Functions in Python

Presented by:-

Er. Honey Pasricha

Eklavya Academy
www.ekaim.in

www.ekaim.in
Fb Page
@eklayaaiim

Functions

- A function is a group of statements that exist within a program for the purpose of performing a specific task.

Benefits of Modularizing of Function

- Simpler
- Code Reuse
- Better Testing
- Faster Development
- Easier Facilitation of Team Work

Overview

- Introduction to repetition Structure
- The while loop: A Condition-Controlled Loop
- For Loop: A Count-Controlled Loop

Rules to write the function

- You cannot use one of Python's key words as a function name. (See Table 1-2 for a list of the key words.)
- A function name cannot contain spaces.
- The first character must be one of the letters a through z, A through Z, or an underscore character (_).
- After the first character you may use the letters a through z or A through Z, the digits 0 through 9, or underscores.
- Uppercase and lowercase characters are distinct.

Define the function, Calling the Function

To create a function you write its *definition*. Here is the general format of a function definition in Python:

```
def function_name():
    statement
    statement
    etc.
```

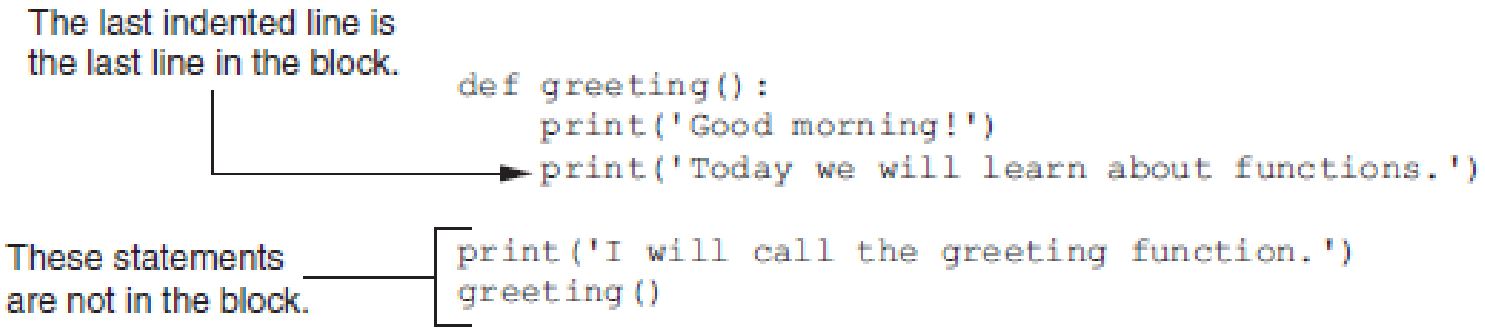
Calling a Function

A function definition specifies what a function does, but it does not cause the function to execute. To execute a function, you must *call* it. This is how we would call the message function:

```
message()
```

Indentation of function

Figure 3-7 All of the statements in a block are indented



3.5 Passing Arguments to Functions

CONCEPT: **An argument is any piece of data that is passed into a function when the function is called. A parameter is a variable that receives an argument that is passed into a function.**

Parameter Variable Scope

Earlier in this chapter, you learned that a variable's scope is the part of the program in which the variable may be accessed. A variable is visible only to statements inside the variable's scope. A parameter variable's scope is the function in which the parameter is used. All of the statements inside the function can access the parameter variable, but no statement outside the function can access it.

Local and Global Variable

```
def fun1():
    a=4       #local variable
    b=5       #local variable
    c=a+b
    print(c)
def fun2():
    z=a+b    # cannot access the a and b as both are local of function fun1()
    print(c)
```

Local and Global Variable

```
a=4       #global variable
b=5       #global variable

def fun1():
    c=a+b # can access the a and b variables as both are global
    print(c)
def fun2():
    z=a*b    # can access the a and b variables as both are global
    print(c)
```

Keyword Argument

parameter_name=value

In this format, *parameter_name* is the name of a parameter variable and *value* is the value being passed to that parameter. An argument that is written in accordance with this syntax is known as a *keyword argument*.

Program 3-11 (keyword_args.py)

```
1  # This program demonstrates keyword arguments.
2
3  def main():
4      # Show the amount of simple interest, using 0.01 as
5      # interest rate per period, 10 as the number of periods,
       # and $10,000 as the principal.
       show_interest(rate=0.01, periods=10, principal=10000.0)

# The show_interest function displays the amount of
# simple interest for a given principal, interest rate
# per period, and number of periods.

def show_interest(principal, rate, periods):
    interest = principal * rate * periods
    print('The simple interest will be $', \
          format(interest, ',.2f'), \
          sep='')
```