list-ds

August 27, 2023

1 List Data Structure

1. List is one among the four data collections in python that is used for storing mulitple values on to a single variable.
2. List is created using square brackets and can contain elements of any type.
3. Characteristics of List:
    1. Ordered
    2. Countable
    3. Mutable
    4. Indexable
    5. Contains duplicate values
    6. Used in places where we need an heterogenous data with mutable property(OLTP-Online Transactional Process)(i.e) Live streaming data.
    7. Empty structure is possible

2 Basic Operation

1. len 2.type 3.check for a sequence 4.accessing(single or a range of elements) 5.Looping(Using items or lenght)

[50]:

```
# List

a=['Jayam_ravi','Vijay','Surya','Vikram','Gautham','Kamal','Rajini']
```

[51]:

```
# Length

len(a)
```

[51]:

```
7
```

[52]:

```
# Type

type(a)
```

[52]:

```
list
```

```
[53]: # Check for a sequence

      if "Surya" in a:
        print("Surya","Jyothika")
      else:
        print("Not there!!")
```

```
Surya Jyothika
```

```
[54]: # Accessing

      a[3]            # Single elements access
```

```
[54]: 'Vikram'
```

```
[55]: a[2:6]            # Range of element access
```

```
[55]: ['Surya', 'Vikram', 'Gautham', 'Kamal']
```

```
[56]: a[-6:-1]                # Negative indexing
```

```
[56]: ['Vijay', 'Surya', 'Vikram', 'Gautham', 'Kamal']
```

```
[57]: # Looping

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

```
Jayam_ravi
Vijay
Surya
Vikram
Gautham
Kamal
Rajini
```

```
[58]: for i in range(len(a)):
        print(a[i])
```

```
Jayam_ravi
Vijay
Surya
Vikram
Gautham
Kamal
Rajini
```

[59]:

```
i=0
while i<len(a):
  if i%2==0:
    print("Hello",a[i])
  else:
    print(a[i])
  i+=1
```

```
Hello Jayam_ravi
Vijay
Hello Surya
Vikram
Hello Gautham
Kamal
Hello Rajini
```

3 Advanced Operation

1. Changing(With replacement) 2.Inserting(Without replacement) 3.Adding 4.Removing 5.Sorting 6.Reversing 7.Copy 8.Join 9.List Comprehension 10.List methods

4 Changing

[60]:

```
# Changing

a[3]='Dhanush'         # Single element change
```

[61]:

```
a[1:5]=['apple','banana','chickoo','rasberry']        # Range of elements␣
  ↪change4`1
```

[62]:

```
# Inserting

a.insert(2,'strawberry')
```

5 Adding

[64]:

```
# Adding

a.append('fruit')
```

[66]:

```
a.append(['fruit','veggie'])
```

[67]:

```
a.extend('bug')
```

[69]:

```
a.extend(['carrot','spinach','kale'])
```

[72]:
```
a.extend(['fruit','apple','Kamal'])
```

[73]:
```
a
```

[73]:
```
['Jayam_ravi',
 'apple',
 'strawberry',
 'banana',
 'chickoo',
 'rasberry',
 'Kamal',
 'Rajini',
 'fruit',
 ['fruit', 'veggie'],
 'b',
 'u',
 'g',
 'carrot',
 'spinach',
 'kale',
 'fruit',
 'apple',
 'Kamal']
```

6 Removing

[74]:
```
# Removing

a.remove('Kamal')
```

[76]:
```
a.pop(3)
```

[76]:
```
'banana'
```

[78]:
```
del a[4]
```

[80]:
```
a.clear()
```

[81]:
```
a
```

[81]:
```
[]
```

7 Sorting

```
[83]: b=['pizza','waffle','chocolate','bombolini','sandwich','pasta','sour_bread','briyani']
```

```
[84]: b.sort()
      b
```

```
[84]: ['bombolini',
       'briyani',
       'chocolate',
       'pasta',
       'pizza',
       'sandwich',
       'sour_bread',
       'waffle']
```

```
[85]: b.sort(reverse=True)
      b
```

```
[85]: ['waffle',
       'sour_bread',
       'sandwich',
       'pizza',
       'pasta',
       'chocolate',
       'briyani',
       'bombolini']
```

8 Reversing

```
[88]: b.reverse()
      b
```

```
[88]: ['bombolini',
       'briyani',
       'chocolate',
       'pasta',
       'pizza',
       'sandwich',
       'sour_bread',
       'waffle']
```

9 Joining

[91]:

```
c=[1,2,3,4]
d=[5,6,7,8]

c.append(0)
```

[93]:

```
c.extend(d)
c
```

[93]:

```
[1, 2, 3, 4, 0, 5, 6, 7, 8]
```

[95]:

```
f=['a','b','c','d']
g=['f','j','h','e']
```

[96]:

```
f+g
```

[96]:

```
['a', 'b', 'c', 'd', 'f', 'j', 'h', 'e']
```

10 Copying

[100]:

```
h=[9,8,3,5]
j=h.copy()
```

[101]:

```
j
```

[101]:

```
[9, 8, 3, 5]
```

11 List Comprehension

Whenever we want to create a new list from the existing one, based on some condition, list comprehension provides a simpler way to do that without the hassle to write many lines of code.

[103]:

```
z=[1,2,3,4,5,6,7,8,9,10]                # Ususal method
v=[]
for i in z:
  if i%2==0:
    v.append(i)
  else:
    continue
print(v)
```

```
[2, 4, 6, 8, 10]
```

[113]:

```
y=[1,2,3,4,5,6,7,8,9,10]
p= [i for i in v if i%2==0]
```

[114]:

```
p
```

[114]:

```
[2, 4, 6, 8, 10]
```

12 List methods

[123]:

```
# Count method

h=[1,2,5,2,2,4,7,6,3,4,7,9]
h.count(2)
```

[123]:

```
3
```

[124]:

```
# Index method

h.index(4)
```

[124]:

```
5
```