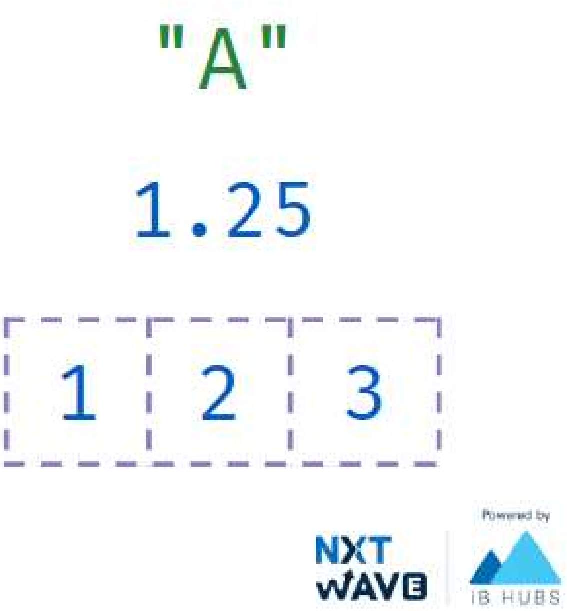
# Cheat Sheet

# Scope & Namespaces

## Object

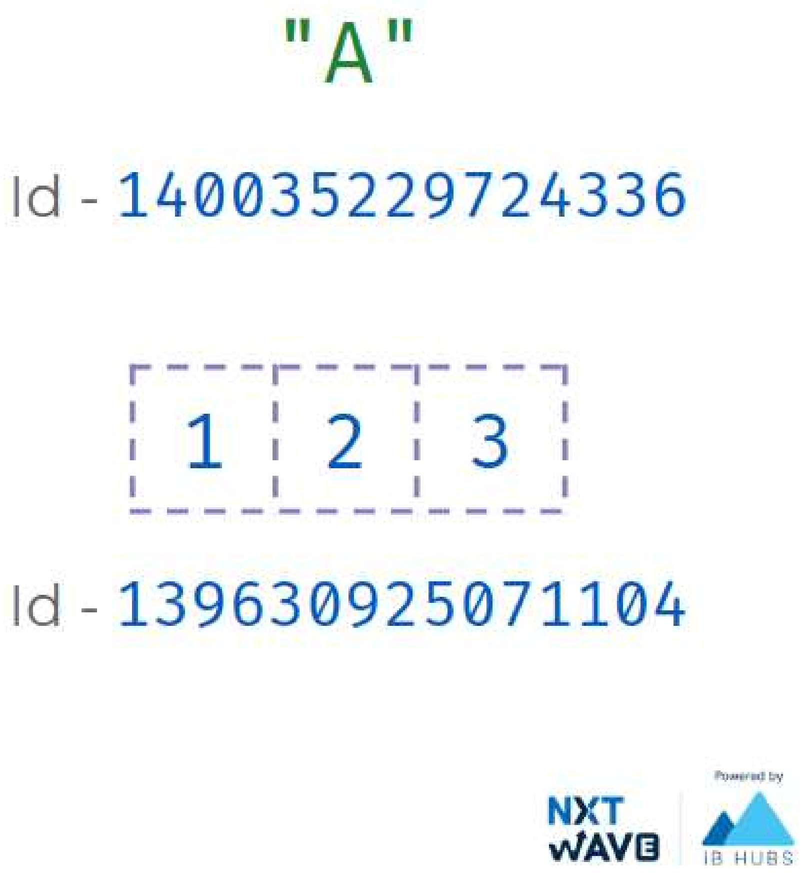
In general, anything that can be assigned to a variable in Python is referred to as an **object**.

Strings, Integers, Floats, Lists, Functions, Module etc. are all objects.



## Identity of an Object

Whenever an object is created in Python, it will be given a **unique identifier (id)**.This unique id can be different for each time you run the program.

