```
In [1]: import sys
        import keyword
        import operator
        from datetime import datetime
        import os
```

Keywords

Keywords are the reserved words in Python and can't be used as an identifier

```
In [4]: print(keyword.kwlist) # List all Python Keywords
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'clas
s', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from',
'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass',
'raise', 'return', 'try', 'while', 'with', 'yield']
```

```
In [8]: len(keyword.kwlist) # List all Python Keywords
```

```
Out[8]: 35
```

Identifiers

An identifier is a name given to entities like class, functions, variables, etc. It helps to differentiate one entity from another.

```
In [11]: 1var = 10 # Identifier can't start with a digit
```

```
  Cell In[11], line 1
    1var = 10 # Identifier can't start with a digit
    ^
SyntaxError: invalid decimal literal
```

```
In [13]: val2@ = 35 # Identifier can't use special symbols
```

```
  Cell In[13], line 1
    val2@ = 35 # Identifier can't use special symbols
          ^
SyntaxError: invalid syntax
```

```
In [15]: import = 125 # Keywords can't be used as identifiers
```

```
  Cell In[15], line 1
    import = 125 # Keywords can't be used as identifiers
           ^
SyntaxError: invalid syntax
```

```
In [17]: """
         Correct way of defining an identifier
         (Identifiers can be a combination of letters in lowercase (a to z) or uppercase
         val2 = 10
```

In [19]:
```
val_ = 99
```

Comments in Python

Comments can be used to explain the code for more readabilty.

In [22]:
```
# Single line comment
val1 = 10
```

In [26]:
```
# Multiple
# line
# comment
val1 = 10
```

In [ ]:
```
'''
Multiple line
comment '''
val1 = 10
```

In [28]:
```
"""
Multiple line
comment """
val1 = 10
```

Statements

Instructions that a Python interpreter can execute.

In [31]:
```
p = 20  # Creates an integer object with value 20 and assigns the variable p to
q = 20  # Create new reference q which will point to value 20. p & q will be poi
r = q   # variable r will also point to the same location where p & q are pointi
p , type(p), hex(id(p)) # Variable P is pointing to memory location '0x7fff6d71a
```

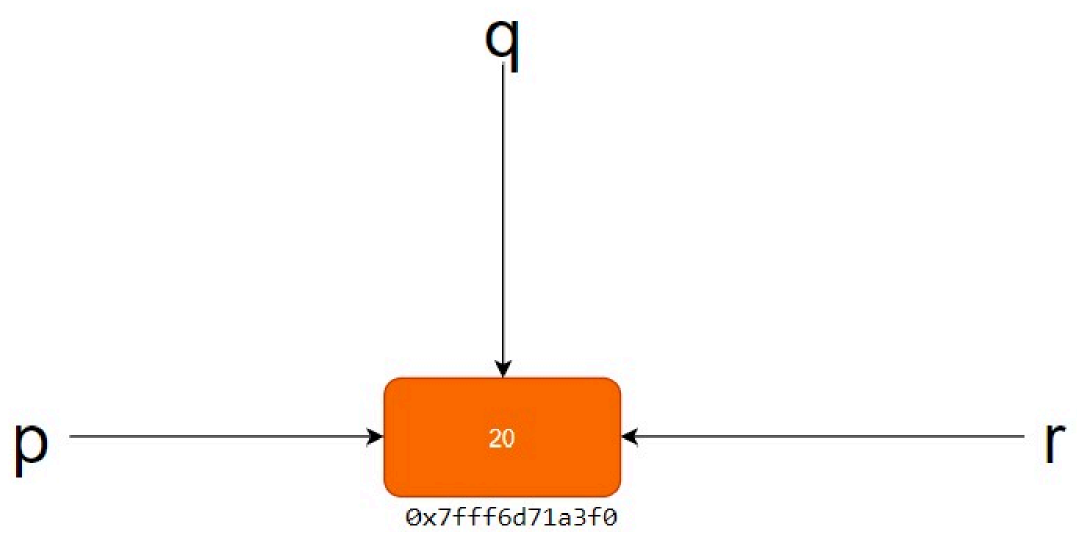
Out[31]:
```
(20, int, '0x7ffd7fb82c18')
```

