Python Data Structures and Boolean

Boolean
Boolean and Logical Operators
lists
comparision
Dictionaries
Tuples
sets

Boolean variables

Boolean values are the two constant objects false and true
They are used to represent truth values(other values can also be considered false or true).
In numeric contexts (for examples, when used as the argument to an arithmetic operator),they behave like the integers 0 and 1 respectively

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

Out[1]:
```
False
```

In [3]:
```
False
```

Out[3]:
```
False
```

In [4]:
```
True
```

Out[4]:
```
True
```

In [5]:
```
print(True,False)
```

```
True False
```

In [7]:
```
type(True)   ## true capitallatter
```

Out[7]:
```
bool
```

In [45]:
```
my_str='Desh1Deepak'
```

In [46]:
```
my_str.isalnum()
```

Out[46]:
```
True
```

In [47]:
```
print(my_str.isalnum()) # check if all the character are numbers
print(my_str.isalpha()) # check if all char in the string are alphabetic
print(my_str.isdigit()) # test if string contains digits
print(my_str.istitle()) # test if string contains title words
print(my_str.isupper()) # test if string contains upper case
print(my_str.islower()) # test if string contains lower case
print(my_str.isspace()) # test if string contains spaces
print(my_str.endswith('k'))  # test if string contains with k
print(my_str.startswith('d')) # test if string contains startswith D
```

```
True
False
False
True
False
False
False
True
False
```

Boolean and logical Operators

AND , OR

In [48]:
```
True and True
```

Out[48]:
```
True
```

In [49]:
```
True and False
```

Out[49]:
```
False
```

In [50]:
```
False and False
```

Out[50]:
```
False
```

In [51]:
```
True or True
```

Out[51]:
```
True
```

In [52]:
```
True or False
```

Out[52]:
```
True
```

In [53]:
```
False or False
```

Out[53]:
```
False
```

In [54]:
```
str_example='hello world'
my_str='deepak'
```

In [58]:
```
my_str.isalpha() or str_example.isnum()
```

Out[58]:
```
True
```

list

A list is a data structure in python that is a mutable, or changeable,ordered sequence of elements. Each element or value that is inside of a list is called an item.Just as strings are defined as characters between quotes ,list are defined having a values between square brackets[]

In [59]:
```
type([])
```

Out[59]:
```
list
```

In [60]:
```
list_example=[]
```

In [61]:
```
type(list_example)
```

Out[61]:
```
list
```

In [63]:
```
lst=list()
```

In [65]:
```
type(lst)
```

Out[65]:
```
list
```

In [74]:
```
lst=['mathematics','chemistry',100,200,300,400]
```

In [75]:
```
len(lst)
```

Out[75]:
```
6
```

In [77]:
```
type(lst)
```

Out[77]:
```
list
```

In [76]:
```
len(lst)
```

Out[76]:
```
6
```

In [ ]:
```

```

Append

In [78]:
```
# append is used to add elements in the list
lst.append("desh")
```

In [79]:
```
lst
```

Out[79]:
```
['mathematics', 'chemistry', 100, 200, 300, 400, 'desh']
```

In [82]:
```
## indexing in list
lst[3]
```

Out[82]:
```
200
```

In [83]:
```
lst[:]
```

Out[83]:
```
['mathematics', 'chemistry', 100, 200, 300, 400, 'desh']
```

In [84]:
```
lst[1:3]
```

Out[84]:
```
['chemistry', 100]
```

In [85]:
```
lst[1:]
```

Out[85]:
```
['chemistry', 100, 200, 300, 400, 'desh']
```

In [86]:
```
lst[1:6]
```

Out[86]:
```
['chemistry', 100, 200, 300, 400]
```

In [89]:
```
lst[1:4]
```

Out[89]:
```
['chemistry', 100, 200]
```

In [91]:
```
lst.append(["verma","deepak"])
```

In [92]:
```
lst
```

Out[92]:
```
['mathematics', 'chemistry', 100, 200, 300, 400, 'desh', ['verma', 'deepak']]
```

Insert

In [93]:
```
## insert in specific order
lst.insert(2,"radhe")
```

In [94]:
```
lst
```

Out[94]:
```
['mathematics',
 'chemistry',
 'radhe',
 100,
 200,
 300,
 400,
 'desh',
 ['verma', 'deepak']]
```

In [97]:
```
lst.append(["hello world"])
```

In [98]:
```
lst
```

Out[98]:
```
['mathematics',
 'chemistry',
 'radhe',
 100,
 200,
 300,
 400,
 'desh',
 ['verma', 'deepak'],
 'hello world',
 ['hello world']]
```

Extend method

In [111…
```
lst=[1,2,3,4,5,6]
```

In [112…
```
lst
```

Out[112]:
```
[1, 2, 3, 4, 5, 6]
```

In [113…
```
lst.extend([8,9])
```

In [114…
```
lst
```

Out[114]:
```
[1, 2, 3, 4, 5, 6, 8, 9]
```

Various operation that we can perform in list

In [135…
```
lst``
```

Out[135]:
```
[1, 1, 2, 3, 4, 5]
```

In [136…
```
sum(lst)
```

Out[136]:
```
16
```

In [138…
```
lst*3 # list 3 times appended
```

Out[138]:
```
[1, 1, 2, 3, 4, 5, 1, 1, 2, 3, 4, 5, 1, 1, 2, 3, 4, 5]
```

Pop() Method

In [118…
```
# remove the last elements from list .pop()
# remove the first elements from list .pop(0)
lst.pop()
```

Out[118]:
```
9
```

In [119…
```
lst
```

Out[119]:
```
[1, 2, 3, 4, 5, 6, 8]
```

In [120…
```
lst.pop(0)
```

Out[120]:
```
1
```

In [121…
```
lst
```

Out[121]:
```
[2, 3, 4, 5, 6, 8]
```

In [122…
```
lst.pop(2)
```

Out[122]:
```
4
```

In [123…
```
lst
```

Out[123]:
```
[2, 3, 5, 6, 8]
```

count(): calculates total occurenceof given element of list

In [124…
```
lst=[1,1,2,3,4,5]
lst.count(1)
```

Out[124]:
```
2
```

In [125…
```
# length calculate the total length of list
len(lst)
```

Out[125]:
```
6
```

In [132…
```
# index(): returns the index of first occurence,start and End index are not necessa
lst.index(3,1,4)  # first 1 is value and 1,4 is index
```

Out[132]:
```
3
```

In [127…
```
## min and max
min(lst)
```

Out[127]:
```
1
```

In [128…
```
max(lst)
```

Out[128]:
```
5
```

SET