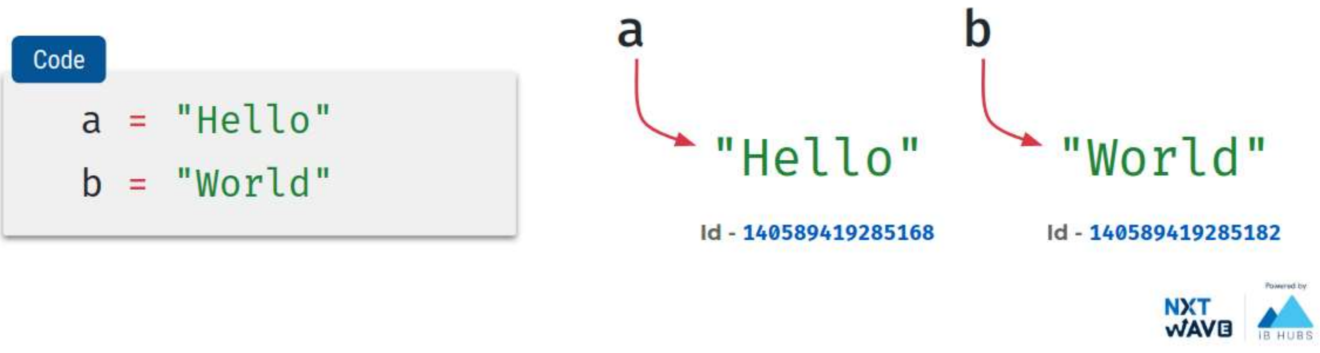


Every object that you use in a Python Program will be stored in Computer Memory

The unique id will be related to the location where the object is stored in the **Computer Memory**.

