Compound_DT

May 5, 2022

1 Lists

1.1 Create Lists

————————————————————————-

Ordered collection of elements which are mutable

————————————————————————-

[3]:
```
a=[1,2,3,4]
print(a)
inp = eval(input())
print(type(inp))
```

```
 [2,3,4,5]
<class 'list'>
```

[4]:
```
a=[3,"abcd",75.5] # store heterogenous elements
```

[5]:
```
print(id(a)) # Identity of object
```

```
2757939961600
```

1.2 Copying Lists

[10]:
```
b=a # only reference is copied
print(id(b))
# another instance of entire list is created
b=a[:]
print(id(b))
b=a.copy()
print(id(b))
```

```
2757939961600
2757939709632
2757939960448
```

1.3 Built-in Functions with lists

[1]:
```
a=[1,2,3,4]
print(len(a)) # length of list
print(min(a)) # min element in the list
print(max(a)) # max element in the list
print(sum(a)) #sum of all elements in the list
```

```
4
1
4
10
```

[14]:
```
a=[1,2.5,3,6.7]
print(len(a)) # length of list
print(min(a)) # min element in the list
print(max(a)) # max element in the list
print(sum(a)) #sum of all elements in the list
```

```
4
1
6.7
13.2
```

1.4 Operators

[2]:
```
a=[1,2,3,4]
b=[3,4,5]

print(a+b) # creates a new list by appending all elements in b to a
print(a*2) # creates a new list by repeating it twice
print(3 in a) # checks whether an element is present in the list
```

```
[1, 2, 3, 4, 3, 4, 5]
[1, 2, 3, 4, 1, 2, 3, 4]
True
```

1.5 Indexing and Slicing

[3]:
```
print(a[1])
print(a[-1])
print(a[5])
```

```
2
4
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
```

```
~\AppData\Local\Temp/ipykernel_14532/2229650636.py in <module>
      1 print(a[1])
      2 print(a[-1])
----> 3 print(a[5])

IndexError: list index out of range
```

[4]:
```
print(a[1:3])
print(a[1:3:2])
print(a[::-1])
```

```
[2, 3]
[2]
[4, 3, 2, 1]
```

1.6 Updating lists in-place

[5]:
```
# using assignment operator individual elements can be modified
a=[1,2,3,4]
a[3] = 5
print(a)
a[5] = 6
```

```
[1, 2, 3, 5]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_14532/4214328974.py in <module>
      3 a[3] = 5
      4 print(a)
----> 5 a[5] = 6

IndexError: list assignment index out of range
```

1.7 Comparing Lists

[6]:
```
# Comparing lists
a=[1,2,3,4]
print(id(a))
b=[1,2,3,4]
print(id(b))
```

```
2116718624576
2116717882048
```

[7]:
```
if(a==b):
    print("YES")
else:
    print("NO")
```

```
YES
```

1.8 Packing and unpacking

[44]:
```
x,y,*t = [1,2,3,4,5,6]
print(x)
print(y)
```

```
1
2
```

[45]:
```
*t,x,y = [1,2,3,4,5,6]
print(x)
print(y)
```

```
5
6
```

[46]:
```
x,*t,y = [1,2,3,4,5,6]
print(x)
print(y)
```

```
1
6
```

1.9 Traversing the elements in the list

[11]:
```
L = [1,2,3,4]
for i in L: # It iterates over elements in the list
    print(i)
```

```
1
2
3
4
```

[12]:
```
L=[1,2,3,4]
# find len and use positional index to access elements
for i in range(len(L)):
    print(L[i])
```

```
1
2
```

```
3
4
```

[13]:
```
#Use enumerate method over list
# this returns the index and element as a tuple
for (ind,ele) in enumerate(L):
    print(ind,ele)
```

```
0 1
1 2
2 3
3 4
```