In [33]:
```
q , type(q), hex(id(q))
```

Out[33]:
```
(20, int, '0x7ffd7fb82c18')
```

In [35]:
```
r , type(r), hex(id(r))
```

Out[35]:
```
(20, int, '0x7ffd7fb82c18')
```



In [39]:
```python
p = 20
p = p + 10 # Variable Overwriting
p
```

Out[39]:
```
30
```

Variable Assigment

In [48]:
```python
intvar = 10 # Integer variable
floatvar = 2.57 # Float Variable
strvar = "Python Language" # String variable
print(intvar)
print(floatvar)
print(strvar)
```

```
10
2.57
Python Language
```

Multiple Assignments

In [53]:
```python
intvar , floatvar , strvar = 10,2.57,"Python Language" # Using commas to separat
print(intvar)
print(floatvar)
print(strvar)
```

```
10
2.57
Python Language
```

In [55]:
```python
p1 = p2 = p3 = p4 = 44 # All variables pointing to same value
print(p1,p2,p3,p4)
```

```
44 44 44 44
```

Data Types

Numeric

In [58]:
```
val1 = 10 # Integer data type
print(val1)
print(type(val1)) # type of object
print(sys.getsizeof(val1)) # size of integer object in bytes
print(val1, " is Integer?", isinstance(val1, int)) # val1 is an instance of int
```

```
10
<class 'int'>
28
10  is Integer? True
```

In [60]:
```
val2 = 92.78 # Float data type
print(val2)
print(type(val2)) # type of object
print(sys.getsizeof(val2)) # size of float object in bytes
print(val2, " is float?", isinstance(val2, float)) # Val2 is an instance of floa
```

```
92.78
<class 'float'>
24
92.78  is float? True
```

In [62]:
```
val3 = 25 + 10j # Complex data type
print(val3)
print(type(val3)) # type of object
print(sys.getsizeof(val3)) # size of float object in bytes
print(val3, " is complex?", isinstance(val3, complex)) # val3 is an instance of
```

```
(25+10j)
<class 'complex'>
32
(25+10j)  is complex? True
```

In [64]:
```
sys.getsizeof(int()) # size of integer object in bytes
```

Out[64]:
```
28
```

In [66]:
```
sys.getsizeof(float()) # size of float object in bytes
```

Out[66]:
```
24
```

In [68]:
```
sys.getsizeof(complex()) # size of complex object in bytes
```

Out[68]:
```
32
```

Boolean

Boolean data type can have only two possible values true or false.

In [71]:
```
bool1 = True
```

In [73]:
```
bool2 = False
```

In [75]:
```
print(type(bool1))
```

```
<class 'bool'>
```

In [77]:
```
print(type(bool2))
```

```
<class 'bool'>
```

In [79]:
```
isinstance(bool1, bool)
```

Out[79]:
```
True
```

In [81]:
```
bool(0)
```

Out[81]:
```
False
```

In [83]:
```
bool(1)
```

Out[83]:
```
True
```

In [85]:
```
bool(None)
```

Out[85]:
```
False
```

In [87]:
```
bool (False)
```

Out[87]:
```
False
```

Strings

String Creation

In [92]:
```
str1 = "HELLO PYTHON"
print(str1)
```

```
HELLO PYTHON
```

In [94]:
```
mystr = 'Hello World' # Define string using single quotes
print(mystr)
```

```
Hello World
```

In [96]:
```
mystr = "Hello World" # Define string using double quotes
print(mystr)
```

```
Hello World
```

In [100…
```
mystr = '''Hello
                World ''' # Define string using triple quotes
print(mystr)
```

```
Hello
                World
```

In [102…
```
mystr = """Hello
                World""" # Define string using triple quotes
print(mystr)
```

```
Hello
                World
```

In [106…
```
mystr = ('Happy    '
         'Monday '
         'Everyone')
print(mystr)
```

```
Happy    Monday Everyone
```