## Name of an Object

**Name** or **Identifier** is simply a name given to an object.

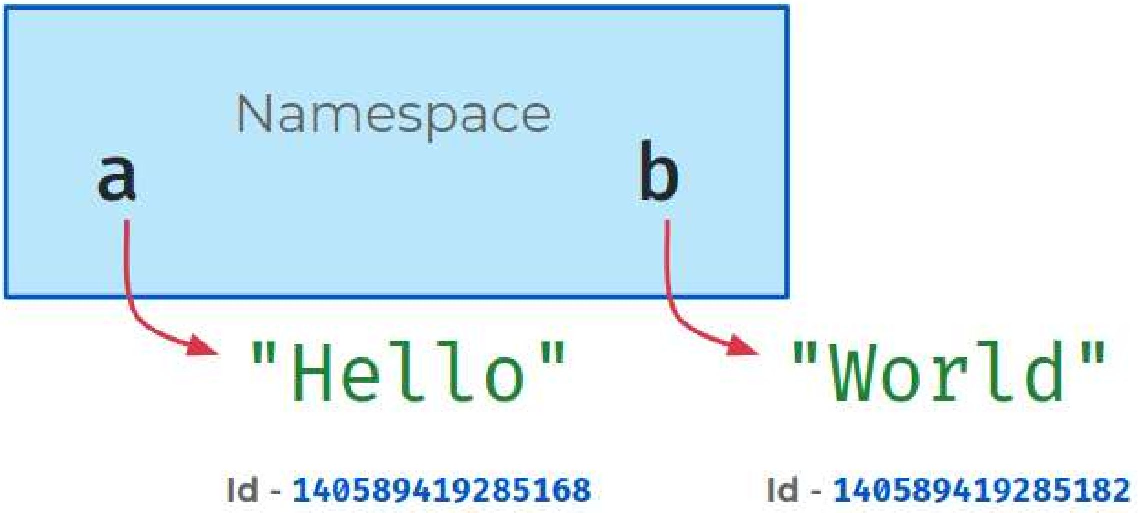

Code

```
a = "Hello"
b = "World"
```

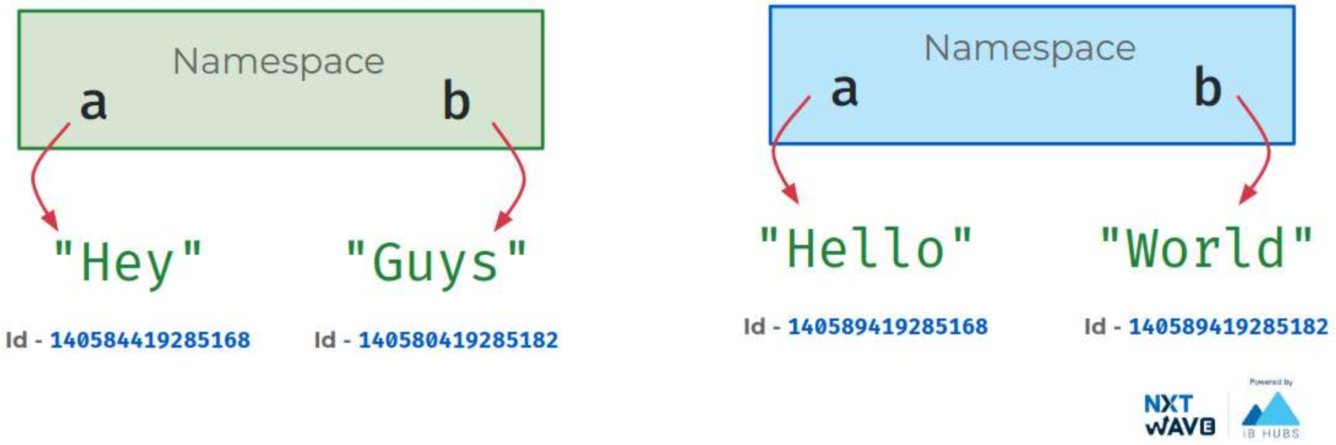
# Namespaces

A **namespace** is a collection of currently defined names along with information about the object that the name references.

It ensures that names are **unique** and won't lead to any conflict.



Namespaces allow us to have the same name referring different things in **different namespaces**.



## Code

PYTHON

```
def greet_1():
    a = "Hello"
    print(a)
    print(id(a))

def greet_2():
```

```python
def greet_2():
    a = "Hey"
    print(a)
    print(id(a))

print("Namespace - 1")
greet_1()
print("Namespace - 2")
greet_2()
```

Collapse

**Output**

```
Namespace - 1
Hello
140639382368176
Namespace - 2
Hey
140639382570608
```

# Types of namespaces

As Python executes a program, it creates namespaces as necessary and forgets them when they are no longer needed.

Different namespaces are:

1. Built-in
2. Global
3. Local

## Built-in Namespace

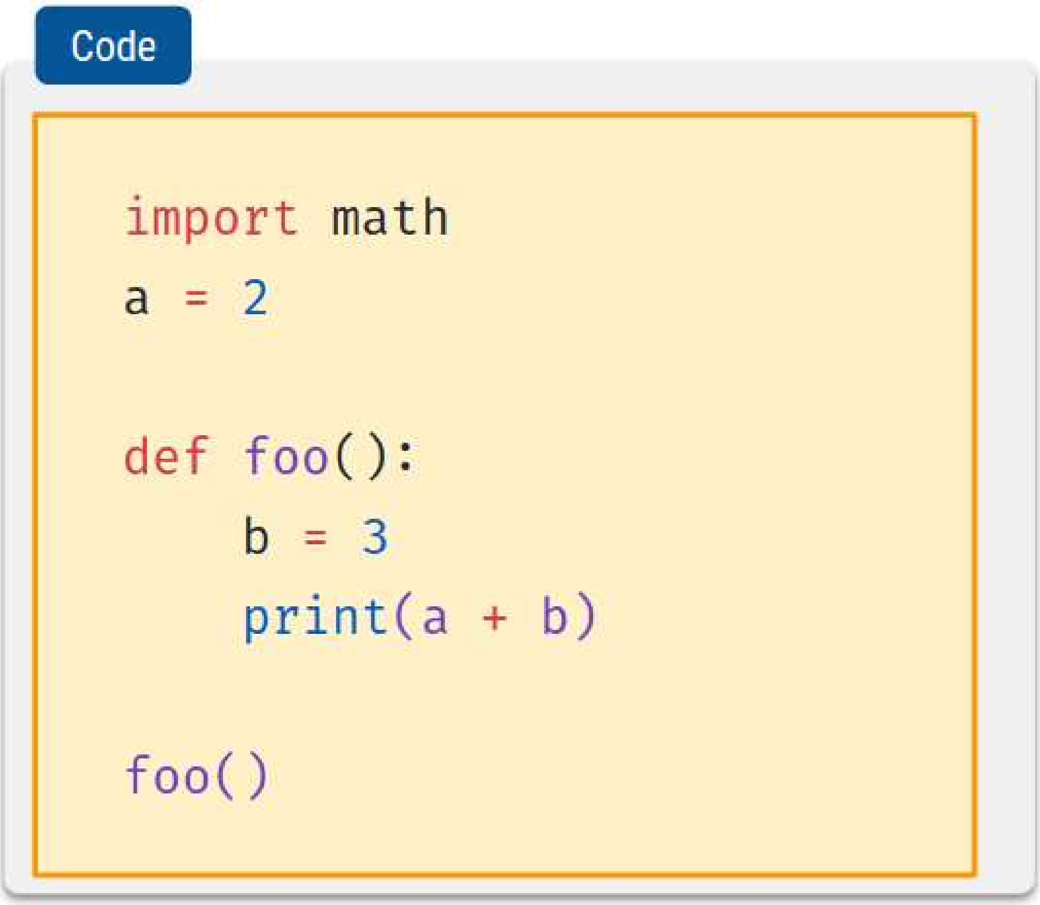
Created when we start executing a Python program and exists as long as the program is running.

