argumen**

   Secara default, suatu fungsi harus dipanggil dengan jumlah argumen yang benar. Artinya jika fungsi Anda mengharapkan 2 argumen, Anda harus memanggil fungsi dengan 2 argumen, tidak lebih, dan tidak kurang.

Contoh:

```
def my_function(fname, lname):
  print(fname + " " + lname)
my_function("Budi", "Refsnes")
```

Output :

```
Budi Refsnes
```

Fungsi ini mengharapkan 2 argumen, dan mendapat 2 argumen

Jika Programmer mencoba memanggil fungsi dengan 1 atau 3 argumen, programmer akan mendapatkan kesalahan.

Contoh:

```
def my_function(fname, lname):
  print(fname + " " + lname)
my_function("Budi")
```

Output :

```
Traceback (most recent call last):
  File "demo_function_args_error.py", line 4, in <module>
    my_function("Budi")
TypeError: my_function() missing 1 required positional
argument: 'lname'
```

Fungsi ini mengharapkan 2 argumen, tetapi hanya mendapatkan 1

2. Argumen Berubah-Ubah, *args

Jika Programmer tidak tahu berapa banyak argumen yang akan diteruskan ke fungsi, tambahkan * sebelum nama parameter dalam definisi fungsi. Dengan cara ini fungsi akan menerima tupel argumen, dan dapat mengakses item yang sesuai

Contoh:

```
def my_function(*kids):
  print("Anak yang paling muda adalah" + kids[2])
my_function("Budi", "Hanif", "Rudi")
```

Output :

```
Anak yang paling muda adalah Hanif
```

Argumen berubah-ubah sering disingkat menjadi args* dalam dokumentasi Python

3. Argumen Kata Kunci

Programmer juga dapat mengirim argumen dengan sintaks kunci = nilai. Dengan cara ini urutan argumen tidak menjadi masalah.

Contoh:

```
def my_function(child3, child2, child1):
  print("Anak yang paling muda adalah " + child3)
my_function(child1 = "Budi", child2 = "Hanif", child3 =
"Rudi")
```

Output :

```
Anak yang paling muda adalah Rudi
```

4. Passing List Argumen

Anda dapat mengirim tipe data argumen apa pun ke suatu fungsi (string, angka, daftar, kamus, dll.), dan itu akan diperlakukan sebagai tipe data yang sama di dalam fungsi.

Contoh:

```
def my_function(food):
  for x in food:
    print(x)
fruits = ["apel", "jeruk", "pisang"]
my_function(fruits)
```

Output :

```
apel
jeruk
pisang
```

5. Mengembalikan Values

Untuk membuat suatu fungsi mengembalikan nilai, gunakan *statement* return

Contoh:

```
def my_function(x):
  return 5 * x
print(my_function(3))
print(my_function(5))
print(my_function(9)
```

Output :

```
15

25

45
```

1.3 Praktikum

Contoh : Membuat fungsi dan memanggilnya

Pada contoh ini kita akan membuat sebuah fungsi untuk kemudian ditampilkan dengan keyword def diikuti dengan nama fungsi. Berikut ini adalah langkah-langkahnya:

1. Tuliskan fungsi dengan nama buah

```
def(buah)
```

2. Kemudian ketikkan perintah print untuk memanggil fungsi tersebut

```
print(“Jeruk)
```

1.4 Percobaan 1

Sekarang kerjakan percobaan 1 dengan mengikuti langkah-langkah dibawah ini !

1. Buatlah fungsi dengan nama hitug_luas_segitiga
2. Tambahkan variabel alas dengan nilai 5
3. Tambahkan variabel tinggi dengan nilai 7
4. Kemudian tambahkan variabel luas dengan (alas x tinggi) : 2
5. Tampilkan hasil dari Luas Segitiga dengan fungsi print
6. Jalankan kode program percobaan 1 diatas dengan menekan tombol Check Code Validity