**keywords in python**

Keywords are the reserved words in Python. We cannot use a keyword as a variable name, function name or any other identifier.

In Python, keywords are case sensitive.

There are 33 keywords in Python 3.7. This number can vary slightly over the course of time.

All the keywords except True, False and None are in lowercase and they must be written as they are. The list of all the keywords is given below.

### True, False

True and False are truth values in Python. They are the results of comparison operations or logical (Boolean) operations in Python.

### None

None is a special constant in Python that represents the absence of a value or a null value.

### and, or , not

and, or, not are the logical operators in Python. and will result into True only if both the operands are True.

or will result into True if any of the operands is True.

not operator is used to invert the truth value.

### as

as is used to create an alias while importing a module. It means giving a different name (user-defined) to a module while importing it.

### assert

assert is used for debugging purposes.

While programming, sometimes we wish to know the internal state or check if our assumptions are true. assert helps us do this and find bugs more conveniently. assert is followed by a condition.

If the condition is true, nothing happens. But if the condition is false, AssertionError is raised.

### async, await

The async and await keywords are provided by the asyncio library in Python. They are used to write concurrent code in Python.

### break, continue

break and continue are used inside for and while loops to alter their normal behaviour.

break will end the smallest loop it is in and control flows to the statement immediately below the loop. continue causes to end the current iteration of the loop, but not the whole loop.

### class

class is used to define a new user-defined class in Python.

### def

def is used to define a user-defined function.

### del

del is used to delete the reference to an object. Everything is object in Python. We can delete a variable reference using del

### if, else, elif

if, else, elif are used for conditional branching or decision making.

When we want to test some condition and execute a block only if the condition is true, then we use if and elif. elif is short for else if. else is the block which is executed if the condition is false.

### except, raise, try

except, raise, try are used with exceptions in Python.

Exceptions are basically errors that suggests something went wrong while executing our program. IOError, ValueError, ZeroDivisionError, ImportError, NameError, TypeError etc. are few examples of exception in Python. try...except blocks are used to catch exceptions in Python.

We can raise an exception explicitly with the raise keyword.

### finally

finally is used with try…except block to close up resources or file streams.

Using finally ensures that the block of code inside it gets executed even if there is an unhandled exception.

### for

for is used for looping. Generally we use for when we know the number of times we want to loop.

### from, import

import keyword is used to import modules into the current namespace. from…import is used to import specific attributes or functions into the current namespace.

### global

global is used to declare that a variable inside the function is global (outside the function).

### in

in is used to test if a sequence (list, tuple, string etc.) contains a value. It returns True if the value is present, else it returns False.

### is

is is used in Python for testing object identity. While the == operator is used to test if two variables are equal or not, is is used to test if the two variables refer to the same object.

It returns True if the objects are identical and False if not.

### lambda

lambda is used to create an anonymous function (function with no name). It is an inline function that does not contain a return statement. It consists of an expression that is evaluated and returned.

### nonlocal

The use of nonlocal keyword is very much similar to the global keyword. nonlocal is used to declare that a variable inside a nested function (function inside a function) is not local to it, meaning it lies in the outer inclosing function. If we need to modify the value of a non-local variable inside a nested function, then we must declare it with nonlocal. Otherwise a local variable with that name is created inside the nested function.

### pass

pass is a null statement in Python. Nothing happens when it is executed. It is used as a placeholder.

### return

return statement is used inside a function to exit it and return a value.

### while

while is used for looping in Python.

The statements inside a while loop continue to execute until the condition for the while loop evaluates to False or a break statement is encountered.

### with

with statement is used to wrap the execution of a block of code within methods defined by the context manager.

Context manager is a class that implements \_\_enter\_\_ and \_\_exit\_\_ methods. Use of with statement ensures that the \_\_exit\_\_ method is called at the end of the nested block. This concept is similar to the use of try…finally block.

### yield

yield is used inside a function like a return statement. But yield returns a generator.

Generator is an iterator that generates one item at a time. A large list of values will take up a lot of memory. Generators are useful in this situation as it generates only one value at a time instead of storing all the values in memory.