This is the reason that built-in functions like **id(), print()** etc. are always available to us from any part of the program.

## Global Namespace

This namespace includes all names defined directly in a module (outside of all functions).

It is created when the module is loaded, and it lasts until the program ends.



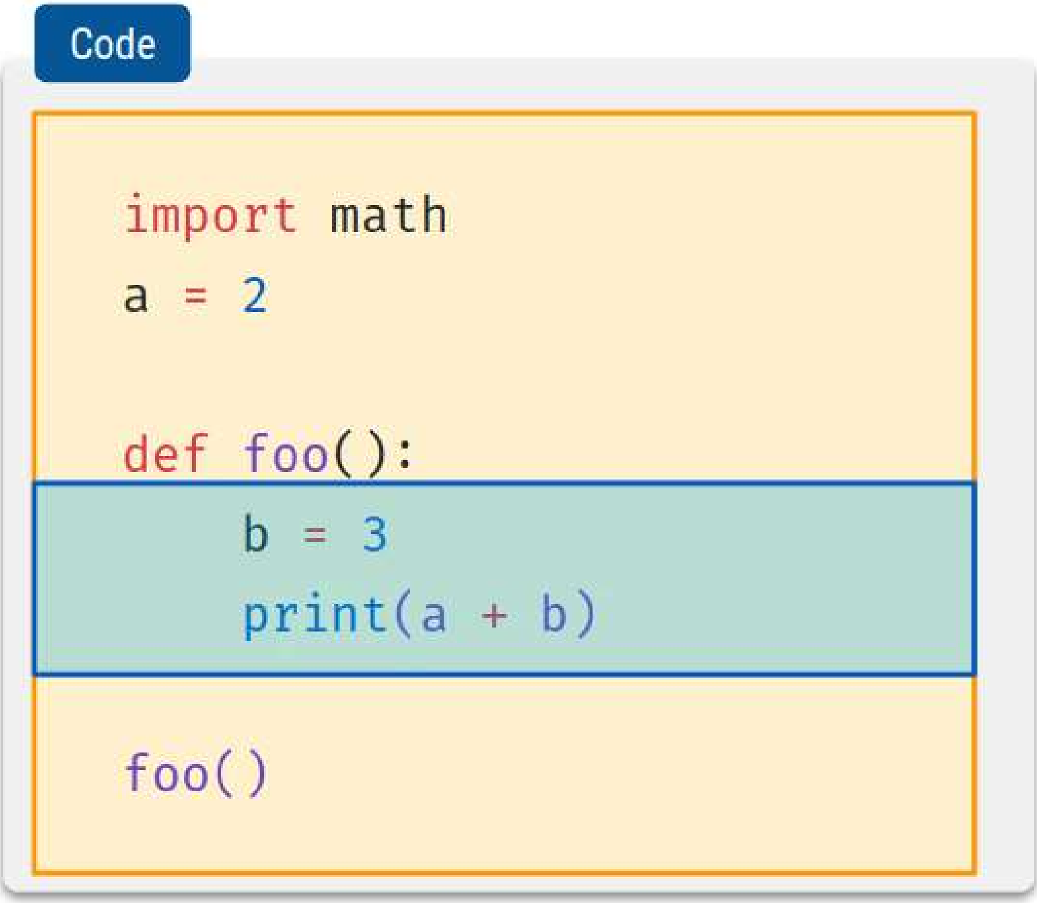
```python
import math
a = 2

def foo():
    b = 3
    print(a + b)

foo()
```

