30 Days Of Python: Day 5 – Lists

Day 5

Lists

There are four collection data types in Python :

- List: is a collection which is ordered and changeable(modifiable). Allows duplicate members.
- Tuple: is a collection which is ordered and unchangeable or unmodifiable(immutable). Allows duplicate members.
- Set: is a collection which is unordered, un-indexed and unmodifiable, but we can add new items to the set. Duplicate members are not allowed.
- Dictionary: is a collection which is unordered, changeable(modifiable) and indexed. No duplicate members.

A list is collection of different data types which is ordered and modifiable(mutable). A list can be empty or it may have different data type items.

How to Create a List

In Python we can create lists in two ways:

- Using list built-in function

```
# syntax
lst = list()
```

```
empty_list = list() # this is an empty list, no item in the list
print(len(empty_list)) # 0
```

- Using square brackets, []

```
# syntax
lst = []
```

```
empty_list = [] # this is an empty list, no item in the list
print(len(empty_list)) # 0
```

Lists with initial values. We use *len()* to find the length of a list.

```
fruits = ['banana', 'orange', 'mango', 'lemon']                     # list of
fruits
vegetables = ['Tomato', 'Potato', 'Cabbage','Onion', 'Carrot']      # list of
vegetables
animal_products = ['milk', 'meat', 'butter', 'yoghurt']             # list of
animal products
web_techs = ['HTML', 'CSS', 'JS', 'React','Redux', 'Node', 'MongDB'] # list of web
technologies
countries = ['Finland', 'Estonia', 'Denmark', 'Sweden', 'Norway']

# Print the lists and its length
print('Fruits:', fruits)
print('Number of fruits:', len(fruits))
print('Vegetables:', vegetables)
print('Number of vegetables:', len(vegetables))
print('Animal products:',animal_products)
print('Number of animal products:', len(animal_products))
print('Web technologies:', web_techs)
print('Number of web technologies:', len(web_techs))
print('Countries:', countries)
print('Number of countries:', len(countries))
```

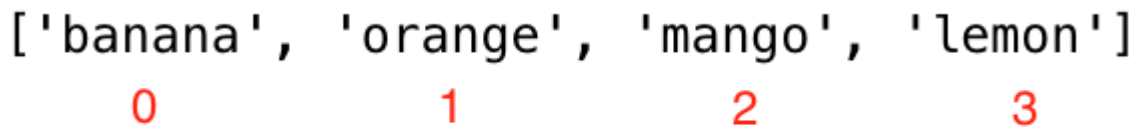
```
output
Fruits: ['banana', 'orange', 'mango', 'lemon']
Number of fruits: 4
Vegetables: ['Tomato', 'Potato', 'Cabbage', 'Onion', 'Carrot']
Number of vegetables: 5
Animal products: ['milk', 'meat', 'butter', 'yoghurt']
Number of animal products: 4
Web technologies: ['HTML', 'CSS', 'JS', 'React', 'Redux', 'Node', 'MongDB']
Number of web technologies: 7
Countries: ['Finland', 'Estonia', 'Denmark', 'Sweden', 'Norway']
Number of countries: 5
```

- Lists can have items of different data types

```
 lst = ['Suniksha', 20, True, {'country':'Finland', 'city':'Helsinki'}] # list containing different data types
```

Accessing List Items Using Positive Indexing

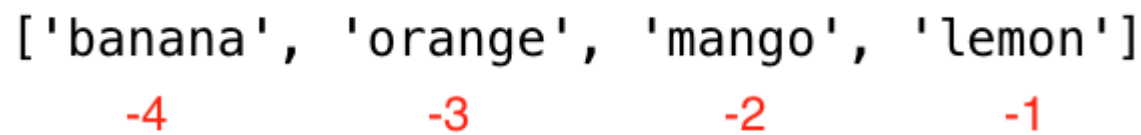
We access each item in a list using their index. A list index starts from 0. The picture below shows clearly where the index
starts




```
fruits = ['banana', 'orange', 'mango', 'lemon']
first_fruit = fruits[0] # we are accessing the first item using its index
print(first_fruit)      # banana
second_fruit = fruits[1]
print(second_fruit)     # orange
last_fruit = fruits[3]
print(last_fruit) # lemon
# Last index
last_index = len(fruits) - 1
last_fruit = fruits[last_index]
```

