Functions in Python

Built-In and User Define

OBJECTIVE

- HOW TO DEFINE A FUNCTION
- HOW TO CALL A FUNCTION
- FUNCTION ARGUMENTS
  - POSITIONAL
  - KEYWORD
  - VARIABLE LENGTH ARGUMENTS
- SCOPE OF VARIABLES
  - LOCAL
  - GLOBAL
  - NONLOCAL
- ANONYMOUS FUNCTION

  (LAMBDA FUNCTION)

FUNCTION

- FUNCTION IS A GROUP OF RELATED STATEMENTS THAT PERFORM A SPECIFIC TASK
- FUNCTIONS HELP BREAK OUR PROGRAM INTO SMALLER AND MODULAR CHUCKS

➢ Functions in python are defined using the block keyword "def", followed with the function's name as the block's name.
For example:

```
def my_fun():
        print("GLA Online classes")
```

➢ Functions may also receive arguments (variables passed from the caller to the function. For example:

```
def my_fun_with_argv(name, msg):
        print(f"Hello {name}, message for you {msg} ")
```

➢ Functions may return a value to the caller, using the keyword- 'return' . For example:

```
def sum_two_num(a, b):
        return a + b
```

DEFINING FUNCTION

Functions in python are defined using the block keyword "def"

def marks the start of function

function name to uniquely identify a function.

def function_name (parameter) :

Argument to pass a value in function

colon(:) to mark end of function header

EXAMPLE OF FUNCTION

```
def prime_or_not(number):
    for i in range(2, number):
        if number % i == 0:
            return False
    else:
        return True
# filer prime numbers
lst = [2, 4, 7, 3, 14, 9, 10, 67]
res = []
for n in lst:
    if prime_or_not(n):
            res.append(n)
print(res)
```

Output : [2, 7, 3, 67]

CALLING A FUNCTION

HOW TO CALL A FUNCTION?

Once we have defined a function, we can call it from another function, program or even the Python prompt.

To call a function we simply type the function name with appropriate parameters.

```
>>> prime_or_not(2)

>>>True
```

TYPES OF FUNCTION

THERE ARE TWO TYPES OF FUNCTION IN PYTHON

- BUILT-IN FUNCTION

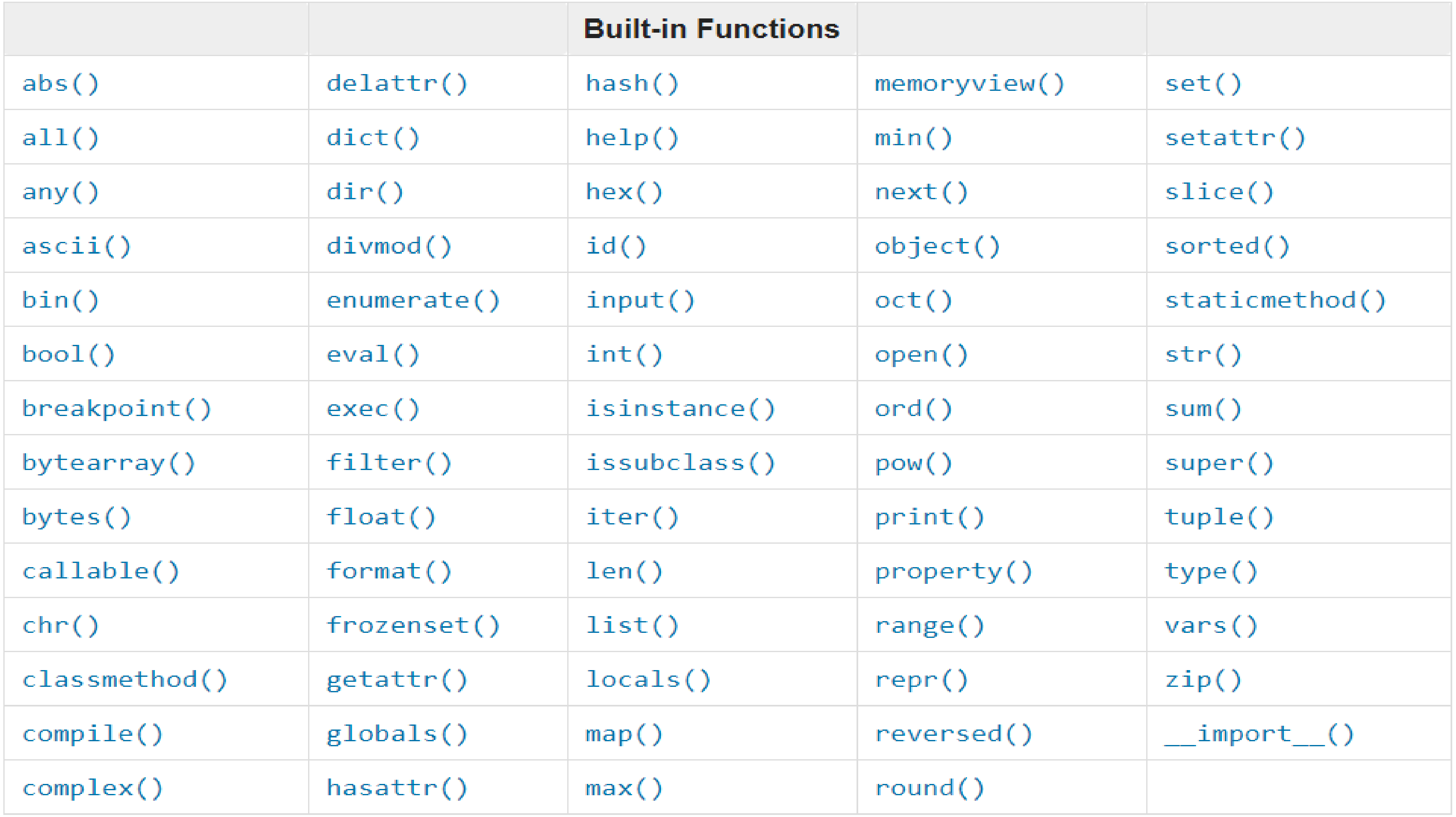
Example: print(), input(), eval().