## Local Namespace

Modules can have various

functions and classes .

A new local namespace is created when a function is called, which lasts until the function returns.

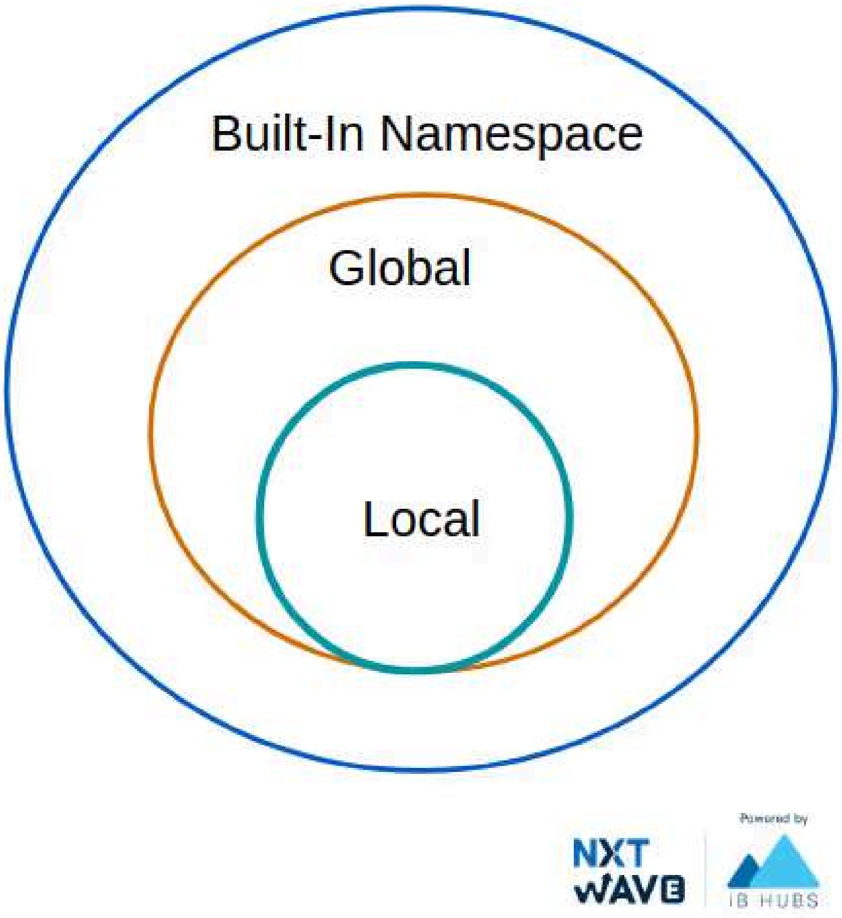


```
import math
a = 2

def foo():
    b = 3
    print(a + b)

foo()
```

## Scope of a Name

The scope of a name is the region of a program in which that name has meaning.

Python searches for a name from the inside out, looking in the local , global , and finally the built-in namespaces.



## Global variables

In Python, a variable defined outside of all functions is known as a **global variable**.

This variable name will be part of **Global Namespace**.