Accessing List Items Using Negative Indexing

Negative indexing means beginning from the end, -1 refers to the last item, -2 refers to the second last item.




```
fruits = ['banana', 'orange', 'mango', 'lemon']
first_fruit = fruits[-4]
last_fruit = fruits[-1]
second_last = fruits[-2]
print(first_fruit)      # banana
print(last_fruit)       # lemon
print(second_last)      # mango
```

Unpacking List Items

```
lst = ['item1','item2','item3', 'item4', 'item5']
first_item, second_item, third_item, *rest = lst
print(first_item)     # item1
print(second_item)    # item2
print(third_item)     # item3
print(rest)           # ['item4', 'item5']

# First Example
fruits = ['banana', 'orange', 'mango', 'lemon','lime','apple']
first_fruit, second_fruit, third_fruit, *rest = fruits
print(first_fruit)     # banana
print(second_fruit)    # orange
print(third_fruit)     # mango
print(rest)           # ['lemon','lime','apple']
# Second Example about unpacking list
first, second, third,*rest, tenth = [1,2,3,4,5,6,7,8,9,10]
print(first)          # 1
print(second)         # 2
print(third)          # 3
print(rest)           # [4,5,6,7,8,9]
print(tenth)          # 10
# Third Example about unpacking list
countries = ['Germany',
'France','Belgium','Sweden','Denmark','Finland','Norway','Iceland','Estonia']
gr, fr, bg, sw, *scandic, es = countries
print(gr)
print(fr)
print(bg)
print(sw)
print(scandic)
print(es)
```

Slicing Items from a List

- Positive Indexing: We can specify a range of positive indexes by specifying the start, end and step, the return value will be a new list. (default values for start = 0, end = len(lst) - 1 (last item), step = 1)

```
fruits = ['banana', 'orange', 'mango', 'lemon']
all_fruits = fruits[0:4] # it returns all the fruits
# this will also give the same result as the one above
all_fruits = fruits[0:] # if we don't set where to stop it takes all the rest
orange_and_mango = fruits[1:3] # it does not include the first index
orange_mango_lemon = fruits[1:]
orange_and_lemon = fruits[::2] # here we used a 3rd argument, step. It will take
every 2cnd item - ['banana', 'mango']
```

- Negative Indexing: We can specify a range of negative indexes by specifying the start, end and step, the return value will be a new list.

```
fruits = ['banana', 'orange', 'mango', 'lemon']
all_fruits = fruits[-4:] # it returns all the fruits
orange_and_mango = fruits[-3:-1] # it does not include the last index,['orange',
'mango']
orange_mango_lemon = fruits[-3:] # this will give starting from -3 to the
end,['orange', 'mango', 'lemon']
```

```
reverse_fruits = fruits[::-1] # a negative step will take the list in reverse
order,['lemon', 'mango', 'orange', 'banana']
```

Modifying Lists

List is a mutable or modifiable ordered collection of items. Lets modify the fruit list.

```
fruits = ['banana', 'orange', 'mango', 'lemon']
fruits[0] = 'avocado'
print(fruits)       #  ['avocado', 'orange', 'mango', 'lemon']
fruits[1] = 'apple'
print(fruits)       #  ['avocado', 'apple', 'mango', 'lemon']
last_index = len(fruits) - 1
fruits[last_index] = 'lime'
print(fruits)        #  ['avocado', 'apple', 'mango', 'lime']
```

Checking Items in a List

Checking an item if it is a member of a list using *in* operator. See the example below.

```
fruits = ['banana', 'orange', 'mango', 'lemon']
does_exist = 'banana' in fruits
print(does_exist)  # True
does_exist = 'lime' in fruits
print(does_exist)  # False
```