- USERDEFINE FUNCTION

Example:
```
def my_addition(x,y):
        sum = x +y
        return sum
```

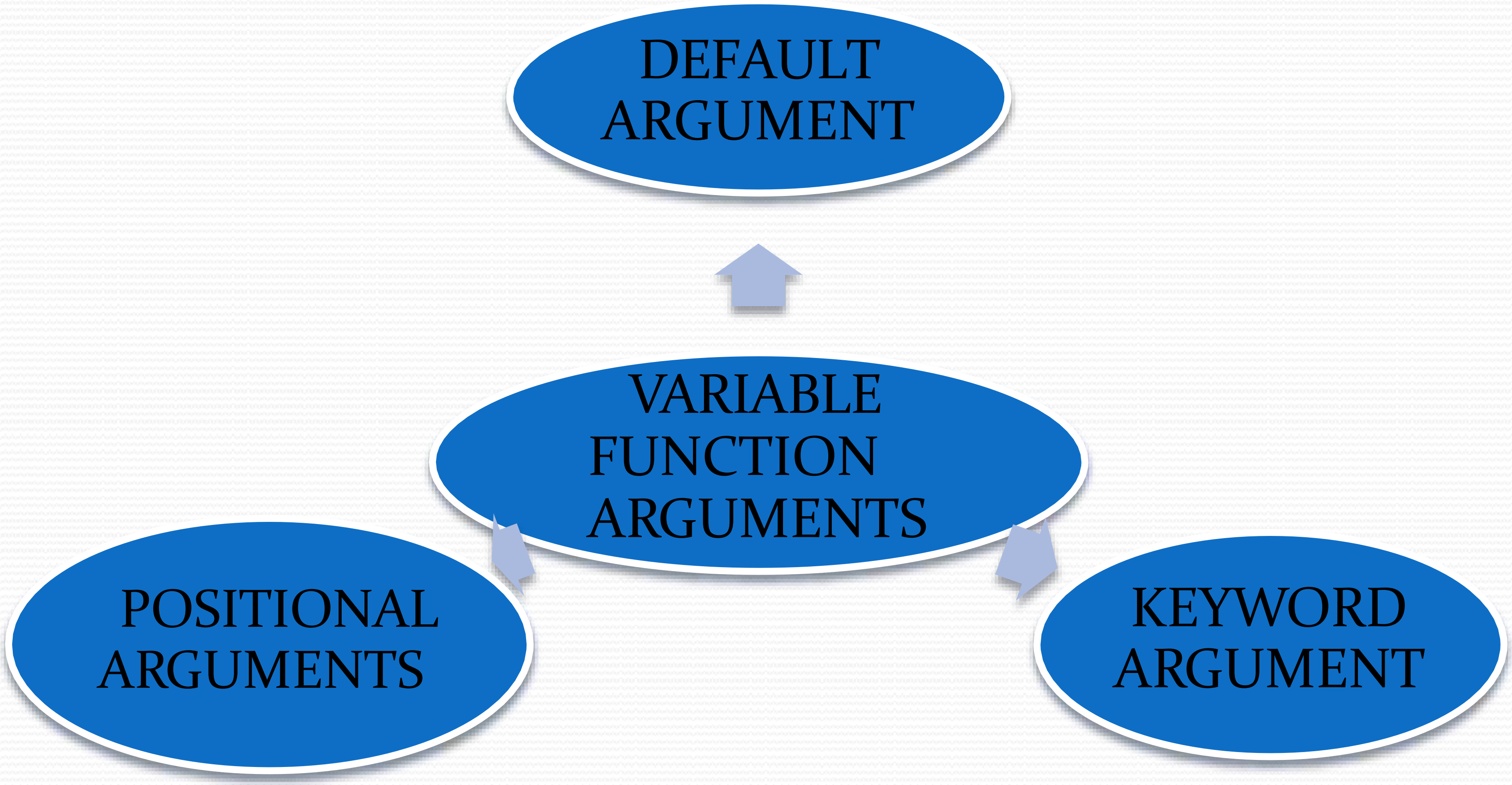
BUILT - IN FUNCTION

- THERE ARE 69 BUILT-IN FUNCTION VERSION3.8.2. CHANGES WITH RESPECT TO UPDATEDVERSION.