1.10 Delete elements in list

[16]:
```
L=[1,2,3,4]
del L[2]
print(L)
del L[1:2]
print(L)
del L
print(L)
```

```
[1, 2, 4]
[1, 4]
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_14532/717409545.py in <module>
      5 print(L)
      6 del L
----> 7 print(L)

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

1.11 Member Functions

1.11.1 Search

[18]:
```
L=[1,2,3,4]
#-----------Count----------------------------------------#
print(L.count(1)) # If element is found, returns its count
print(L.count(5)) # if element is not found returns 0

#-----------index----------------------------------------#
#return lowest index of the element
```

```
print(L.index(1)) # If element is found, returns its lowest index (multiple␣
 ↪occurrences)
print(L.index(5)) # if element is not found raises exception
```

```
1
0
0
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_14532/4162976962.py in <module>
      8 #return lowest index of the element
      9 print(L.index(1)) # If element is found, returns its lowest index␣
 ↪(multiple occurrences)
---> 10 print(L.index(5)) # if element is not found raises exception

ValueError: 5 is not in list
```

1.11.2 Insert elements

[20]:
```
L.append(5) # insert the element to list
print(L)
L.extend([1,2,3]) # insert all elements in the iterable to the list
print(L)
L.insert(0,8) # index, value
print(L)
```

```
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5, 1, 2, 3]
[8, 1, 2, 3, 4, 5, 1, 2, 3]
```

1.11.3 Delete elements

[22]:
```
L.remove(5) # deletes the first occurence of the element from list
print(L)
```

```
[8, 1, 2, 3, 4, 1, 2, 3]
```

[23]:
```
L.remove(3)
print(L)
```

```
[8, 1, 2, 4, 1, 2, 3]
```

[24]:
```
L.pop() # removes last element
L.pop(3) #delets the element at index 3
```

[24]:
```
4
```

[25]:
```
print(L)
```

```
[8, 1, 2, 1, 2]
```

1.11.4 Utilities

[26]:
```
L.sort() #sort elements in ascending order
print(L)
```

```
[1, 1, 2, 2, 8]
```

[27]:
```
L.sort(reverse=True)
print(L)
```

```
[8, 2, 2, 1, 1]
```

[28]:
```
L=[2,1,5,6]
L.reverse()
print(L)
```

```
[6, 5, 1, 2]
```

1.12 List Comprehension

[29]:
```
#Compose a new list
#[op for ___ in ______]

L = [1 for i in range(10)]
print(L)
```

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```

[30]:
```
#apply operation to each element in a list
L=[1,2,3,4]
#find a new list where each element is 2^element
L1 = [2**i for i in L]
print(L1)
```

```
[2, 4, 8, 16]
```

[31]:
```
#filter (or) remove elements in the list based on condition
#[i for i in L if <condition>]

L1=[i for i in L if i>3]
print(L1)
```

```
[4]
```

1.13 Two dimensional Lists

[13]:
```
# rows, columns
# list of lists
# each list will be a row
L = [[1,2,3],[4,5,6],[7,8,9]]
print(L)
print(L[0])
print(L[0][0])
```

```
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
[1, 2, 3]
1
```

[16]:
```
#using index
for i in range(len(L)):
    for j in range(len(L[i])):
        print(L[i][j], end=' ')
    print('')
```

```
1 2 3
4 5 6
7 8 9
```

[20]:
```
#using iterable
for i in L:
    for j in i:
        print(j, end=' ')
    print('')
```

```
1 2 3
4 5 6
7 8 9
```

[19]:
```
#Flattening lists
L1 = [j for r in L for j in r]
print(L1)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

2 Tuples

2.1 Create Tuple

————————————————————————-

Ordered collection of elements which are immutable

————————————————————————-

[32]:
```
tup = (1,2,3)
print(len(tup))
tup=()
print(type(tup))
print(len(tup))
tup=(3,) # Singleton tuple
print(type(tup))
print(len(tup))
```

```
3
<class 'tuple'>
0
<class 'tuple'>
1
```

