Keywords and Identifiers

Python Keywords

Well simply, Python keywords are the words that are reserved. That means you can't use them as name of any entities like variables, classes and functions.

So, you might be thinking what are these keywords for. They are for defining the syntax and structures of Python language.

You should know there are 35 keywords in Python programming language. Although the number can vary in course of time. Also keywords in Python is case sensitive. So, they are to be written as it is. Here is a list of all keywords in python programming.

```
break             for               not
```

If you look at all the keywords and try to figure out all at once, you will be overwhelmed. So, for now, just know these are the keywords. We will learn their uses respectively. You can get the list of python keywords through python shell help.

List of All Python Keywords

and	Logical operator
as	Alias
assert	For debugging
break	Break out of Python loops
class	Used for defining Classes in Python
continue	Keyword used to continue with the Python loop by skipping the existing
def	Keyword used for defining a function
del	Used for deleting objects in Python
elif	Part of the if-elif-else conditional statement in Python
else	Same as above
except	A Python keyword used to catch exceptions
FALSE	Boolean value
finally	This keyword is used to run a code snippet when no exceptions occur
for	Define a Python for loop
from	Used when you need to import only a specific section of a module

Keywords and Identifiers

global	Specify a variable scope as global
if	Used for defining an “if” condition
import	Python keyword used to import modules
in	Checks if specified values are present in an iterable object
is	This keyword is used to test for equality.
lambda	Create anonymous functions
None	The None keyword represents a Null value in PYthon
nonlocal	Declare a variable with non-local scope
not	Logical operator to negate a condition
or	A logical operator used when either one of the conditions needs to be true
pass	This Python keyword passes and lets the function continue further
raise	Raises an exception when called with the specified value
return	Exits a running function and returns the value specified
TRUE	Boolean value
try	Part of the try…except statement
while	Used for defining a Python while loop
with	Creates a block to make exception handling and file operations easy
yield	Ends a function and returns a generator object

Below is a simple example showing usage of if-else in python program.

```
var = 1

if(var==1):

    print("odd")

else:

    print("even")
```

When we run the above program, Python understands the if-else block because of fixed keywords and syntax and then do the further processing.

Keywords and Identifiers

What are Python Identifiers?

Python Identifier is the name we give to identify a variable, function, class, module or other object. That means whenever we want to give an entity a name, that's called identifier.

Sometimes variable and identifier are often misunderstood as same but they are not. Well for clarity, let's see what is a variable?

What is a Variable in Python?

A variable, as the name indicates is something whose value is changeable over time. In fact, a variable is a memory location where a value can be stored. Later we can retrieve the value to use. But for doing it we need to give a nickname to that memory location so that we can refer to it. That's identifier, the nickname.

Rules for Writing Identifiers

There are some rules for writing Identifiers. But first you must know Python is case sensitive. That means **Name** and **name** are two different identifiers in Python. Here are some rules for writing Identifiers in python.

- Identifiers can be combination of uppercase and lowercase letters, digits or an underscore (_). So **myVariable**, **variable_1**, **variable_for_print** all are valid python identifiers.
- An Identifier cannot start with digit. So, while **variable1** is valid, **1variable** is not valid.
- We can't use special symbols like,! #, @, %, $ etc. in our Identifier.
- Identifier can be of any length.

Though these are hard rules for writing identifiers, also there are some naming conventions which are not mandatory but rather good practices to follow.

- Class names start with an uppercase letter. All other identifiers start with a lowercase letter.
- Starting an identifier with a single leading underscore indicates the identifier is private.
- If the identifier starts and ends with two underscores, then means the identifier is language-defined special name.

Keywords and Identifiers

- While **c = 10** is valid, writing **count = 10** would make more sense and it would be easier to figure out what it does even when you look at your code after a long time.
- Multiple words can be separated using an underscore, for example **this_is_a_variable**.

Here's a sample program for python variables.

```
myVariable="hello world"
print(myVariable)

var1=1
print(var1)

var2=2
print(var2)
```