Computer Technology Engineering Department

Lecture: (1)

Variables and Operators (Variables and Operators)

Subject: Object oriented programming II
Class: Second
Lecturer: Baqer Kareem

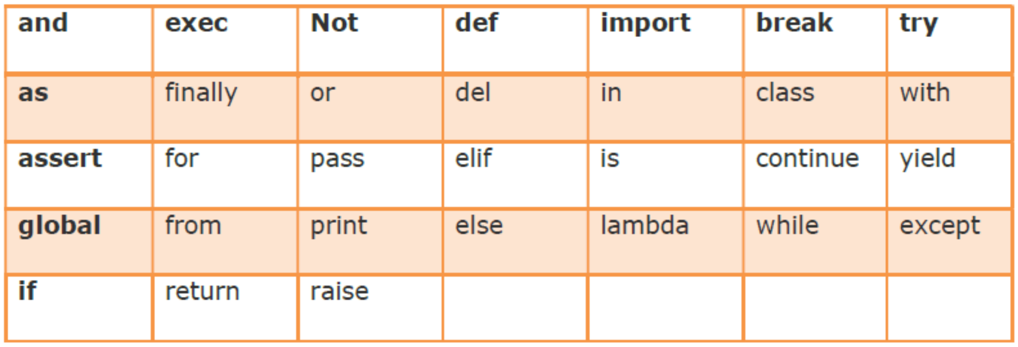
Python IDE shell

IDE (Integrated Development Environment) is a graphical user interface for doing Python development, and is a standard and free part of the Python system.. Python shell allows us to use Python in interactive mode and waits for a command from the user, executes it and returns the result. Usually termed a REPL which stands for "Read - Execute - Print - Loop". To download the Python interpreter (Python shell) go to the site http://www.python.org/downloads and click on the suitable icon.

Python does not allow punctuation characters such as @, $, and % within identifiers. Python is a case sensitive programming language. Thus, Manpowerand manpower are two different identifiers in Python.

Reserved Words:

The following list shows the Python keywords. These are reserved words and you cannot use them as constant or variable or any other identifier names. All the Python keywords contain lowercase letters only.



and	**exec**	**Not**	**def**	**import**	**break**	**try**
as	finally	or	del	in	class	with
assert	for	pass	elif	is	continue	yield
global	from	print	else	lambda	while	except
if	return	raise				

Table1. Table of reserved words

Lines and Indentation

Python doesn't use braces ({}) to indicate blocks of code for class and function definitions or flow control. Blocks of code are denoted by line indentation, which is rigidly enforced. The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount. For

Example (1):

```
if True:
   print("True")
else:
    print("False")
```

Your First Program

• # Prints the words Hello Python

print("Hello Python")

print("It's nice learning Python")

print("Python is easy to learn")

• In the above program ,the lines appear after the # sign are treated as

comments. They are ignored and not executed by Python interpreter. This

will help only to improve the readability of the program.

Hello Python

It's nice learning Python

Python is easy to learn.

Variables and Operators

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory. Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

For example (3), to store the age of the user, we can write a variable userAge and it is defined as below.

userAge = 0 or userAge = 50

After we define userAge ,the program will allocate some memory to store this data. Afterwards this variable can be accessed by referring its name userAge. Every time we declare a variable, some initial value must be given to it. We can also define multiple variables at a time.

For ex(4): userAge, userName=30,'Peter' // This is same as userAge=30 userName='Peter.'

In the above example userAge=30,the '=' sign is known as Assignment Sign, It means ,we are assigning the value on the right side of the = sign to the variable on the left. So the statements x=y and y=x have different meanings in programming.

For ex(5), open your IDLE editor and type the following program.

```
ageX = 5
ageY = 10
ageX = ageY

print("ageX =", ageX)
print('ageY =', ageY)
```

The output should be as below.

```
ageX =10 , ageY =10
```

Here in the program you have assigned x=y in the third line. So the value of x is changed to10 while the value of y doesn't change.

Data Types

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them. Python data types:
• Integer, like 2 ,-5 , 27 ,0 ,1375 etc..
• Float, like 1.2467 ; -0.7865 ;
• String, "Hello World"
• List ["Hello World"]

Ex(6):
userAge = 23 (Integer) // userWeight= 75.50 (Float)
userName='Ahmad' (String) // userAge = "23" (String)

Multiple substrings can be combined by using the concatenate sign(+)
Ex(7): "Ahmad" + "Ali"is equivalent to the string "AhmadAli"

```
print ("Ahmad" + "Ali")
```

EX (8)

```
name1 = "Ahmad"
name2 = "Ali"

full_name = name1 + name2

print(full_name)
```

Some precautions:

- The user name can contain any letters (lower case or upper case) numbers or underscores(-) but the first character shouldn't be a number.

- For Ex: userName,user_Name,user_Name2 etc.. Are valid.
- But 2user_name ,2userName ..are not valid
- Variable names are case sensitive .username and userNAME are not same.

List:

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type. The values stored in a list can be accessed using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (*) is the repetition operator. For Example(9) :

```
list = [ 'abcd', 786 , 2.23, 'Ali', 70.2 ]
addlist = [123, 'Ahmad']
print (list) # Prints complete list
print (list[0]) # Prints first element of the list
print (list[1:3]) # Prints elements starting from 2nd till 3rd
print (list[2:]) # Prints elements starting from 3rd element
print (addlist * 2) # Prints list two times
print (list + addlist) # Prints concatenated lists
```

Output:

```
[abcd', 786, 2.23, 'Ali', 70.2']
abcd
[2.23 ,786]
[Ali', 70.2' ,2.23]
['Ahmad', 123, 'Ahmad' ,123]
['abcd', 786, 2.23, 'Ali', 70.2, 123, 'Ahmad']
```

Basic Operators

Python accepts all the basic operators like:
- + (addition)
- - (subtraction),
- * (multiplication),
- / (division),
- // (floor division),
- % (modulus)
- ** (exponent)
- +=: x+= 2 is same as equal to x=x+2 ,, Similarly for subtraction:

x-=2 is same as x=x-2 .

For ex(10) ,if x= 7,y=2:
• addition: x+y=9 ,, subtraction: x-y=5 ,, Multiplication: x*y=14 ,,
Division: x/y=3.5 ..
• Floor division: x // y=3 (rounds off the answer to the nearest whole number)
• Modulus: x%y= 1 (Gives the remainder when 7 is divided by 2
• Exponent : x**y=49 (7 to the power of 2).

Comparison Operators

These operators compare the values on either sides of them and decide the relation among them. They are also called Relational operators.
• "equals" (= =)
• "not equals" (!=)
• "smaller than"(<)
• "greater than"(>)
• "greater than or equal to (>=)
Example(11): 6 = = 6 ---- equals
8!= 4 ----Not equals
6>=2 ----Greater than or equal to
7>2 ---Greater than
4<9 – Less than or equal to

Logical Operators

There are three logical operators in Python:
• And (The and operator returns True if all the conditions are met ,else it will return False).
• Or (The or operator returns true if atleast one condition is met. Else it will return false).
• Not (The not operator returns True if the condition after the not keyword is false.Else it will return False).