[21]:
```
# While creating singleton tuple, if comma is not used after the number it is␣
 ↪treated as
# integer
tup=(3)
print(type(tup))
```

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

2.2 Immutability

[41]:
```
tup[1] = 5
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_22676/1511130136.py in <module>
----> 1 tup[1] = 5

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

2.3 Uses

Returning values from function

Using collection of elements as keys in dictionary

2.4 Operators / Indexing and slicing (similar to list)

2.5 Functions (count and index)

3 Dictionaries

3.1 Create dictionary

[180]:
```
empty_dict = {}
person_dict = {'name':'rani','dept':'it','designation':'AP'}
person_dict = dict([('name','rani'),('dept','it'),('designation','AP')])
```

3.2 Access elements in dictionary

[181]:
```
#<variable>[<key>]
person_dict['dept']

print(person_dict.get('hello')) # return None
person_dict['hello']
```

```
None
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_3568/2504083514.py in <module>
      3
      4 print(person_dict.get('hello')) # return None
----> 5 person_dict['hello']

KeyError: 'hello'
```

3.3 Add, Modify, Delete

[182]:
```
person_dict['name'] = 'raju' # modify assignment
print(person_dict)
person_dict['salary'] = 10000 #add element
print(person_dict)
del person_dict['name'] #delete element
print(person_dict)
```

```
{'name': 'raju', 'dept': 'it', 'designation': 'AP'}
{'name': 'raju', 'dept': 'it', 'designation': 'AP', 'salary': 10000}
{'dept': 'it', 'designation': 'AP', 'salary': 10000}
```

3.4 Traverse dictionary

[183]:
```
print(list(person_dict.items()))
print(list(person_dict.values()))
print(list(person_dict.keys()))
```

```
[('dept', 'it'), ('designation', 'AP'), ('salary', 10000)]
['it', 'AP', 10000]
['dept', 'designation', 'salary']
```

3.5 Update dictionary

[184]:
```
person_dict.update({'dept':'cse','salary':20000, 'name':'raju'})
print(person_dict)
```

```
{'dept': 'cse', 'designation': 'AP', 'salary': 20000, 'name': 'raju'}
```

[185]:
```
x={'k1':1,'k2':2,'k3':3}
x.setdefault('k4') # adds this key with default value as None
print(x)
```

```
{'k1': 1, 'k2': 2, 'k3': 3, 'k4': None}
```

3.6 Delete elements in dictionary

[186]:
```
print(person_dict.popitem())
```

```
('name', 'raju')
```

[187]:
```
print(person_dict)
print(person_dict.pop('dept')) # delete a particular key
print(person_dict)
```

```
{'dept': 'cse', 'designation': 'AP', 'salary': 20000}
cse
{'designation': 'AP', 'salary': 20000}
```

3.7 Dictionary Comprehension

[188]:
```
dict1 = {'A':1, 'B':2, 'C':3}
dict2 = {k:v+1 for (k,v) in dict1.items()}
print(dict2)
```

```
{'A': 2, 'B': 3, 'C': 4}
```

3.8 in operator

[190]:
```
print('A' in dict2)
```

```
True
```

4 Sets

[191]:
```
s = {1,2,3,4} # creation of sets
print(s)
```

```
{1, 2, 3, 4}
```

4.1 Insert, Delete

[193]:
```
# add elements
s.add(5)
print(s)
#remove elements
s.remove(4)
s.discard(8)
s.pop()
```

```
{1, 2, 3, 4, 5}
```

[193]:
```
1
```

4.2 Set operations

[195]:
```
S1={1,2,3,4}
S2 = {1,5,6,8}

print(S1.union(S2)) #  | symbol
print(S1.intersection(S2)) # & symbol
print(S1.difference(S2)) # - symbol
print(S1.symmetric_difference(S2)) # ^ symbol

print(S1.isdisjoint(S2))
```

```
{1, 2, 3, 4, 5, 6, 8}
{1}
{2, 3, 4}
{2, 3, 4, 5, 6, 8}
False
```