1. In Python 3, what is the output of type(range(5)). (What data type it will return).

**Hint**: [range() in Python](https://pynative.com/python-range-function/)

* int
* list
* **range**
* None

**Explanation**:

in Python 3, the range()  function returns range object, not [list](https://pynative.com/python-lists/).

2. What is the data type of print(type(10))

* float
* integer
* **int**

3. What is the data type of the following

aTuple = (1, 'Jhon', 1+3j)

print(type(aTuple[2:3]))

**Refer**:

* [Python Data types](https://pynative.com/python-data-types/)
* [tuples in Python](https://pynative.com/python-tuples/)
* list
* complex
* **tuple**

**Explanation**:

When we access a tuple using the subscript atuple[start : end] operator, it will always return a tuple. We also call it tuple slicing. (taking a subset of a tuple using the range of indexes).

4. What is the data type of print(type(0xFF))

* number
* hexint
* **hex**
* int

**Explanation**:

We can represent integers in binary, octal and hexadecimal formats.

* 0b or 0B for Binary and base is 2
* 0o or 0O for Octal and base is 8
* 0x or 0X for Hexadecimal and base is 16

5. What is the result of print(type([]) is list)

* False
* **True**

6. What is the output of print(type({}) is set)

* **True**
* False

**Explanation**:

When the object is created without any items inside the curly brackets ( {} ) then it will be created as a [**dictionary**](https://pynative.com/python-dictionaries/)which is another built-in data structure in Python.

So whenever you wanted to create an empty [set](https://pynative.com/python-sets/) always use the set() constructor

7. What is the output of the following code?

x = 50

def fun1():

x = 25

print(x)

fun1()

print(x)

* NameError
* 25  
  25
* **25  
  50**

**Explanation**:

A variable declared outside of all functions has a GLOBAL SCOPE. Thus, it is accessible throughout the file. And variable declared inside a function is a local variable whose scope is limited to its function.

8. What is the output of the following variable assignment?

x = 75

def myfunc():

x = x + 1

print(x)

myfunc()

print(x)

Refer [Variables in Python](https://pynative.com/python-variables/).

* **Error**
* 76
* 1
* None

**Explanation**:

Here we have not used a global keyword to reassign a new value to global variable x into myfunc() so Python assumes that x is a local variable.

It means you are accessing a local variable before defining it. that is why you received a UnboundLocalError: local variable 'x' referenced before assignment

**The correct way** to modify the global variable inside a function:

x = 75

def myfunc():

global x

x = x + 1

print(x)

myfunc()

print(x)

9. Select all the right ways to create a string literal Ault'Kelly

* str1 = 'Ault\\'Kelly'
* **str1 = 'Ault\'Kelly'**
* **str1 = """Ault'Kelly"""**

10. What is the output of the following code

print(bool(0), bool(3.14159), bool(-3), bool(1.0+1j))

* False True False True
* True True False True
* True True False True
* **False True True True**

**Explanation**:

* If we pass A zero value to bool() construtor, it will treat as a boolean False
* Any non-zero value is boolean True

11. Please select the correct expression to reassign a global variable “**x**” to 20 inside a function fun1()

x = 50

def fun1():

# your code to assign global x = 20

fun1()

print(x) # it should print 20

* global x =20
* global var x
* x = 20
* global.x = 20
* **global x**
* **x = 20**

12. Select all the valid String creation in Python

* + str1 = 'str1'
  + str1 = "str1"
  + str1 = '''str'''
  + str1 = 'str1'
  + str1 = "str1""
  + str1 = '''str1''
* **str1 = str(Jessa)**

**Explanation**:

Strings in Python are surrounded by either single quotation marks or double quotation marks. Also, You can create a multiline string using three quotation marks.

13. What is the output of the following code

def func1():

x = 50

return x

func1()

print(x)

* 50
* **NameError**
* None
* 0

**Explanation**:

You will get a NameError: name 'x' is not defined. To access the function’s return value we must accept it using an assignment operator like this

def myfunc():

x = 50

return x

x = myfunc()

print(x)