lists

>a list is a collection of characters variable, and

>number variables and boolean values datatypes

>a list is to share multiple data with in a single variables

>a list is a ordered type of data

>a list is denoted as []

>a list item denoted with double quotations

syntax:

items=["item1","item2","item3"]

print(items)

In [1]:

```
# example for the lists
li=["apple","mango","orange","banana","grapes"]
li
```

Out[1]:

```
['apple', 'mango', 'orange', 'banana', 'grapes']
```

In [4]:

```
# type of the list
print(type(li))
```

```
<class 'list'>
```

In [5]:

```
# length of the list
print(len(li))
```

```
5
```

In [6]:

```
print(li[-1])
```

```
grapes
```

In [7]:

```
# accessing the item in a list or not
if "apple" in li:
    print("yes")
else:
    print("no")
```

```
yes
```

In [9]:

```
# how to change item from the list

li
```

Out[9]:

```
['apple', 'mango', 'orange', 'banana', 'grapes']
```

In [12]:

```
li[0]="pineapple"
li
```

Out[12]:

```
['pineapple', 'mango', 'orange', 'banana', 'grapes']
```

In [13]:

```
li.insert(1,"kiwi")
li
```

Out[13]:

```
['pineapple', 'kiwi', 'mango', 'orange', 'banana', 'grapes']
```

In [14]:

```
li1=["asma","samrana","husna","amir"]
li1
```

Out[14]:

```
['asma', 'samrana', 'husna', 'amir']
```

In [15]:

```
li[2:5]
```

Out[15]:

```
['mango', 'orange', 'banana']
```

In [16]:

```
li[2:]
```

Out[16]:

```
['mango', 'orange', 'banana', 'grapes']
```

In [17]:

```
li[:4]
```

Out[17]:

```
['pineapple', 'kiwi', 'mango', 'orange']
```

In [21]:

```
li+li1
```

Out[21]:

```
['pineapple',
 'kiwi',
 'mango',
 'orange',
 'banana',
 'grapes',
 'asma',
 'samrana',
 'husna',
 'amir']
```

In [36]:

```
li1.remove("amir")
li1
```

Out[36]:

```
['asma', 'samrana', 'husna']
```

In [29]:

```
li.pop(3)
```

Out[29]:

```
'banana'
```

In [34]:

```
del li[1]
li
```

Out[34]:

```
['pineapple', 'orange', 'grapes']
```

In [37]:

```
li
```

Out[37]:

```
['pineapple', 'orange', 'grapes']
```

In [38]:

```
li1
```

Out[38]:

```
['asma', 'samrana', 'husna']
```

In [39]:

```
li1.clear()
```

In [40]:

```
li1
```

Out[40]:

```
[]
```

In [43]:

```
li.sort()
li
```

Out[43]:

```
['grapes', 'orange', 'pineapple']
```

In [46]:

```
# list using loop
for i in li:
    print(i)
```

```
grapes
orange
pineapple
```

In [47]:

```
for i in li:
    print(i,end=" ")
```

```
grapes orange pineapple
```

tuples

it is a collection of different types of data

it is immutable(can't change)

operated using()

In [ ]:

```
## to create the empty tuple
## syntax: tuple_name=()
## to create single values
## syntax: tuple_name=(values)
## to create multiple values
## syntax: tuple_name=(value1,value2,....)
```

In [75]:

```
t1=(34,58,77)
t1
```

Out[75]:

```
(34, 58, 77)
```

In [81]:

```
t2=(10)
print(type(t2))
t3=(20)
print(type(t3))
```

```
<class 'int'>
<class 'int'>
```

In [82]:

```
t2
```

Out[82]:

```
10
```

In [83]:

```
t3
```

Out[83]:

```
20
```

In [84]:

```
# how to access the values from the tuple
t1
print(t1[2])
```

```
77
```

In [85]:

```
t1
print(t1[0:2])
```

```
(34, 58)
```

In [86]:

```
t2=(10,20,30,10,20,30,20,20,30,10)
# to count the numbers of occurence
t2.count(10)
```

Out[86]:

```
3
```

In [87]:

```
t2=(10,20,30,10,20,30,20,20,30,10)
# to count the numbers of occurence
t2.count(20)
```

Out[87]:

```
4
```

In [88]:

```
t2=(10,20,30,10,20,30,20,20,30,10)
# to count the numbers of occurence
t2.count(30)
```

Out[88]:

```
3
```

In [89]:

```
# index
t2.index(20)
```

Out[89]:

```
1
```

In [90]:

```
t2.index(10)
```

Out[90]:

```
0
```

In [91]:

```
t2.index(30)
```

Out[91]:

```
2
```

In [96]:

```
tuple1= ("asma",58,"true",77,"mohammad")
print(tuple1)
```

```
('asma', 58, 'true', 77, 'mohammad')
```