Adding Items to a List

To add item to the end of an existing list we use the method *append()*.

```
# syntax
lst = list()
lst.append(item)
```

```
fruits = ['banana', 'orange', 'mango', 'lemon']
fruits.append('apple')
print(fruits)           # ['banana', 'orange', 'mango', 'lemon', 'apple']
fruits.append('lime')   # ['banana', 'orange', 'mango', 'lemon', 'apple', 'lime']
print(fruits)
```

Inserting Items into a List

We can use *insert()* method to insert a single item at a specified index in a list. Note that other items are shifted to the right. The *insert()* methods takes two arguments:index and an item to insert.

```
# syntax
lst = ['item1', 'item2']
lst.insert(index, item)
```

```
fruits = ['banana', 'orange', 'mango', 'lemon']
fruits.insert(2, 'apple') # insert apple between orange and mango
```

```
print(fruits)           # ['banana', 'orange', 'apple', 'mango', 'lemon']
fruits.insert(3, 'lime')   # ['banana', 'orange', 'apple', 'lime', 'mango',
'lemon']
print(fruits)
```

Removing Items from a List

The remove method removes a specified item from a list

```
# syntax
lst = ['item1', 'item2']
lst.remove(item)
```

```
fruits = ['banana', 'orange', 'mango', 'lemon', 'banana']
fruits.remove('banana')
print(fruits)  # ['orange', 'mango', 'lemon', 'banana'] - this method removes the
first occurrence of the item in the list
fruits.remove('lemon')
print(fruits)  # ['orange', 'mango', 'banana']
```

Removing Items Using Pop

The *pop()* method removes the specified index, (or the last item if index is not specified):

```
# syntax
lst = ['item1', 'item2']
lst.pop()       # last item
lst.pop(index)
```

```
fruits = ['banana', 'orange', 'mango', 'lemon']
fruits.pop()
print(fruits)       # ['banana', 'orange', 'mango']

fruits.pop(0)
print(fruits)       # ['orange', 'mango']
```

Removing Items Using Del

The *del* keyword removes the specified index and it can also be used to delete items within index range. It can also delete the list completely

```
# syntax
lst = ['item1', 'item2']
del lst[index] # only a single item
del lst        # to delete the list completely
```

```
fruits = ['banana', 'orange', 'mango', 'lemon', 'kiwi', 'lime']
del fruits[0]
print(fruits)       # ['orange', 'mango', 'lemon', 'kiwi', 'lime']
del fruits[1]
print(fruits)       # ['orange', 'lemon', 'kiwi', 'lime']
del fruits[1:3]     # this deletes items between given indexes, so it does not
delete the item with index 3!
```

```
print(fruits)       # ['orange', 'lime']
del fruits
print(fruits)       # This should give: NameError: name 'fruits' is not defined
```

Clearing List Items

The *clear()* method empties the list:

```
# syntax
lst = ['item1', 'item2']
lst.clear()
```

```
fruits = ['banana', 'orange', 'mango', 'lemon']
fruits.clear()
print(fruits)       # []
```

Copying a List

It is possible to copy a list by reassigning it to a new variable in the following way: list2 = list1. Now, list2 is a reference of list1, any changes we make in list2 will also modify the original, list1. But there are lots of case in which we do not like to modify the original instead we like to have a different copy. One of way of avoiding the problem above is using *copy()*.

```
# syntax
lst = ['item1', 'item2']
lst_copy = lst.copy()
```

```
fruits = ['banana', 'orange', 'mango', 'lemon']
fruits_copy = fruits.copy()
print(fruits_copy)       # ['banana', 'orange', 'mango', 'lemon']
```

Joining Lists

There are several ways to join, or concatenate, two or more lists in Python.

