Nama : Ghina Khoerunnisa
Program : Data Scientist Intern

Function, Basic Module, and Package

- **Function,** merupakan blok kode yang dapat digunakan kembali sehingga program kita lebih terorganisir.. Sintaks dan contoh penggunaan function adalah sebagai berikut:

```
def function_name(parameters):
    """docstring"""
    statement
```

```
def hitung_jumlah_permen(kantong, permen_baru):
    """
        Function untuk menghitung jumlah permen

        parameter
        ---------
            kantong: jumlah permen yang ada di kantong
            permen_baru: jumlah permen baru

        return
        ---------
            total jumlah permen di kantong dengan permen_baru
    """
    return kantong+permen_baru
```

Untuk memanggil fungsi tersebut dapat dilakukan dengan menulis nama fungsinya serta parameter yang dibutuhkan. Misalnya hitung_jumlah_permen(10,3), maka keluarannya adalah 10+3 = 13

  - **Pass by reference vs pass by value**
    Pass by reference artinya adalah ketika kita mengubah apa yang merujuk oleh suatu variabel maka variable awal itu tadi juga ikut berubah karena memiliki alamat memori yang sama. Sedangkan pass by value ini merupakan passing dimana kita mengcopy value dari suatu variabel ke variabel lain maka kedua variabel memiliki alamat yang berbeda. Ketika kita mengubah salah satu variabel maka variabel yang lain itu tidak ikut berubah.
  - **Function Arguments**
    Required Arguments, argumen yang diteruskan ke suatu fungsi dengan posisi yang benar serta jumlah argumen pada pemanggilan harus sama dengan yang ada di fungsinya. Contohnya fungsinya sama pada gambar 1 dan pemanggilannya sama seperti pada contoh sebelumnya.
    Keyword Arguments, terdapat kata kunci pada saat pemanggilan fungsinya contoh hitung_jumlah_permen(kantong=10, permen_baru=3).
    Default Arguments, argumen yang mengasumsi nilai default jika nilai tidak disediakan dalam pemanggilan fungsi untuk argumen tersebut. Contohnya adalah,

```
def default_arg_function(nama, status=0):
    print(f'nama: {nama}\nstatus: {status}')

default_arg_function(nama='Ghina Khoerunnisa')
```

```
nama: Ghina Khoerunnisa
status: 0
```

    Variable-length Arguments, memproses fungsi untuk menerima lebih banyak argumen daripada yang ditentukan pada saat mendefinisikan fungsi. Contohnya adalah,

```
def printinfo(nama, *args, **kwargs):
    print(f'argumen nama: {nama}')
    print('variable length arg(s):')
    print(args)
    print('keyword arg(s):')
    print(kwargs)

printinfo('Ghina', 10, 20, '-', 30, permen=3, pena=4)
```

```
argumen nama: Ghina
variable length arg(s):
(10, 20, '-', 30)
keyword arg(s):
{'permen': 3, 'pena': 4}
```

- **Anonymous Function,** sesuai dengan namanya function ini tidak dideklarasikan dengan pendefinisian function biasanya (tidak menggunakan keyword def). Contohnya adalah,

jumlah = lambda var_1, var_2: print(var_1 + var_2) kemudian dipanggil fungsinya jumlah(11,22) maka outputnya adalah 33. Ini sama saja jika kita mendeklarasikan fungsinya dengan def jumlah(var_1, var_2): print(var_1 + var_2).

- **Scope Variable,** Variabel yang didefinisikan di dalam fungsi adalah local scope, dan variabel yang ditentukan di luar adalah global scope. Variabel local hanya dapat diakses di dalam fungsi yang dideklarasikan.
- **Module,** Isi modul diakses dengan menggunakan import statement. Untuk membuat module yang ditulis dengan python adalah membuat file baru dengan ekstensi .py. Contoh pemanggilan module (asumsi module dan program berada di folder yang sama) adalah [import nama_module] kemudian setiap variabel, fungsi, dan class yang ada di module dapat dipanggil di program tersebut salah satunya [nama_modul.nama_fungsi]. Jika file module berada di luar folder program yang memanggil modul maka kita perlu meng-append direktorinya ke PYTHONPATH. Kemudian jika ingin memanggil bagian module di program tanpa menuliskan modulenya dapat menggunakan [from nama_modul import nama_variabel, nama_fungsi, nama_class] atau jika ingin semuanya [from nama_modul import *]. Hanya saja penggunaan ini tidak disarankan karena khawatir akan terjadi overwrite. Selanjutnya kita dapat membuat alias terhadap module yang diimport dengan cara [import nama_module as nama_alias].
- **Dir() Function,** fungsi built-in dir() mengembalikan daftar nama yang ditentukan dalam namespace. Semua variabel, class, fungsi yang kita definisikan akan masuk ke namespace. Jika menjalanka dir() dengan argumen seperti dir(nama_module), maka dir() akan menampilkan nama yang ditentukan di dalam module.
- **Reload Module**
  Module hanya dimuat sekali per sesi interpreter sehingga jika terdapat perubahan di module maka perlu di reload agar module berisi yang terbaru. Untuk me-reload module dapat dilakukan dengan cara [import importlib; importlib.reload(nama_module)].
- **Packages,** memungkinkan penataan hierarki nama modul menggunakan notasi titik. Dengan cara yang sama seperti modul membantu menghindari tabrakan antara nama variabel global, packages membantu menghindari tabrakan antara nama modul.
- **PIP,** PIP adalah package manager untuk Python. PIP merupakan alat yang memungkinkan kita menginstal dan mengelola library dan dependensi tambahan yang tidak didistribusikan sebagai bagian dari library standar.
  Beberapa perintah yang ada di PIP adalah,
  - pip --version (untuk melihat versi pip)
  - pip help (untuk melihat perintah-perintah apa saja yang ada di pip)
  - pip install [nama_package] (untuk menginstall package)
  - pip freeze > requirements.txt (untuk mengeluarkan package yang diinstal dalam format requirements)
  - Pip install -r requirement.txt (untuk mereplikasi environment di sistem lain)