In [74]:

```
# examples of tuples
tup=("python","sql","java","7","10","c++")
tup
```

Out[74]:

```
('python', 'sql', 'java', '7', '10', 'c++')
```

In [49]:

```
# type of the list
print(type(tup))
```

```
<class 'tuple'>
```

In [50]:

```
# length of the list
print(len(tup))
```

```
5
```

In [51]:

```
print(tup[-1])
```

```
c++
```

In [52]:

```
# accessing the values from the tuple
if "python" in tup:
    print("yes")
else:
    print("no")
```

```
yes
```

In [53]:

```
# how to change item from the tuple
tup
```

Out[53]:

```
('python', 'sql', 'java', 'c', 'c++')
```

In [59]:

```
tup1=("asma","shruthi","bhargavi")
tup1
```

Out[59]:

```
('asma', 'shruthi', 'bhargavi')
```

In [60]:

```
tup+tup1
```

Out[60]:

```
('python', 'sql', 'java', 'c', 'c++', 'asma', 'shruthi', 'bhargavi')
```

In [69]:

```
for i in tup:
    print(i)
```

```
python
sql
java
c
c++
```

In [70]:

```
for i in tup:
    print(i,end=" ")
```

```
python sql java c c++
```

dictionary:

it is a collection of different data types

it is a group of key and values(key:value)->item

in dictionary keys are unique

written in({})

each and every item is separated with commas(,)

accessing dictionary values by using key names

it is muttable(changable)

In [1]:

```
# to create a empty dictionary:
# -dictionary_names
d1={34,58,77}
d1
```

Out[1]:

```
{34, 58, 77}
```

In [5]:

```
# to create dictionary values:
# dictionary_name={key:value,key:value2,......}
d2={'a':100,'b':200,'c':300}
print(d2)
print(type(d2))
```

```
{'a': 100, 'b': 200, 'c': 300}
<class 'dict'>
```

In [4]:

```
# to create a dictionary with different data types
d4={'a':58,'name':'asma','branch':'civil','b':77.0}
print(d4)
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil', 'b': 77.0}
```

In [5]:

```
# accessing the dictionary values using the key names
print(d4['name'])
print(d4['b'])
print(d4['branch'])
```

```
asma
77.0
civil
```

In [6]:

```
# update the dictionary values
print(d4)
d4['branch']='civil'
print(d4)
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil', 'b': 77.0}
{'a': 58, 'name': 'asma', 'branch': 'civil', 'b': 77.0}
```

In [2]:

```
print(dir(dict))
```

```
['__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__do
c__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__',
'__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__',
'__len__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__
repr__', '__reversed__', '__setattr__', '__setitem__', '__sizeof__', '__str_
_', '__subclasshook__', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys',
'pop', 'popitem', 'setdefault', 'update', 'values']
```

In [7]:

```
# pop
print(d4)
print(d4.pop('b'))
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil', 'b': 77.0}
77.0
```

In [8]:

```
# keys
print(d4)
print(d4.keys())
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil'}
dict_keys(['a', 'name', 'branch'])
```

In [9]:

```
# values
print(d4)
print(d4.values())
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil'}
dict_values([58, 'asma', 'civil'])
```

In [10]:

```
# items
print(d4)
print(d4.items())
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil'}
dict_items([('a', 58), ('name', 'asma'), ('branch', 'civil')])
```

In [13]:

```
# copy
print(d4)
d3=d4.copy()
print(d4)
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil'}
{'a': 58, 'name': 'asma', 'branch': 'civil'}
```

In [12]:

```
# get
print(d4)
print(d4.get('a'))
print(d4.get('name'))
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil'}
58
asma
```

In [14]:

```
# set default
print(d4)
print(d4.setdefault('rollno',77))
print(d4)
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil'}
77
{'a': 58, 'name': 'asma', 'branch': 'civil', 'rollno': 77}
```

In [16]:

```
# pop item
print(d4)
print(d4.popitem())
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil', 'rollno': 77}
('rollno', 77)
```

In [17]:

```
print(d4)
print(d4.popitem())
```

```
{'a': 58, 'name': 'asma', 'branch': 'civil'}
('branch', 'civil')
```

In [18]:

```
print(d4)
print(d4.popitem())
```

```
{'a': 58, 'name': 'asma'}
('name', 'asma')
```

In [20]:

```
# clear
print(d4)
print(d4.clear())
```

```
{}
None
```

```
In [ ]:
```