- Variable access in Functions
- Practice Problems
- Recursive Function
- Module and packages

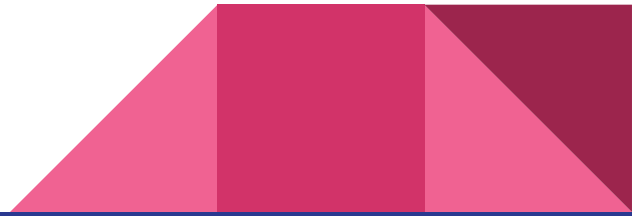
Functions

What is a function in Python?

In Python, a function is a group of related statements that performs a specific task.

Functions help break our program into smaller and modular chunks. As our program grows larger and larger, functions make it more organized and manageable.

Furthermore, it avoids repetition and makes the code reusable.



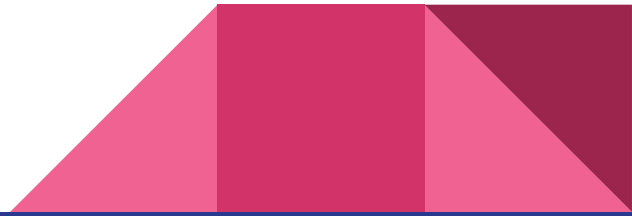
Examples

Python built in functions :

input() print() round() etc..,

These are all python built in functions

print(“ Hello Python World “)



Syntax of Function:

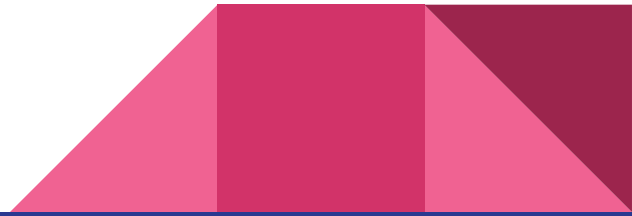
```
def function_name():
    Logical statement(Task performance)
```

Example of a function

```
def greet(name):
    """
    This function greets to
    the person passed in as
    a parameter
    """
    print("Hello, " + name + ". Good morning!")
```

Types of Functions

- **Function just performs a task**
- **Function performs a task and return a value.**



Scope of variables in function

- **Global**
- **Local**

Local Scope

A variable created inside a function belongs to the *local scope* of that function, and can only be used inside that function.

Example

A variable created inside a function is available inside that function:

```
def myfunc():
  x = 300
  print(x)

myfunc()
```

Function Inside Function

As explained in the example above, the variable `x` is not available outside the function, but it is available for any function inside the function:

Example

The local variable can be accessed from a function within the function:

```
def myfunc():
  x = 300
  def myinnerfunc():
    print(x)
  myinnerfunc()

myfunc()
```

Global Scope

A variable created in the main body of the Python code is a global variable and belongs to the global scope.

Global variables are available from within any scope, global and local.

Example

A variable created outside of a function is global and can be used by anyone:

```
x = 300

def myfunc():
  print(x)

myfunc()

print(x)
```

Global Keyword

If you need to create a global variable, but are stuck in the local scope, you can use the `global` keyword.

The `global` keyword makes the variable global.

Example

If you use the `global` keyword, the variable belongs to the global scope:

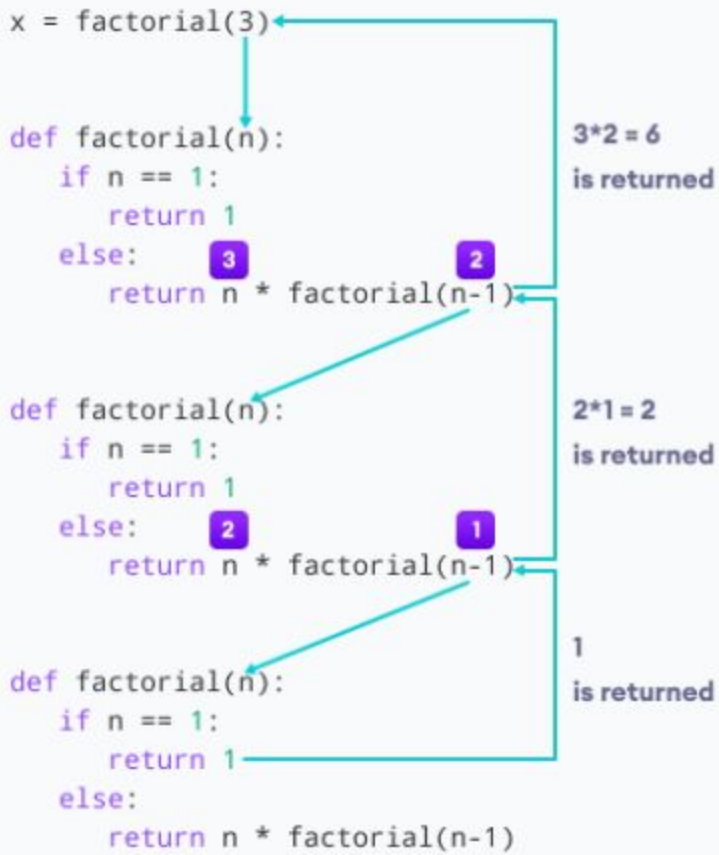
```
def myfunc():
  global x
  x = 300

myfunc()

print(x)
```

Python Recursion

What is recursion?

Recursion is the process of defining something in terms of itself.

A physical world example would be to place two parallel mirrors facing each other. Any object in between them would be reflected recursively.


x = factorial(3)
def factorial(n):
if n == 1:
return 1
else:
return n * factorial(n-1)
3
2
def factorial(n):
if n == 1:
return 1
else:
return n * factorial(n-1)
2
1
def factorial(n):
if n == 1:
return 1
else:
return n * factorial(n-1)
3*2 = 6
is returned
2*1 = 2
is returned
1
is returned

Advantages of Recursion

1. Recursive functions make the code look clean and elegant.

Disadvantages of Recursion

1. Sometimes the logic behind recursion is hard to follow through.
2. Recursive calls are expensive (inefficient) as they take up a lot of memory and time.
3. Recursive functions are hard to debug.

Module in Python

What is a Module?

Consider a module to be the same as a code library.

A file containing a set of functions you want to include in your application.

Package in Python

- **A package is basically a directory with Python files and a file with the name __init__ . py.**

- **This means that every directory inside of the Python path, which contains a file named __init__ . py, will be treated as a package by Python**