python-data-structure-assignment

January 8, 2023

Python Data Structures

1 Types of Data Structures:

- `List` | `Tuple`
- `Set` | `Dictionary`

Data Structures type	Mutable	Comments	Indexing	Ordered	Duplicacy
tuple ()	immutable	immutable version of list	possible	yes	allowed
list []	mutable	-	possible	yes	allowed
set {}	mutable	-	not	no	not
dict {key:value}	mutable	-	possible	no	not

`immutable` => `can't be changed` `mutable` => `can be changed`

[ ]:

1.1 Lists

1.1.1 Create an empty list with the name 'a', print the value of a and type(a).

[1]:
```
# create empty list, name it 'a'
a=[]
```

[2]:
```
# print the value of a

print(a)
```

```
[]
```

[3]:
```
# print the type of a
type(a)
```

[3]: `list`

1.1.2 Create a list , languages = [‘R’,‘Python’, ‘SAS’, ‘Scala’, 42],

[4]:
```
lis=['R','Python','SAS','Scala',42]
```

Print the number of elements in the list

[5]:
```
len(lis)
```

[5]: 5

Using for loop iterate and print all the elements in the list

[6]:
```
for item in lis:
  print(item)
```

```
R
Python
SAS
Scala
42
```

Select the second item, ‘Python’ and store it in a new variable named ‘temp’

[7]:
```
temp=lis[1]
```

Print the value of temp and type(temp)

[8]:
```
print(temp)
```

```
Python
```

Append the element ‘Java’ in the list

[9]:
```
lis.append('Java')
print(lis)
```

```
['R', 'Python', 'SAS', 'Scala', 42, 'Java']
```

Remove the element 42 from the list and print the list

[10]:
```
lis.remove(42)
print(lis)
```

```
['R', 'Python', 'SAS', 'Scala', 'Java']
```

1.1.3 Create a list, colors = [‘Red’, ‘Blue’, ‘White’]

[11]:
```
colors=['Red','Blue','White']
```

Append the element ‘Black’ to colors

[12]:
```
colors.append('Black')
```

Append the color 'Orange' to second position (index=1) and print the list

[13]:
```
colors.insert(2,'Orange')
```

Print the list

[14]:
```
print(colors)
```

```
['Red', 'Blue', 'Orange', 'White', 'Black']
```

Create another list, colors2 = ['Grey', 'Sky Blue']

[15]:
```
colors2=['Grey','Sky Blue']
```

Add the elements of colors2 to colors using extend function in the list

[16]:
```
colors.extend(colors2)
```

Print len of colors and its elements

[17]:
```
len(colors)
print(colors)
```

```
['Red', 'Blue', 'Orange', 'White', 'Black', 'Grey', 'Sky Blue']
```

Sort the list and print it.

[18]:
```
colors.sort()
print(colors)
```

```
['Black', 'Blue', 'Grey', 'Orange', 'Red', 'Sky Blue', 'White']
```

1.1.4 Create a string, sent = 'Coronavirus Caused Lockdowns Around The World."

[19]:
```
str="Coronavirus Caused Lockdowns Around The World."
```

Use split function to convert the string into a list of words and save it in variable words and print the same

[20]:
```
mdlist=[]
words=str.split(" ")
mdlist.append(words)
print(mdlist)
print(words)
```

```
[['Coronavirus', 'Caused', 'Lockdowns', 'Around', 'The', 'World.']]
['Coronavirus', 'Caused', 'Lockdowns', 'Around', 'The', 'World.']
```

Convert each word in the list to lower case and store it in variable words_lower. Print words_lower

[21]:
```
words_lower=[]
for word in (words):
  lower=word.lower()
  words_lower.append(lower)
print(words_lower)
print(lower)
```

```
['coronavirus', 'caused', 'lockdowns', 'around', 'the', 'world.']
world.
```

Check whether 'country' is in the list

[22]:
```
'country' in mdlist
'country' in words
```

[22]:
```
False
```

Remove the element 'the' from the list and print the list.

