WORKING WITH FUNCTIONS USING PYTHON

MODULE 1/4

Mrs. SUJATA PRADHAN
PGT(SS)
AECS,ANUPURAM

Definition OF Function

A **program** is a set of statements that takes some input, does specific computations based on given input and produces desired output. A very Large program with a huge single list of instructions increases complexity.So Python allows us to divide a large program into some small independent units or blocks known as **functions.** Decomposing a complex problem into simpler one using functions improves clarity of the code.

Functions are the most important segments or subprograms of an application used to perform specific tasks. A python program can have one or more functions.

The advantages of using functions

- Reduce duplication of code.
- Induce reusability of code.

By using functions, we can avoid rewriting same logic/code again and again, Thus function reduces program size.

We can call python functions any number of times from any part of the program.So function induces reusability in a program.

Types of Functions:

Basically there are three types of functions used in python program:

- Built in functions (python library functions)
  - These are predefined functions and are always available in python library.
- Functions defined within modules
  - These are also predefined functions available in different modules.
- User defined functions
  - These are defined by programmer.

1.Built- in Functions

- These functions are already built in the library of python and can be accessed by programmer easily.
- These are always available and for using them, we don't have to import any module (file).
- Python has a small set of built-in functions like abs(), max(), min(), len(), range(),round(),bool() chr(), float(), int(),long(),str( ),type( ),id( ) etc.

Example:

max(x, y, z) returns the largest of its 3 arguments.

```
>>>max(80, -70, 100)
100
```

2.Functions defined in modules

When we want to use **module based functions** in our program, we need to **import** the corresponding module of that particular function.

The functions available in math module are:

ceil(), floor(), fabs(), exp(), log(), pow(), sqrt() cos(), sin()etc.

Example:

ceil(x) returns the smallest integer not less than x

fabs(x) returns the absolute value of x,where x is a numeric value.

Example:

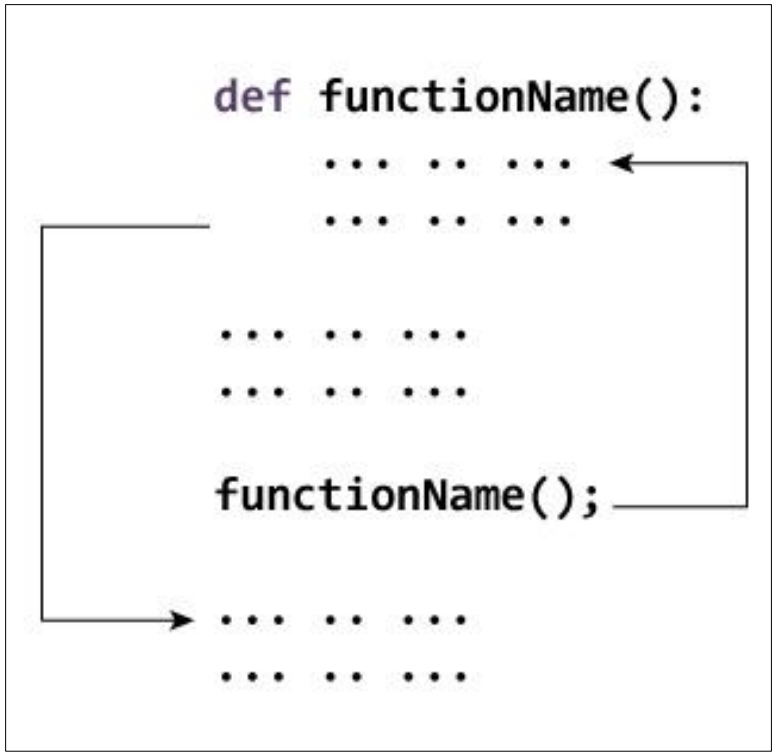
To work with the functions of **math** module, we must import math module in our program.

sqrt() returns the square root of a number

```
>>>import math
>>>math.sqrt(49)
 7.0
```

User defined function

In Python, programmers can also develop their own function(s). They are known as **user defined functions**.



Syntax of function

def function-name(parameters) :
#block of statement(s)

Example:

```
def hello_world():     #called function
  print("hello world")

hello_world()          #calling function
```

Output:

hello world

Defining functions in python

```
def functionName( list of parameters ):
    "_docstring"
    function_block
    return [expression]
Top level statements
```

```
def userfunction (arg1, arg2, arg3 ...):
        program statement1
        program statement2
        program statement3
        ....
        return
userfunction(arg1,arg2,arg3)
```

Top level statements

- In python program, generally all **python definitions** are given at the top followed by statements which are not part of any functions These statements are not indented at all.
- The non indented statements written after all the function definitions are often called **top level statements** .

