welcome

welcome

shift+enter=run the cell

dd=delete the cell

Keywords

In [6]:

```
import keyword
print(keyword.kwlist)
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'brea
k', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finall
y', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonloc
al', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yiel
d']
```

In [8]:

```
false=10
```

In [9]:

```
false
```

Out[9]:

```
10
```

identifier

name given to entity like class,functions,variables etc

In [10]:

```
1var=20
```

```
  Input In [10]
    1var=20
    ^
SyntaxError: invalid decimal literal
```

In [11]:

```
var1=20
```

In [12]:

```
var1
```

Out[12]:

```
20
```

In [13]:

```
var@=30
```

```
---------------------------------------------------------------------------
-
NameError                                 Traceback (most recent call las
t)
Input In [13], in <cell line: 1>()
----> 1 var@=30

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

In [14]:

```
var_=30
```

In [15]:

```
var_
```

Out[15]:

```
30
```

In [16]:

```
finally=40
```

```
  Input In [16]
    finally=40
    ^
SyntaxError: invalid syntax
```

In [17]:

```
Finally=40
```

In [18]:

```
Finally
```

Out[18]:

```
40
```

In [19]:

```
from=50
```

```
  Input In [19]
    from=50
         ^
SyntaxError: invalid syntax
```

In [20]:

```
From=50
```

In [21]:

```
From
```

Out[21]:

```
50
```

comments in python :used to explain the code for more readablity

In [22]:

```
print('python')    #define python
```

```
python
```

In [23]:

```
#single line comment
#multi line comment
```

In [24]:

```
"this
is python
session""
```

Out[24]:

```
'this is\npython\nsession'
```

In [25]:

```
'''welcome
to
the
india'''
```

Out[25]:

```
'welcome\nto\nthe\nindia'
```

statement

In [26]:

```
val5=10
```

In [27]:

```
p1=20+30
p1
```

Out[27]:

```
50
```

In [42]:

```
p2=20+30\
+40+50\
+70+80
p2
```

```
  Input In [42]
    p2=20+30\
              ^
SyntaxError: unexpected character after line continuation character
```

In [41]:

```
p2=20+30\
+40+50\
+70+80
p2
```

Out[41]:

```
290
```

indentation

In [46]:

```
x=10
if x==10:
    print('x is equal to 10')
```

```
x is equal to 10
```

docstrings

In [47]:

```
def square(num):
    """square function will return the square of a number"""
    return num**2
```

In [48]:

```
square(2)
```

Out[48]:

```
4
```

In [49]:

```
square.__doc__
```

Out[49]:

```
'square function will return the square of a number'
```

variables

In [50]:

```
a=5
```

In [51]:

```
a
```

Out[51]:

```
5
```

In [57]:

```
p=50
q=25
r=q
```

In [59]:

```
print(id(p))
```

```
2229501953808
```

In [60]:

```
print(id(q))
```

```
2229501953008
```

data type

int float bool/boolean complex

In [65]:

```
a=1254789631422
```

In [66]:

```
type(a)
```

Out[66]:

```
int
```

In [71]:

```
b=10.2
b
```

Out[71]:

```
10.2
```

In [68]:

```
type(b)
```

Out[68]:

```
float
```

In [69]:

```
bool1=True
```

In [70]:

```
bool1
```

Out[70]:

```
True
```

In [78]:

```
print(bool(-2))
```

```
True
```

In [82]:

```
x=2+3j
```

In [83]:

```
type(x)
```

Out[83]:

```
complex
```

Strings

In [84]:

```
str1="hello"
```

In [86]:

```
type(str1)
```

Out[86]:

```
str
```

In [87]:

```
len(str1)
```

Out[87]:

```
5
```

In [94]:

```
str2=" hello  python "
```

In [99]:

```
str2[0]
```

Out[99]:

```
' '
```

In [95]:

```
len(str2)
```

Out[95]:

```
15
```

indexing

In [96]:

```
str3='welcome'
```

in python we always start indexing from 0

In [97]:

```
str3[0]
```

Out[97]:

```
'w'
```

In [98]:

```
str3[-7]
```

Out[98]:

```
'w'
```

Slicing

In [100]:

```
str3[3:6]
```

Out[100]:

```
'com'
```

In [101]:

```
str3[3:7]
```

Out[101]:

```
'come'
```

In [104]:

```
str3[0:]
```

Out[104]:

```
'welcome'
```

In [103]:

```
str3[3:]
```

Out[103]:

```
'come'
```

In [105]:

```
str4="bangalore"
```

In [106]:

```
str4[2:5]
```

Out[106]:

```
'nga'
```

In [112]:

```
str4[3:]
```

Out[112]:

```
'galore'
```

In [108]:

```
str5='data science'
```

In [109]:

```
str5[5:]
```

Out[109]:

```
'science'
```

In [113]:

```
str5[2:8]
```

Out[113]:

```
'ta sci'
```

In [114]:

```
str6='heloo'
```

In [115]:

```
str[3]='l'
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Input In [115], in <cell line: 1>()
----> 1 str[3]='l'

TypeError: 'type' object does not support item assignment
```

In [116]:

```
del str6
```

In [117]:

```
str6
```

```
---------------------------------------------------------------------------
-
NameError                                 Traceback (most recent call las
t)
Input In [117], in <cell line: 1>()
----> 1 str6

NameError: name 'str6' is not defined
```

string concatenation

In [118]:

```
s1='data'
s2='science'  #data science
```

In [128]:

```
print(s1,s2,"dvnksdvnvdkndvn")
```

```
data science dvnksdvnvdkndvn
```

In [122]:

```
print(s1+'-'+s2)
```

```
data-science
```

String Membership

In [129]:

```
mystr='hello everyone'    #in,not in membership operator
```

In [130]:

```
print('hello' in mystr)
```

```
True
```

In [131]:

```
print('python' in mystr)
```

```
False
```

In [132]:

```
print(' eve' in mystr)
```

```
True
```

In [134]:

```
print('-eve' in mystr)
```

```
False
```

string partitioning

In [137]:

```
mystr1='natural language with python and R and java'
mystr1
```

Out[137]:

```
'natural language with python and R and java'
```

In [136]:

```
l=mystr1.partition('with')
l
```

Out[136]:

```
('natural language ', 'with', ' python and R and java')
```

In [138]:

```
mystr1.capitalize()
```

Out[138]:

```
'Natural language with python and r and java'
```

In [139]:

```
mystr1.upper()
```

Out[139]:

```
'NATURAL LANGUAGE WITH PYTHON AND R AND JAVA'
```

In [140]:

```
mystr1.lower()
```

Out[140]:

```
'natural language with python and r and java'
```

In [141]:

```
mystr1.count('a')
```

Out[141]:

```
8
```

In [142]:

```
mystr2="   hello   "
mystr2
```

Out[142]:

```
'   hello   '
```

In [144]:

```
mystr2.strip()
```

Out[144]:

```
'hello'
```

In [145]:

```
mystr2.rstrip()
```

Out[145]:

```
'   hello'
```

In [146]:

```
mystr2.lstrip()
```

Out[146]:

```
'hello   '
```

In [ ]:

```

```