[23]:
```
#mdlist.remove('the')
words.remove('The')
print(words)
```

```
['Coronavirus', 'Caused', 'Lockdowns', 'Around', 'World.']
```

Select the first 4 words from the list words_lower using slicing and store them in a new variable x4

[24]:
```
x4=words_lower[:4]
```

[25]:
```
# print x4

print(x4)
```

```
['coronavirus', 'caused', 'lockdowns', 'around']
```

Convert the list of elements to single string using join function and print it

[26]:
```
joined=" ".join(x4)
print(joined)
```

```
coronavirus caused lockdowns around
```

1.2 Sets

1.2.1 Create stud_grades = ['A','A','B','C','C','F']

[27]:
```
stud_grades=["A","A","B","C","C","F"]
```

Print the len of stud_grades

[28]:
```
len(stud_grades)
```

[28]: 6

Create a new variable, stud_grades_set = set(stud_grades)

[29]:
```
stud_grades_set=set(stud_grades)
```

Print stud_grades_set.

[30]:
```
print(stud_grades_set)
```

```
{'C', 'F', 'B', 'A'}
```

print the type of stud_grades and stud_grades_set and print their corresponding elements. Try to understand the difference between them.

[31]:
```
type(stud_grades)
type(stud_grades_set)
for item in stud_grades:
  print(item)
for item in stud_grades_set:
  print(item)
```

```
A
A
B
C
C
F
C
F
B
A
```

[32]:
```
type(stud_grades_set)
```

[32]: set

[33]:
```
type(stud_grades)
```

[33]: list

Add a new element 'G' to stud_grades_set

[34]:
```
stud_grades_set.add('G')
```

Add element 'F' to stud_grades_set. and print it.

[36]:
```
stud_grades_set.add('F')
```

```
print(stud_grades_set)
```

```
{'A', 'C', 'G', 'F', 'B'}
```

!!Did you notice? set doesn't add an element if it's already present in it, unlike lists.

Remove 'F' from stud_grades_set

[37]:
```
stud_grades_set.remove('F')
print(stud_grades_set)
```

```
{'A', 'C', 'G', 'B'}
```

Print the elements and the length of stud_grades_set

[38]:
```
print(stud_grades_set)
len(stud_grades_set)
```

```
{'A', 'C', 'G', 'B'}
```

[38]: 4

1.2.2 Create colors = ['red','blue','orange'], and fruits = ['orange','grapes','apples']

[39]:
```
colors=['red','blue','orange']
fruits=['orange','grapes','apples']
```

Print color and fruits

[40]:
```
print(colors)
print(fruits)
```

```
['red', 'blue', 'orange']
['orange', 'grapes', 'apples']
```

Create colors_set, and fruits_set. (using set()) and print them

[41]:
```
colors_set=set(colors)
fruits_set=set(fruits)
print(colors_set)
print(fruits_set)
```

```
{'red', 'orange', 'blue'}
{'orange', 'grapes', 'apples'}
```

Find the union of both the sets.

[42]:
```
z=colors_set.union(fruits_set)
print(z)
```

```
{'orange', 'blue', 'red', 'grapes', 'apples'}
```

Find the intersection of both the sets

[43]:
```
z=colors_set.intersection(fruits_set)
print(z)
```

```
{'orange'}
```

Find the elements which are Fruits but not colors (using set.difference())

[44]:
```
z=fruits_set.difference(colors_set)
print(z)
```

```
{'grapes', 'apples'}
```

[31]:
```

```

1.3 TUPLES

1.3.1 Create temp = [17, 'Virat', 50.0]

[45]:
```
temp=[17,'Virat',50.0]
```

Iterate through temp and print all the items in temp

[46]:
```
for item in temp:
  print(item)
```

```
17
Virat
50.0
```

replace first element with 11 in temp

[47]:
```
temp[0]=11
print(temp)
```

```
[11, 'Virat', 50.0]
```

Set temp1 = tuple(temp)

[48]:
```
temp1=tuple(temp)
```

Iterate through temp1 and print all the items in temp1.

[50]:
```
for item in temp1:
  print(item)
```

```
11
Virat
50.0
```

replace first element with 17 in temp1

[51]:
```
temp1[0]=17
print(temp1) #gives error as tuple is immutable.
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-51-abc19f5d2b96> in <module>
----> 1 temp1[0]=17
      2 print(temp1) #gives error as tuple is immutable.

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

Oops!! You got an error. Hey Don't worry! Its because Once a tuple is created, you cannot change its values unlike list.

1.3.2 Create city = ("Bangalore", 28.9949521, 72)

[52]:
```
city=("Bangalore",28.9949521,72)
```

Print first element of city

[53]:
```
city[0]
```

[53]:
```
'Bangalore'
```

Create city2 = ('Chennai', 30.01, 74)

[54]:
```
city2=('Chennai',30.01,74)
```

Create cities which consist of city and city2

[55]:
```
cities=city+city2 #merging 2 tuples
```

Print cities

[56]:
```
print(cities)
```

```
('Bangalore', 28.9949521, 72, 'Chennai', 30.01, 74)
```

Print type of first element in cities

[57]:
```
type(cities[0])
```

[57]:
```
str
```

print the type of cities

[58]:
```
type(cities)
```

[58]: tuple

Hey that implies you made a nested tuples!!

1.4 DICTIONARY

1.4.1 Create a dictionary d = {“actor”:“amir”,“animal”:“cat”,“earth”:2,“list”:[23,32,12]}

[59]:
```
d={"actor":"amir","animal":"cat","earth":2,"list":[23,32,12]}
```

Print the value of d[0]

[60]:
```
d[0]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-60-123a9cc6df61> in <module>
----> 1 d[0]

KeyError: 0
```

Oops!! again an error. again a fun fact. Dictionary return the value for key if key is in the dictionary, else throws KeyError and we don’t have key 0 here :(

Store the value of d[‘actor’] to a new variable actor.

[61]:
```
actor=d['actor']
print(actor)
```

```
amir
```

Print the type of actor

[62]:
```
type(actor)
```

[62]: str

Store the value of d[‘list’] in new variable List.

[63]:
```
list=d['list']
print(list)
```

```
[23, 32, 12]
```

Print the type of List.

[64]:
```
type(list)
```

[64]: list

Create d1 = { ‘singer’ : ‘Kr$na’ , ‘album’: ‘Still here’, ‘genre’ : ‘hip-hop’}

[65]:
```
d1={'singer':'Kr$na','album':'Still here','genre':'hip-hop'}
```

Merge d1 into d.

[66]:
```
d.update(d1)
```

print d

[67]:
```
print(d)
```

```
{'actor': 'amir', 'animal': 'cat', 'earth': 2, 'list': [23, 32, 12], 'singer':
'Kr$na', 'album': 'Still here', 'genre': 'hip-hop'}
```

Print all the keys in d

[68]:
```
d.keys()
```

[68]:
```
dict_keys(['actor', 'animal', 'earth', 'list', 'singer', 'album', 'genre'])
```

Print all the values in d

[69]:
```
d.values()
```

[69]:
```
dict_values(['amir', 'cat', 2, [23, 32, 12], 'Kr$na', 'Still here', 'hip-hop'])
```

Iterate over d, and print each key, value pair as this - (actor —-> amir)

[70]:
```
for key,value in d.items():
  print(key,'---->',value)
```

```
actor ----> amir
animal ----> cat
earth ----> 2
list ----> [23, 32, 12]
singer ----> Kr$na
album ----> Still here
genre ----> hip-hop
```

count the number of occurences of charachters in string named “sent” using dictionary and print the same.

[71]:
```
sent=input()
dict={}
for i in sent:
  dict[i]=sent.count(i)
print(dict)
```

```
abcdefghijklmnoprqstuvwxyz
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 1, 'f': 1, 'g': 1, 'h': 1, 'i': 1, 'j': 1,
```

```
'k': 1, 'l': 1, 'm': 1, 'n': 1, 'o': 1, 'p': 1, 'r': 1, 'q': 1, 's': 1, 't': 1,
'u': 1, 'v': 1, 'w': 1, 'x': 1, 'y': 1, 'z': 1}
```

[73]:
```
#for words in string.
string=input()
words=[]
words=string.split()
mydict={}
for key in words:
  mydict[key]=words.count(key)
print(mydict)
```

```
hello akshay
{'hello': 1, 'akshay': 1}
```

[ ]: