PYTHON

23 Session

Step by Step Coding

All Code in Git

Function

Function

If a group of statements is repeatedly required then it is not recommended to write these statements every time separately. We have to define these statements as a single unit and we can call that unit any number of times based on our requirement without rewriting. This unit is nothing but function. The main advantage of functions is code Reusability.

Python Support Two Types of Function

- **Built in Function**
- **User Defined Function**

Built in Function

The functions which are coming along with Python software automatically, are called built in functions or pre defined functions

User Defined Function

The functions which are developed by programmer explicitly according to business requirements ,are called user defined functions.

Syntax

```
def  function_Name(parameters):
                # Logic
return value
```

Where return is optional

Parameters

Parameters are inputs to the function. If a function contains parameters, then at the time of calling, compulsory we should provide values otherwise, otherwise we will get error.

User Defined Function

Syntax

```
def  function_Name(parameters):
                # Logic
return value
```

Where return is optional

return Statement

Function can take input values as parameters and executes business logic and return output to the caller with return statement.

User Defined Function

Types of Parameters allowed in Function

1. Positional Arguments
2. Keyword Arguments
3. Default Arguments
4. Variable Length Arguments

Variable in Function

1. Global Variables

The variables which are declared outside of function are called global variables. These variables can be accessed in all functions of that module

2. Local Variables

The variables which are declared inside a function are called local variables. Local variables are available only for the function in which we declared it.i.e from outside of function we cannot access

NOTE:

If global variable and local variable having the same name then we can access global variable inside a function as using globals() .

Various Functions

1. Recursive Function

A function that calls itself is known as Recursive Function .

2. Anonymous Function or Lambda Function

Sometimes we can declare a function without any name, such type of nameless functions are called anonymous functions or lambda functions. This function internally returns expression value and we are not required to write return statement explicitly

Syntax

```
lambda  argument_list : expression
```

Various Functions

3. Filter Function

We can use filter() function to filter value from the given sequence based on some condition .

Syntax

```
filter (function, sequence)
```

Where function argument is responsible to perform conditional check . And sequence can be list , set or tuple

Various Functions

4. Map Function

For every element present in the given sequence, apply some functionality and generate new element with the required modification. For this requirement we should go for map() function.

Syntax

```
map (function, sequence)
```

The function can be applied on each element of sequence and generates new sequence.

Various Functions

5. Reduce Function

reduce() function reduces sequence of elements into a single element by applying the specified function

Syntax

```
map (function, sequence)
```

reduce() function present in functools module and hence we should write import statement.

Functions Aliasing

For the existing function we can give another name, which is nothing but function aliasing.

Example

```
def operatingSystem(name):
    print("Welcome to Operating System :", name)


windows = operatingSystem
print("ID of Function Name operatingSystem ", id(operatingSystem))
print("ID of Function Name windows ", id(windows))

operatingSystem('windows ')
windows('windows')
```

```
Lab28 ×
/usr/local/bin/python3 /Users/om/SB015/string/Lab28.py
ID of Function Name operatingSystem  4297981760
ID of Function Name windows  4297981760
Welcome to Operating System : windows
Welcome to Operating System : windows
```

Functions Decorators

Decorator is a function which can take a function as argument and extend its functionality and returns modified function with extended functionality.

The main objective of decorator functions is we can extend the functionality of existing functions without modifies that function

Be Safe

Thank You
Happy Learning
Keep Watching