***Example 1***

Code

PYTHON

```python
x = "Global Variable"
print(x)

def foo():
    print(x)

foo()
```

### Output

```
Global Variable
Global Variable
```

### *Example 2*

### Code

PYTHON

```python
def foo():
    print(x)

x = "Global Variable"

foo()
```

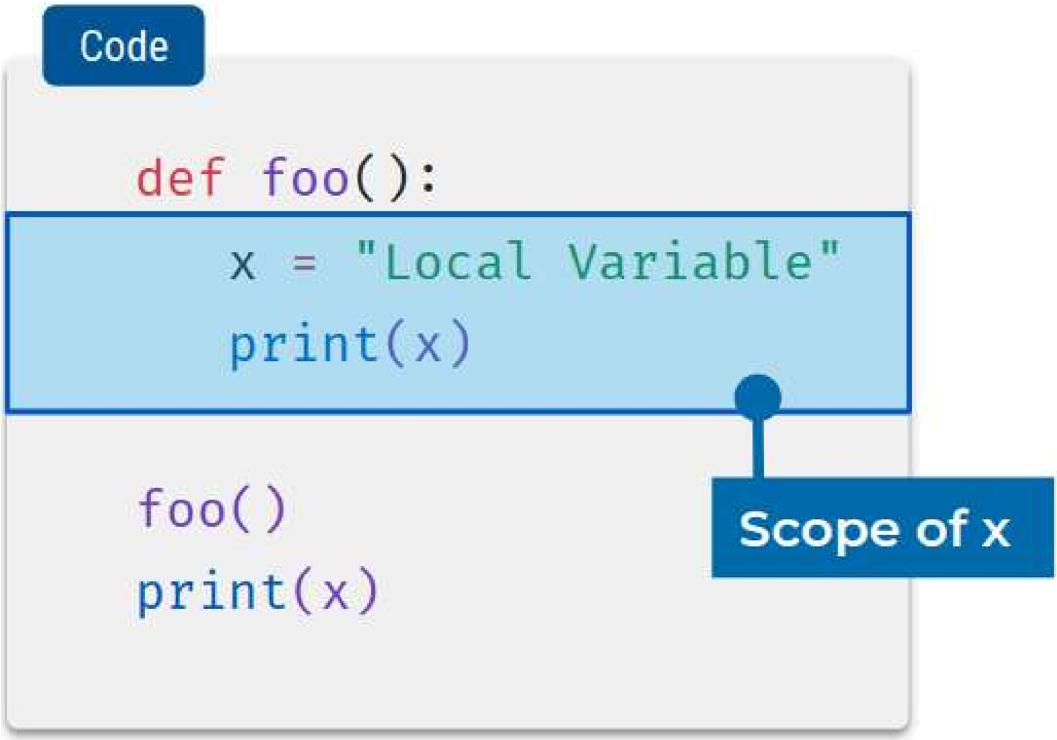
### Output

```
Global Variable
```

## Local Variables

In Python, a variable defined inside a function is a local variable.

This variable name will be part of the Local Namespace which will be created when the function is called and lasts until the function returns.


Code

```
def foo():
    x = "Local Variable"
    print(x)

foo()
print(x)
```

Scope of x

### Code

PYTHON

```python
def foo():
    x = "Local Variable"
    print(x)

foo()
print(x)
```

### Output

```
Local Variable
NameError: name 'x' is not defined
```

As,

x is not declared before assignment, python throws an error.

## Local Import

### Code

PYTHON

```python
def foo():
   import math
   print(math.pi)

foo()
print(math.pi)
```

### Output

```
3.141592653589793
NameError: name 'math' is not defined
```

## Local Variables & Global Variables

### Code

PYTHON

```python
x = "Global Variable"

def foo():
    x = "Local Variable"
    print(x)

print(x)
foo()
print(x)
```

### Output

```
Global Variable
Local Variable
Global Variable
```

## Modifying Global Variables

global keyword is used to define a name to refer to the value in Global Namespace.

## Code

PYTHON

```python
x = "Global Variable"

def foo():
    global x
    x = "Global Change"
    print(x)

print(x)
foo()
print(x)
```

## Output

```
Global Variable
Global Change
Global Change
```

Submit Feedback