Function definition with example

- Keyword **def** marks the start of function header.
- A **function name** to uniquely identify it. Name of the Function follows the same rule of naming the identifier.
- **Parameters** (arguments) through which we pass values to a function are optional.
- A **colon (:)** is used to mark the end of function header.
- The string after the function header is called the **docstring** . It is briefly used to explain what a function does. **comments** are ignored by python interpreter but **docstrings** can be viewed when the program is running.
- One or more python statements form function body.All the statements of the block should have **same indentation level**.
- A **return statement is used** to return value(s) from the function but it is always optional.

```
#python function to calculate the sum o
    f two variables

#defining the function
def sum(a,b):
   '"takes a and b and return the
     sum'"
    return a+b;

#taking values from the user
a = int(input("Enter a: "))
b = int(input("Enter b: "))

#printing the sum of a and b
print("Sum = ",sum(a,b))
```

Output:

```
Enter a: 10
Enter b: 20
Sum = 30
```

HOW A FUNCTION WORKS

- Execution always begins from the first statement of the program.
- A python program may contain several funtion definitions.
- If any function definition is found,python executes only function header for the correctness of it and skips all lines of function body(block).
- When python sequentially reaches top level statement's function call, python transfers control to the function header and then execution of function body takes place.
- Finally function execution ends with a return statement if any or the last statement of function body.

FLOW OF EXECUTION IN A FUNCTION CALL

- **Flow of execution** refers to the order in which statements are executed.
- A function body is executed in **execution frame**.
- Whenever a **function call** statement is executed, **execution frame** for the called function is created and the **control** is transferred to invoke the **called function**.
- Within the function's execution frame,the body of the function gets executed and after the last statement of the function the **control** returns to the statement with/without any value(s) to the function from where it is called(calling function).

Function Parameters:

The values being passed through a **function call** statement are called **arguments or actual parameters**.The values received in the **function definition** are called **parameters or formal parameters**.

A function has two types of parameters:

- **Formal Parameter(parameters):** Formal parameters are written in the function prototype(function definition). **Formal parameters** are local variables which are assigned values from the arguments when the function is called.
- **Actual Parameter(arguments):** When a function is called, the values that are passed are called **actual parameters.** At the time of the call, each actual parameter is assigned to the corresponding formal parameter in the function definition.

Note:

1. Function which is called by another Function is called **Called Function**. The **called function** contains the **definition of the function and formal parameters** are associated with them.
2. The Function which calls another Function is called **Calling Function** and **actual paramaters** are associated with them.
3. In python, a function must be defined **before** the function calling otherwise python interpreter gives an **error**.

Lambda function

Python lambda function doesn't have any return statement.It has only a single expression which is always returned by default. The Python lambda function is anonymous as it is a function without a **def keyword** and **name**. To create a Python lambda function, we have to use the **lambda keyword**.

The basic syntax of python lambda is

Lambda arguments : expression

The Python lambda function accepts any number of arguments but use only one expression.

For instance, lambda a, b: a + b. Here, a and b are the arguments accepted by the lambda function. a + b is the expression.

Example:

```
add = lambda x, y : x + y
print(add(10, 20))

print("\nResult from a Function")
def add_func(x, y):
    return x + y

print(add_func(10, 20))
```

Both lambda function and regular function returns the same result. However, the **regular function** needs a **def** keyword, **function name**, and a **return value**. Whereas, **lambda function** does not need any of them. By default, it returns the expression result.

Summary of Module 1

- **What is a function**
- **How a function works**
- **Syntax of user defined function**
- **Calling function and called function**
- **Formal parameter and actual parameter**

SOLVED QUESTIONS

1. Differentiate between round () and floor() functions with suitable examples.

Ans. The function round() is used to convert a fractional number into whole as the nearest next whereas the function floor() is used to convert the nearest lower whole number. e.g.,

round (4.1) = 5 and floor (6.9) = 6

2. Name the Python Library modules which need to be imported to invoke the following functions:

(i) sin( ) (ii) randint ( )

1

Ans.

(i) math (ii) random

3. What will be the output of the following code?

```
a=1
def f():
    a=10
print(a)
```

OUTPUT:

1

4. What is a lambda function? Explain with an example.

A lambda function is a small anonymous function which can take any number of arguments, but can only have one expression.

E.g.

```
x = lambda a, b : a * b
print(x(5, 6))
```

OUTPUT:

30

WORKSHEET 1

1. What is the significance of having function in a program?
2. Why are docstrings used? How are they different from comments?
3. How do we define a function?
4. What is lambda function?
5. What is the difference between formal parameter and actual parameter?
6. Write the syntax of function definition and explain with an example.
7. Write a program to generate fibonacci series with a function.
8. What is the importance of void function?
9. Differentiate between built in function and functions defined in module.
10. Rewrite the following Python function after removing all the syntactical errors (if any)

```
def checkval:
    x = raw_input("Enter a number")
    if x % 2 = 0 :
        print x,"is even"
    else if x<0 :
        print x,"should be positive"
    else ;
        print x,"is odd"
```

11. Write a python program using function to find the largest element in a list.

```
def largest(L,n) :
    max = L[0]
    for i in range(1, n) :
        if L[i] > max :
            max =L[i]
    return max
M = [10, 24, 45, 90, 98]
n = len(M)
max= largest(M, n)
print ("Largest in the given List is", max)
```

Output

Largest in the give n List is 98

THANK YOU