Local Variables

A variable created inside function is called local variable.
Local variables are used within function; these variables cannot accessible outside function.
Local variables are crated when function is invoked and deleted after execution of function. Lifetime of local variables is until execution of function.

When function is called, execution control switched from calling place to called function and after execution of called function, it returns to calling place.

Example:

```
def fun1(): # function without parameters/arguments
    x=100 # Local variable
    y=200 # Local variable
    print(x,y)


fun1() # executing function/invoking function/calling function
print("continue")
```

Output:

100 200
Continue

Example:

```
def fun1():
    x=100 # L.V
    y=200 # L.V

def fun2():
```

```
    print(x)
    print(y)


fun1()
fun2()
```

Output:

```
Traceback (most recent call last):
  File "E:/python7amdec23/funtest4.py", line 11, in <module>
    fun2()
  File "E:/python7amdec23/funtest4.py", line 6, in fun2
    print(x)
NameError: name 'x' is not defined
```

Example:

```
def fun1():
    x=100
    print(x)
    x=x+100

def fun2():
    x=400
    print(x)


fun1() # calling function/executing function
fun2() # calling function/executing function
```

Output:

```
100
400
```

Example:

```
def fun1():          # Old Function
   print("abc")

fun1()  # Calling function

def fun1():   # New Function
   print("xyz")


fun1() # Calling function
```

Output:

abc
xyz

Global Variables

The variable created outside the function is called global variable. Global variables are used to share data between number of functions.
Global variables are created when program is executed and global variables are deleted after execution of program.

Example:

```
x=100 # Global variable
y=200 # Global variable

def fun1():
   print(x)
   print(y)
```

```
def fun2():
    print(x)
    print(y)


fun1()
fun2()
print(x)
print(y)
```

Output:

```
100
200
100
200
100
200
```

Example:

```
def fun1():
    print(x)

fun1()

def fun2():
    print(y)


x=100
y=200
fun1()
fun2()
```

Output:

```
Traceback (most recent call last):
  File "E:/python7amdec23/funtest8.py", line 4, in <module>
    fun1()
  File "E:/python7amdec23/funtest8.py", line 2, in fun1
    print(x)
NameError: name 'x' is not defined
```

Example:

```
def add():
    print(f'sum of {a} and {b} is {a+b}')

def sub():
    print(f'diff of {a} and {b} is {a-b}')

def multiply():
    print(f'product of {a} and {b} is {a*b}')

def div():
    print(f'division of {a} and {b} is {a/b}')

a=int(input("Enter First Number ")) # G.V
b=int(input("Enter Second Number ")) # G.V
add()
sub()
multiply()
div()
```

Output:

```
Enter First Number 5
Enter Second Number 2
```

sum of 5 and 2 is 7
diff of 5 and 2 is 3
product of 5 and 2 is 10
division of 5 and 2 is 2.5

Example:

```
x=100 # G.V

def fun1():
    y=200 # L.V
    print(x) # 100
    print(y) # 200

def fun2():
    x=500 # Create L.V
    print(x) # 500

def fun3():
    print(x)


fun1()
fun2()
fun3()
```

Output:

100
200
500
100

global keyword

A global keyword is a keyword that allows a user to modify a global variable outside the current scope. It is used to create global variables in Python from a non-global scope, i.e. inside a function. Global keyword is used inside a function only when we want to do assignments or when we want to change a global variable.

Global keyword is used to modify or update global variable inside function. A function can access global variables without using global keyword.

Syntax:
global <variable-name>,<variable-name>,…

after this statement declaration, all variables are defined as global variables.

Example:

```
x=100 # G.V

def fun1():
    print(x) # Accessing G.V

def fun2():
    x=200 # Create L.V
    print(x)

def fun3():
    global x
    x=500
    print(x)
```

```
fun1()
fun2()
fun3()
print(x)
fun1()
```

Output:

```
100
200
500
500
500
```

Example:

```
# finding area of triangle

base=0.0 # G.V
height=0.0 # G.V

def read():
    global base,height
    base=float(input("Enter Base "))
    height=float(input("Enter Height "))

def find_area():
    area=0.5*base*height
    print(f'Area of triangle is {area:.2f}')


read()
```

find_area()

Output:

Enter Base 1.2
Enter Height 1.5
Area of triangle is 0.90