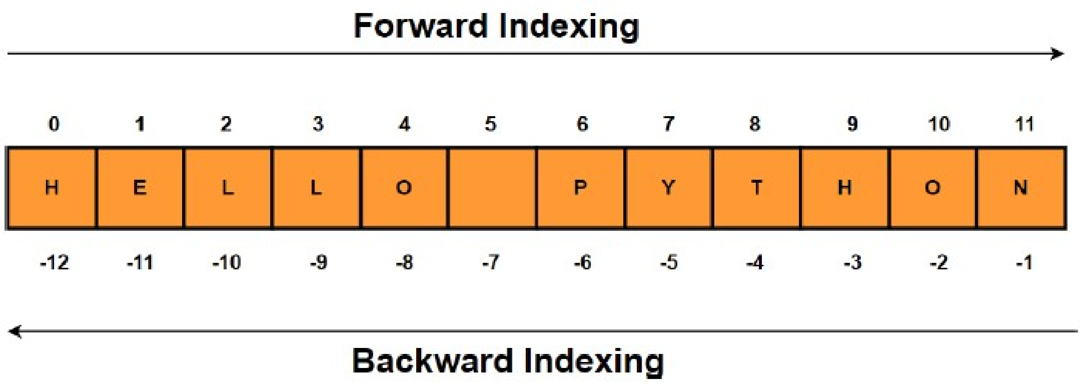
In [108…
```
mystr2 = 'Woohoo '
mystr2 = mystr2*5
mystr2
```

Out[108…
```
'Woohoo Woohoo Woohoo Woohoo Woohoo '
```

In [110…
```
len(mystr2) # Length of string
```

Out[110…
```
35
```

String Indexing



In [114…
```
str1
```

Out[114…
```
'HELLO PYTHON'
```

In [116…
```
str1[0] # First character in string "str1"
```

Out[116…
```
'H'
```

In [118…
```
str1[len(str1)-1] # Last character in string using len function
```

Out[118…
```
'N'
```

In [120…
```
str1[-1] # Last character in string
```

Out[120…
```
'N'
```

In [122…
```
str1[6] #Fetch 7th element of the string
```

Out[122…
```
'P'
```

In [124…
```
str1[5]
```

Out[124…
```
' '
```

String Slicing

In [127…
```
str1[0:5] # String slicing - Fetch all characters from 0 to 5 index location exc
```
Out[127…
```
'HELLO'
```
In [129…
```
str1[6:12] # String slicing - Retreive all characters between 6 - 12 index loc e
```
Out[129…
```
'PYTHON'
```
In [131…
```
str1[-4:] # Retreive last four characters of the string
```
Out[131…
```
'THON'
```
In [133…
```
str1[-6:] # Retreive last six characters of the string
```
Out[133…
```
'PYTHON'
```
In [135…
```
str1[:4] # Retreive first four characters of the string
```
Out[135…
```
'HELL'
```
In [137…
```
str1[:6] # Retreive first six characters of the string
```
Out[137…
```
'HELLO '
```

Update & Delete String

In [140…
```
str1
```
Out[140…
```
'HELLO PYTHON'
```
In [142…
```
#Strings are immutable which means elements of a string cannot be changed once t
str1[0:5] = 'HOLAA'
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[142], line 2
      1 #Strings are immutable which means elements of a string cannot be changed
once t
----> 2 str1[0:5] = 'HOLAA'

TypeError: 'str' object does not support item assignment
```
In [144…
```
del str1 # Delete a string
print(srt1)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[144], line 2
      1 del str1 # Delete a string
----> 2 print(srt1)

NameError: name 'srt1' is not defined
```

In [146…

```
del str1 # Delete a string
print(str1)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[146], line 1
----> 1 del str1 # Delete a string
      2 print(str1)

NameError: name 'str1' is not defined
```

In [150…

```
str1 = "HELLO PYTHON"
print(str1)
```

```
HELLO PYTHON
```

In [152…

```
str1 = "HELLO PYTHON"
del str1 # Delete a string
```

In [154…

```
print(str1)  # String is deleted thus we got an error
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[154], line 1
----> 1 print(str1)

NameError: name 'str1' is not defined
```

String concatenation

In [157…

```
# String concatenation
s1 = "Hello"
s2 = "Asif"
s3 = s1 + s2
print(s3)
```

```
HelloAsif
```

In [ ]: