Practical-4

June 29, 2021

1 List

1.0.1 A list in Python is used to store the sequence of various types of data.

1.0.2 Python lists are mutable type its mean we can modify its element after it created.

1.0.3 A list can be defined as a collection of values or items of different types.

1.0.4 The items in the list are separated with the comma (,) and enclosed with the square brackets [].

[1]:
```
thislist = ["apple", "banana", "cherry"]
print(thislist)
```

```
['apple', 'banana', 'cherry']
```

[2]:
```
print(len(thislist))
```

```
3
```

[3]:
```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "mango"]
print(len(thislist))
```

```
6
```

[4]:
```
thislist[5]="pineapple"
thislist[1:3]=["blackcurrant", "watermelon"]
print(thislist)
```

```
['apple', 'blackcurrant', 'watermelon', 'orange', 'kiwi', 'pineapple']
```

[5]:
```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "mango"]
thislist.insert(1,"hi")
print(thislist)
```

```
['apple', 'hi', 'banana', 'cherry', 'orange', 'kiwi', 'mango']
```

[6]:
```
thislist = ["apple", "banana", "cherry"]
thislist.append("orange")
```

```
print(thislist)
```

```
['apple', 'banana', 'cherry', 'orange']
```

[7]:
```
thislist = ["apple", "banana", "cherry"]
tropical = ["mango", "pineapple", "papaya"]
thislist.extend(tropical)
print(thislist)
```

```
['apple', 'banana', 'cherry', 'mango', 'pineapple', 'papaya']
```

2 Tuple

2.1 Tuples are used to store multiple items in a single variable

2.2 A tuple is a collection which is ordered and unchangeable.

[8]:
```
thistuple = ("apple", "banana", "cherry")
print(thistuple)
```

```
('apple', 'banana', 'cherry')
```

[9]:
```
print(len(thistuple))
```

```
3
```

[10]:
```
print(thistuple[-1])
print(thistuple[-2])
```

```
cherry
banana
```

[11]:
```
thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")
print(thistuple[2:5])
```

```
('cherry', 'orange', 'kiwi')
```

2.2.1 Once a tuple is created, you cannot change its values. Tuples are unchangeable, or immutable as it also is called,But there is a workaround. You can convert the tuple into a list, change the list, and convert the list back into a tuple.

[12]:
```
y=list(thistuple)
print(y)
```

```
['apple', 'banana', 'cherry', 'orange', 'kiwi', 'melon', 'mango']
```

[13]:
```
y[1]="pineapple"
print(y)
```

```
['apple', 'pineapple', 'cherry', 'orange', 'kiwi', 'melon', 'mango']
```

[14]:
```
x=tuple(y)
print(x)
```

```
('apple', 'pineapple', 'cherry', 'orange', 'kiwi', 'melon', 'mango')
```

3 Set

3.0.1 Sets are used to store multiple items in a single variable.

3.0.2 A set is a collection which is both unordered and unindexed.

3.0.3 Sets are written with curly brackets.

3.0.4 Sets cannot have two items with the same value!

[15]:
```
thisset = {"apple", "banana", "cherry"}
print(thisset)
```

```
{'banana', 'apple', 'cherry'}
```

[16]:
```
print(len(thisset))
```

```
3
```

[17]:
```
for x in thisset:
    print(x)
```

```
banana
apple
cherry
```

[20]:
```
x={"apple", "banana", "cherry"}
x.add("orange")
print(x)
```

```
{'banana', 'apple', 'orange', 'cherry'}
```

[19]:
```
thisset = {"apple", "banana", "cherry"}
tropical = {"pineapple", "mango", "papaya"}
thisset.update(tropical)

print(thisset)
```

```
{'pineapple', 'banana', 'apple', 'papaya', 'mango', 'cherry'}
```

4 Dictionaries

Dictionaries are used to store data values in key:value pairs.

A dictionary is a collection which is ordered*, changeable and does not allow duplicates.

[22]:
```
new={"No":1, "Name":"Nisha A. Panchal", "Address":"ABC"}
print(new)
```

```
{'No': 1, 'Name': 'Nisha A. Panchal', 'Address': 'ABC'}
```

[24]:
```
new={"No":2, "Name":"Rohith", "Address":"XYZ"}
print(new)
```

```
{'No': 2, 'Name': 'Rohith', 'Address': 'XYZ'}
```

[25]:
```
print(new["Name"])
```

```
Rohith
```

[27]:
```
new={"No":1, "Name":"Nisha A. Panchal", "Address":"ABC", "No":2, "Name":
 ↪"Rohith", "Address":"XYZ"}
print(new)
```

```
{'No': 2, 'Name': 'Rohith', 'Address': 'XYZ'}
```

[33]:
```
new={12:'abc',21:'xyz',31:'pqr'}
print(new)
```

```
{12: 'abc', 21: 'xyz', 31: 'pqr'}
```

[34]:
```
for x in new:
    print(new[x])
```

```
abc
xyz
pqr
```

[41]:
```
my1 = {'one': 'Ram', 'two': 13, 'three': 'Jordge', 'four': 'Gill'};
print(my1)
my1.update({'five':'Kelly'});
print(my1);
```

```
{'one': 'Ram', 'two': 13, 'three': 'Jordge', 'four': 'Gill'}
{'one': 'Ram', 'two': 13, 'three': 'Jordge', 'four': 'Gill', 'five': 'Kelly'}
```

[42]:
```
print("Sr_no\t\t Name")
for x,y in my1.items():
    print(x,"\t\t",y)
```

```
Sr_no            Name
one              Ram
two              13
```

```
three           Jordge
four            Gill
five            Kelly
```

4.1 create a Dictionary using word and meaning format

4.1.1 for example: To adapt : become adjusted to new conditions.

[50]:
```
D1={'to adapt':'become adjusted to new conditions','to alter':'make structural␣
↪changes to (a building)','transform':'make a marked change in the form,␣
↪nature, or appearance of.'}
```

[51]:
```
print(D1)
```

```
{'to adapt': 'become adjusted to new conditions', 'to alter': 'make structural
changes to (a building)', 'transform': 'make a marked change in the form,
nature, or appearance of.'}
```

[55]:
```
i=1
print("sr\t Word \t\t Meaning")
for x,y in D1.items():
    print(i,"\t",x,"\t",y)
    i+=1
```

```
sr      Word            Meaning
1       to adapt        become adjusted to new conditions
2       to alter        make structural changes to (a building)
3       transform       make a marked change in the form, nature, or appearance
of.
```

[ ]:
```

```