Function

Fungsi adalah blok kode terorganisir dan dapat digunakan kembali yang digunakan untuk melakukan sebuah tindakan/action.

Pembuatan Function

Berikut adalah aturan sederhana untuk mendefinisikan fungsi dengan Python.

- Fungsi blok dimulai dengan `def` kata kunci diikuti oleh nama fungsi dan tanda kurung (`()`).
- Setiap parameter masukan atau argumen harus ditempatkan di dalam tanda kurung ini. Anda juga dapat menentukan parameter di dalam tanda kurung ini.
- Blok kode dalam setiap fungsi dimulai dengan titik dua (:) dan indentasi.
- Pernyataan kembali [ekspresi] keluar dari sebuah fungsi, secara opsional menyampaikan kembali ekspresi ke pemanggil. Pernyataan pengembalian tanpa argumen sama dengan `return None`.

In [5]:
```
# Function tanpa parameter
def my_function():
    print("Hello from a function")


# Memanggil fungsi
my_function()
```

```
Hello from a function
```

In [6]:
```
# Function dengan parameter
def my_function(fname):
    print(fname + " Refsnes")


my_function("Emil")
my_function("Tobias")
my_function("Linus")
```

```
Emil Refsnes
Tobias Refsnes
Linus Refsnes
```

In [7]:
```
# Function dengan lebih dari satu parameter
def my_function(fname, lname):
    print(fname + " " + lname)


my_function("Emil", "Refsnes")
```

```
Emil Refsnes
```

In [8]:
```
# Jika jumlah parameter tidak diketahui,
# Tambahkan * sebelum nama parameter
def my_function(*kids):
    print("The youngest child is " + kids[2])


my_function("Emil", "Tobias", "Linus")
```

```
The youngest child is Linus
```

In [9]:
```
# Function juga memungkinkan paramter dengan key = value
# Urutan paramater tidak berpengaruh

def my_function(child3, child2, child1):
    print("The youngest child is " + child3)


my_function(child1="Emil", child2="Tobias", child3="Linus")
```

```
The youngest child is Linus
```

In [10]:
```
# Jika jumlah keyword parameter tidak diketahui,
# Tambahkan double bintang (**) sebelum nama parameter
def my_function(**kid):
    print("His last name is " + kid["lname"])


my_function(fname="Tobias", lname="Refsnes")
```

```
His last name is Refsnes
```

In [11]:
```
# Jika pemanggilan function tanpa parameter yang dikirim,
# Function akan menggunakan nilai default

def my_function(country="Norway"):
    print("I am from " + country)


my_function("Sweden")
my_function("India")
my_function()
my_function("Brazil")
```

```
I am from Sweden
I am from India
I am from Norway
I am from Brazil
```

In [12]:
```
# You can send any data types of argument to a function
# (string, number, list, dictionary etc.),
# and it will be treated as the same data type inside the function

def my_function(food):
    for x in food:
        print(x)


fruits = ["apple", "banana", "cherry"]

my_function(fruits)
```

```
apple
banana
cherry
```

In [13]:
```
# To let a function return a value, use the return statement:

def my_function(x):
    return 5 * x


print(my_function(3))
print(my_function(5))
print(my_function(9))
```

```
15
25
45
```

In [14]:
```
# function definitions cannot be empty,
# but if you for some reason have a function definition with no content,
# put in the pass statement to avoid getting an error.

def myfunction():
    pass
```

In [15]:
```
# subprograms
# function - it returns something
def getName():
    uName = input("Please enter your name: ")
    return uName

# subroutine - it doesn't need to return anything
# passing in the 'userName' parameter, as it will be used in this subroutine


def printMsg(userName):
    print("Hello", userName)


userName = getName()
# calling the function and adding an argument in the brackets
printMsg(userName)
```

```
Hello Tobias
```

In [16]:
```
# Create a global variable.
my_value = 10


# The show_value function prints
# the value of the global variable.
def show_value():
    print(my_value)


# Call the show_value function.
show_value()
```

```
10
```

In [17]:
```
# To assign values to global variables you have to use the keyword global

# Create a global variable.
number = 0


def main():
    global number
    number = int(input('Enter a number: '))
    show_number()


def show_number():
    print('The number you entered is', number)


# Call the main function.
main()
```

```
The number you entered is 17
```

In [18]:
```
# This program demonstrates what happens when you
# change the value of a parameter.

def main():
    value = 99
    print('The value is', value)
    change_me(value)
    print('Back in main the value is', value)


def change_me(arg):
    print('I am changing the value.')
    arg = 0
    print('Now the value is', arg)


# Call the main  function.
main()
```

```
The value is 99
I am changing the value.
Now the value is 0
Back in main the value is 99
```