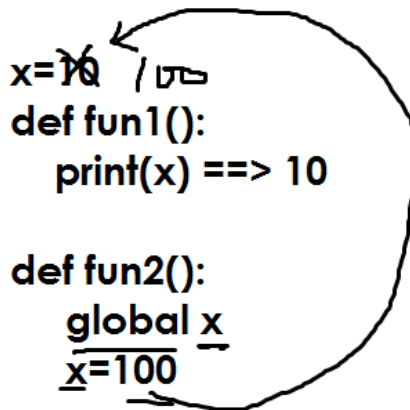


## global keyword

The [global](#) statement is a declaration which holds for the entire current code block. It means that the listed identifiers are to be interpreted as globals. It would be impossible to assign to a global variable without global, although free variables may refer to globals without being declared global.



```
x=10 / 10
def fun1():
    print(x) ==> 10

def fun2():
    global x
    x=100
```

### Example:

```
x=10
def fun1():
    print(x)
def fun2():
    global x
    x=200

def main():
    fun2()
    fun1()
```

main()

### Output:

200

### Example:

```
# find area of triangle
base=0.0
height=0.0
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}')
```

```
def main():  
    read()  
    find_area()
```

```
main()
```

**Output:**

```
enter base1.5  
enter height1.2  
area of triangle is 0.90  
>>>
```

**Example:**

```
x=100 # global variable  
def fun1():  
    global x  
    print(x) # 100
```

```
    x=200  
def fun2():  
    x=200 # local variable  
    print(x)  
    global x  
    print(x)
```

```
def main():  
    fun1()
```

```
main()
```

Output:

The above code generate syntax error

In order access and update global variable with function, first we declare those variables with global keyword

## Function with arguments

Function with arguments receive input at the time of calling or invoking function

If function required input to perform some operation, we define function with arguments

Python allows you to write function with 4 types of arguments

1. Required arguments or required positional arguments
2. Default arguments or optional arguments
3. Variable length arguments
4. Keyword arguments

Arguments are local variables, which receive values at the time of calling function

## Required arguments or required positional arguments

Required arguments required values at time of calling/invoking function. If values are not given for these arguments it generates syntax error.

### Syntax:

```
def function-name(arg1,arg2,arg2,arg3,...):  
    statement-1  
    statement-2  
    statement-3
```

### Example:

```
def draw_line(ch,l):  
    for i in range(l):  
        print(ch,end="")  
    print()  
  
def main():  
    draw_line("*",30)  
    draw_line("$",20)  
    draw_line("^",10)  
    draw_line(l=10,ch='@')  
main()
```

### Output:

\*\*\*\*\*

```
$$$$$$$$$$$$$$$$$$$$
^AAAAAAAAA
@@@@@@@@
>>>
```

**Example:**

**# write a function to count number of vowels in input string**

```
def count_vowels(s):
    c=0
    for ch in s:
        if ch in "aeiouAEIOU":
            c=c+1
    return c

def main():
    str1=input("enter any string")
    x=count_vowels(str1)
    print(f'{str1} is having {x} vowels')

main()
```

**Output:**

```
enter any stringjava
java is having 2 vowels
>>>
```

**return keyword**

**return** is a passes control statement or branching statement

return is a keyword

return statement, return value from called function to calling function.

After returning value, it terminates execution of function

**Syntax:**

return [expression/value]

