Understanding

1. Functions in Python are a collection of commands or lines of code that are grouped into one unit and can then be called or used many times.
2. A function can accept parameters, can return a value, and can be called many times independently.

Profit

1. Divide the program code into small parts with their respective tasks.
2. Make program code more "reusable" and more structured.

Syntax

```
def  function_name ():
    Program logic
```

How to call a function by writing function_name followed by open-close brackets () and parameters if any

```
function_name ()
```

Create a function named "helloPython" that prints "Welcome in Python Language"

In[7]:
```
def  helloPython ():
    print (  "Welcome to Python Language" )

helloPython ()
helloPython ()
helloPython ()
```

```
Welcome to Python Language
Welcome to Python Language
Welcome to Python Language
```

Functions with parameters

A function can accept parameters or arguments which are values/variables that are thrown into the function for further processing.

Syntax

```
def  function_name ( param ):
    Program logic
```

How to call a function by writing function_name followed by open-close brackets () and parameters

```
function_name ( param )
```

Create a function named "fullName" which has parameters "firstname" and "lastname" used to print "firstname" and "lastname"

In[1]:
```
def  fullName ( firstName = input ( 'enter your first name: ' ), lastName = input
    print ( f "Your full name is { firstName } { lastName } " )

fullName ()
```

```
Enter your first name: Rizki Panca
Enter your last name: Pamungkas
Your full name is Rizki Panca Pamungkas
```

Function parameters are allowed to use more than 1, where some of these parameters are mandatory and some are not required

```
def  function_name ( param_1 ,  param_2 ,  param_3 , .... ):
    Logic  Program
```

How to call a function by writing function_name followed by open-close brackets () and parameters

```
function_name ( param_1 ,  param_2 ,  param_3 )
```

Create a function named maxValue which has parameters "val_1", "val_2" and "val_3" which aims to find the largest value of the 3 values

In[3]:
```
def  maxValue ( val11 = input ( 'enter first number: ' ),  val12 = input ( ' input
    if  val11  and  val13  <  val12 :
        print ( f 'the largest number is { val12 } ' )
    elif  val11  and  val12  <  val13 :
        print ( f'the largest number is { val13 } ' )
    else :
        print ( f 'the largest number is { val11 } ' )

maxValue ()
```

```
first number input: 16
Enter the second number: 20
Enter third number: 36
the biggest number is 36
```

Optional parameters are used in functions by providing default values, meaning that the parameter values have been given beforehand without being called

```
def  function_name ( param_1 ,  param_2 ,  param_3  =  'Value' ):
    Program Logic
```

Create a function "countCircleArea" with 2 parameters namely "phi" and "diameter" where the parameter phi has a default value of 3.14

In [6]:
```
def  countCircleArea  ( phi = 3.14 , diameter =  float ( input ( 'enter the diamet
  circle  =  phi * diameter
```

```
  print ( f 'area of the circle is: { area of the circle } ' )

countCircleArea ()
```

```
Enter the diameter of the circle to be calculated: 60.15
the area of the circle is: 188,871
```

Functions with 2 optional Parameters are also possible in Python.

```
def  info ( temperature ,  area = 'Sukabumi' ,  unit  =  'Celsius' ):
    print ( f "Current temperature in { area } : { temperature } { unit
} " )
```

As for How to summon it

```
info ( 30 )
```

In[7]:
```
#Try this function in the code area here
def  info ( temperature ,  area = 'Sukabumi' ,  unit  =  'Celsius' ):
    print ( f "Current temperature in { area } : { temperature } { unit } " )

info ( 30 )
```

```
Current temperature in Sukabumi: 30 Celsius
```

Functions with Return Values

That is a function where the end of the program is the return value or return value. This means that the value in the function can be stored again in another variable to be used for further operations.

Create a function with a return value that is used to check a number including negative numbers, neutral numbers (0) and positive numbers

In [12]:
```
def  FunctionReturn ( val ):
    if  val  ==  0 :
        return  "neutral"
    elif  val  %  2  ==  0 :
        return  "positive"
    else :
         return  "negative"
valinput  =  float ( input ( 'input the value = ' ))
print ( FunctionReturn ( valinput ))
```

```
input the value = 8
positive
```

Exercise

1. Create a function to add up the total value of the list
2. Create a function to find the largest value from a set of lists
3. Create a function to add 2 lists

In [19]:
```
#Problem 1
def  TotalValueList ( list ):
    sum  =  sum ( list )
    return  sum
```

```
yahist  =  [ 10 , 20 , 30 , 40 , 50 , 60 , 70 , 80 , 90 , 100 ]

print ( 'TotalValuesList' ,  TotalValuesList ( yahalist ))
```

```
TotalValueLis 550
```

In [21]:
```
#Problem 2
def  TotalValueList ( list ):
    maxval  =  max ( list )
    return  maxval

yahalist  =  [ 10 , 20 , 30 , 40 , 50 , 60 , 70 , 80 , 90 , 100 ]
TotalValueList ( yahalist )
```

Out[21]:
```
100
```

In [24]:
```
#Problem 3
def  TotalValuesList ( list1 , list2 ):
    sum  =  sum ( list1 )  +  sum ( list2 )
    return  sum

yahist  =  [ 10 , 20 , 30 , 40 , 50 , 60 , 70 , 80 , 90 , 100 ]
yahalist2  =  [ 110 , 120 , 130 , 140 , 150 , 160 , 170 , 180 yes , 190 , 200 ]
TotalValueList ( halist yahist2 )
```

Out[24]:
```
2100
```

In [ ]: