Lists in Python

Dr. Seema Gupta Bhol

Python Collections

There are four collection data types in the Python programming language:

- **List** is a collection which is ordered and changeable. Allows duplicate members.
- **Tuple** is a collection which is ordered and unchangeable. Allows duplicate members.
- **Set** is a collection which is unordered, unchangeable, and unindexed. No duplicate members.
- **Dictionary** is a collection which is ordered and changeable. No duplicate members.

List

Lists are used to store multiple items in a single variable.

Lists are created using square brackets

e.g.

```
fruitlist = ["apple", "banana", "cherry"]
print(fruitlist)
```

- List items are ordered, changeable, and allow duplicate values.
- List items are indexed, the first item has index `[0]`, the second item has index `[1]` etc.

```
fruitlist = ["apple", "banana", "cherry", "apple", "cherry"]
print(fruitlist)
```

- Print the second item of the list:

```
mylist = ['apple', 'banana',  'cherry']
print(mylist[1])
```

Negative Indexing

Negative indexing means start from the end

-1 refers to the last item, -2 refers to the second last item etc.

Example

Print the last item of the list:

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

Range of Indexes

One can specify a range of indexes by specifying where to start and where to end the range.

When specifying a range, the return value will be a new list with the specified items.

- Print the third, fourth, and fifth item:

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

The search will start at index 2 (included) and end at index 5 (not included).

- By leaving out the start value, the range will start at the first item:

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[:4])
```

This example returns the items from the beginning to, but NOT including, "kiwi“

- By leaving out the end value, the range will go on to the end of the list:

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[2:]
```

This example returns the items from "cherry" to the end

List Length

List length is number of items in the list.
To determine how many items a list has, use the `len()` function:

Example
Print the number of items in the list:

```
fruitlist = ["apple", "banana", "cherry"]
print(len(fruitlist))
```

List Items - Data Types

List items can be of any data type:

Example

String, int, float,boolean etc. data types:

- list1 = ["apple", "banana", "cherry"]
- list2 = [1, 5, 7, 9, 3]
- list3 = [True, False, False]

A list can contain different data types:

Example

A list with strings, integers and boolean values:

list4 = ["abc", 34, True, 40, “Apple"]

Data type of Lists

What is the data type of a list?

lists are defined as objects with the data type 'list':

```
mylist = ["apple", "banana", "cherry"]
print(type(mylist))
```

Change List Items

- To change the value of a specific item, refer to the index number:

Change the second item:

```
thislist = ["apple", "banana", "cherry"]

thislist[1] = "blackcurrant"

print(thislist)
```

- To change the value of items within a specific range, define a list with the new values, and refer to the range of index numbers where you want to insert the new values:

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "mango"]

thislist[1:3] = ["blackcurrant", "watermelon"]

print(thislist)
```

Specifying more items

If we insert more items than replaced, the new items will be inserted where specified, and the remaining items will move accordingly:

Example

Change the second value by replacing it with two new values:

```
thislist = ["apple", "banana", "cherry"]
thislist[1:2] = ["blackcurrant", "watermelon"]
print(thislist)
```

The length of the list will change when the number of items inserted does not match the number of items replaced.

Specifying less items

If we insert less items than replaced, the new items will be inserted where specified, and the remaining items will move accordingly:

- Change the second and third value by replacing it with one value:

```
thislist = ["apple", "banana", "cherry"]
thislist[1:3] = ["watermelon"]
print(thislist)
```

Insert Items in List

To insert a new list item, without replacing any of the existing values, we can use the insert() method.

The insert() method inserts an item at the specified index:

- Insert "watermelon" as the third item:

```
thislist = ["apple", "banana", "cherry"]
thislist.insert(2, "watermelon")
print(thislist)
```

As a result , the list will now contain 4 items.

Append Items

To add an item to the end of the list, use the append() method:

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

Extend List

To append elements from another list to the current list, use the extend() method.

Add the elements of tropical to thislist:

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

Remove List Items

The remove() method removes the specified item.

Remove "banana“ from the thislist:

```
thislist = ["apple", "banana", "cherry"]
thislist.remove("banana")
print(thislist)
```

If there are more than one item with the specified value, the remove() method removes the first occurrence

Remove Specified Index

- The pop() method removes the specified index.

Example :Remove the second item

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

- If no index is specified, the pop() method removes the last item.

Example :Remove the last item:

```
thislist = ["apple", "banana", "cherry"]
thislist.pop()
print(thislist)
```

del keyword

- The del keyword also removes the specified index:

Example: Remove the first item:

```
thislist = ["apple", "banana", "cherry"]
del thislist[0]
print(thislist)
```

- The del keyword can also delete the list completely.

Example: Delete the entire list:

```
thislist = ["apple", "banana", "cherry"]
del thislist
```

Clear the List

The clear() method empties the list.

The list still remains, but it has no content.

Example: Clear the list content:

```
thislist = ["apple", "banana", "cherry"]
thislist.clear()
print(thislist)
```

Some List functions

- max() : Finds the largest element in a list.
- min() :Finds the smallest element in a list.

Note: All elements must be of the same type (all numbers, or all strings).

```
a = [10, 25, 5, 40]
print(max(a))
print(min(a))
```

- sum() : Calculates the total of all numeric elements in a list.

```
a = [10, 20, 30]
print(sum(a))
```

Note: Works only with numbers not strings.

- count(): Counts how many times a given element appears in a list.

```
a = [1, 2, 2, 3, 2]
print(a.count(2))
```

Exercise

- Q1.Write a Python program to create a list of 5 integers and display the list.
- Q2.Write a Python program to find the length of a list entered by the user.
- Q3.Write a Python program to access and display the first and last elements of a list.
- Q4.Write a Python program to add an element to an existing list using append().
- Q5.Write a Python program to remove a given element from a list.
- Q6. Write a Python program to insert an element at a given position in a list.
- Q7. Write a Python program to compute sum of list elements.
- Q8. Write a Python program to append multiple elements to a list.
- Q9. Write a Python program to clear all elements from a list.
- Q10. Write a program to find maximum, minimum in list of integers.