		Built-in Functions		
abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	slice()
ascii()	divmod()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	

FUNCTION ARGUMENTS



DEFAULT ARGUMENTS

- Default value to function argument is passed using assignment operator ‘ = ‘.

Example:

```
def greet(name, msg = "Good morning!"):
        print("Hello, " name + ', ' + msg)
```

- Non-default argument cannot follow default argument

Example: `def greet(msg = "Good morning!", name):` ✗

SyntaxError:

non-default argument follows default argument

KEYWORD ARGUMENTS

- When we call a function with some values, these values get assigned to the arguments according to their position .
- Keyword arguments follows positional argument.

EXAMPLE OF KEYWORD ARGUMENT

- greet(name = "Bruce",msg = "How do you do?")

  2 keyword arguments

- greet(msg = "How do you do?",name = "Bruce")

  2 keyword arguments (out of order)

- greet("Bruce",msg = "How do you do?")

  1 positional, 1 keyword argument

- greet(name="Bruce","How do you do?")

SyntaxError:

non-keyword arg after keyword arg

ARBITRARY ARGUMENTS

(Variable length arguments)

- If number of arguments unknown we use arbitrary arguments
- (*) Asterisk before arguments define arbitrary arguments.

Example:

```
def greet(*names):
    for name in names:
        print("Hello",name)
>>>greet("Monica", "Luke", "Steve", "John")
```

SCOPE OF VARIABLES

- **Global Variables**

In Python, a variable declared outside of the function or in global scope is known as global variable. This means, global variable can be accessed inside or outside of the function.

Let's see an example on how a global variable is created in Python.

Example 1: Create a Global Variable

```
x = "global "
def my_function():
            print("x inside :", x)
foo()
print("x outside:", x)
```

SCOPE OF VARIABLES

- **Local Variables**

A variable declared inside the function's body or in the local scope is known as local variable.

Example: Accessing local variable outside the scope

```
def my_function():
        y = "local "
my_function()
print(y)
```

```
Output : NameError: name 'y' is not defined
```

SCOPE OF VARIABLES

- **Global and local variables**

Here, we will show how to use global variables and local variables in the same code.

Example : Using Global and Local variables in same code

```
x = "global"
def function():
        global x
        y = "local"
        x = x * 2
        print(x)
        print(y)
function()
print(x)
```

```
Output:
global global
Local
global global
```

SCOPE OF VARIABLES

Example 5: Global variable and Local variable with same name

```
x = 5
def function():
        x = 10
        print("local x:", x)
function()
print("global x:", x)
```

Output:

```
local x: 10
global x: 5
```

SCOPE OF VARIABLES

- **Nonlocal Variables**

Nonlocal variable are used in nested function whose local scope is not defined. This means, the variable can be neither in the local nor the global scope.

Let's see an example on how a global variable is created in Python.

We use nonlocal keyword to create nonlocal variable.

Example: Create a nonlocal variable

```
def outer():
        x = "local"
        def inner():
                nonlocal x
                x = "nonlocal"
                print(" inner:", x)
        inner()
        print("outer:", x)
outer()
```

Output:

```
inner: nonlocal
outer: nonlocal
```

ANONYMOUS FUNCTION

- Function without function name.
- Function is defined using lambda keyword, hence function is also called Lambdafunction.
- Used for short period in python.
- Can have any number of arguments but only single expression.
- Specially used with built-in functions like filter(), map() etc.

EXAMPLE OF LAMBDA FUNCTION

1.

```
double = lambda x: x * 2
print(double(5))
```

2.

```
my_list = [1, 5, 4, 6, 8, 11, 3, 12]
new_list = list(filter(lambda x: (x%2 == 0) , my_list))
print(new_list)
```

RESOURCES

- https://hyperskill.org/curriculum
- www.programiz.com/python-programming
- https://exercism.io
- https://docs.python.org/3/

Practice Questions

- 1. Write a function to calculate area and perimeter of a rectangle.
- 2. Write a function to calculate area and circumference of a circle.
- 3. Write a function to calculate power of a number raised to other. E.g.- ab.
- 4. Write a function to tell user if he/she is able to vote or not.
- (Consider minimum age of voting to be 18.)
- 5. Write a function to check if a number is even or not.
- 6. Write a function to check if a number is prime or not.
- 7. Write a function to find factorial of a number but also store the factorials calculated in a dictionary as done in the Fibonacci series example.
- 8. Write a Python program to detect the number of local variables declared in a function

Link for MCQ questions

click here

THANK YOU
HAVE A NICE DAY!