- Plus Operator (+)

```
# syntax
list3 = list1 + list2
```

```
positive_numbers = [1, 2, 3, 4, 5]
zero = [0]
negative_numbers = [-5,-4,-3,-2,-1]
integers = negative_numbers + zero + positive_numbers
print(integers) # [-5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5]
fruits = ['banana', 'orange', 'mango', 'lemon']
vegetables = ['Tomato', 'Potato', 'Cabbage', 'Onion', 'Carrot']
fruits_and_vegetables = fruits + vegetables
print(fruits_and_vegetables ) # ['banana', 'orange', 'mango', 'lemon', 'Tomato',
'Potato', 'Cabbage', 'Onion', 'Carrot']
```

- Joining using extend() method The *extend()* method allows to append list in a list. See the example below.

```
# syntax
list1 = ['item1', 'item2']
list2 = ['item3', 'item4', 'item5']
list1.extend(list2)
```

```
num1 = [0, 1, 2, 3]
num2= [4, 5, 6]
num1.extend(num2)
print('Numbers:', num1) # Numbers: [0, 1, 2, 3, 4, 5, 6]
negative_numbers = [-5,-4,-3,-2,-1]
positive_numbers = [1, 2, 3,4,5]
zero = [0]

negative_numbers.extend(zero)
negative_numbers.extend(positive_numbers)
print('Integers:', negative_numbers) # Integers: [-5, -4, -3, -2, -1, 0, 1, 2, 3,
4, 5]
fruits = ['banana', 'orange', 'mango', 'lemon']
vegetables = ['Tomato', 'Potato', 'Cabbage', 'Onion', 'Carrot']
fruits.extend(vegetables)
print('Fruits and vegetables:', fruits ) # Fruits and vegetables: ['banana',
'orange', 'mango', 'lemon', 'Tomato', 'Potato', 'Cabbage', 'Onion', 'Carrot']
```

Counting Items in a List

The *count()* method returns the number of times an item appears in a list:

```
# syntax
lst = ['item1', 'item2']
lst.count(item)
```

```
fruits = ['banana', 'orange', 'mango', 'lemon']
print(fruits.count('orange'))   # 1
ages = [22, 19, 24, 25, 26, 24, 25, 24]
print(ages.count(24))           # 3
```

Finding Index of an Item

The *index()* method returns the index of an item in the list:

```
# syntax
lst = ['item1', 'item2']
lst.index(item)
```

```
fruits = ['banana', 'orange', 'mango', 'lemon']
print(fruits.index('orange'))   # 1
ages = [22, 19, 24, 25, 26, 24, 25, 24]
print(ages.index(24))           # 2, the first occurrence
```

Reversing a List

The *reverse()* method reverses the order of a list.

```
# syntax
lst = ['item1', 'item2']
lst.reverse()
```

```
fruits = ['banana', 'orange', 'mango', 'lemon']
fruits.reverse()
print(fruits) # ['lemon', 'mango', 'orange', 'banana']
ages = [22, 19, 24, 25, 26, 24, 25, 24]
ages.reverse()
print(ages) # [24, 25, 24, 26, 25, 24, 19, 22]
```

Sorting List Items

To sort lists we can use *sort()* method or *sorted()* built-in functions. The *sort()* method reorders the list items in ascending order and modifies the original list. If an argument of *sort()* method reverse is equal to true, it will arrange the list in descending order.

- sort(): this method modifies the original list

- ```
  # syntax
  lst = ['item1', 'item2']
  lst.sort()                # ascending
  lst.sort(reverse=True)    # descending
  ```

  Example:

  ```
  fruits = ['banana', 'orange', 'mango', 'lemon']
  fruits.sort()
  print(fruits)             # sorted in alphabetical order, ['banana',
  'lemon', 'mango', 'orange']
  fruits.sort(reverse=True)
  print(fruits) # ['orange', 'mango', 'lemon', 'banana']
  ages = [22, 19, 24, 25, 26, 24, 25, 24]
  ages.sort()
  print(ages) #  [19, 22, 24, 24, 24, 25, 25, 26]

  ages.sort(reverse=True)
  print(ages) #  [26, 25, 25, 24, 24, 24, 22, 19]
  ```

  sorted(): returns the ordered list without modifying the original list **Example:**

  ```
  fruits = ['banana', 'orange', 'mango', 'lemon']
  print(sorted(fruits))   # ['banana', 'lemon', 'mango', 'orange']
  # Reverse order
  fruits = ['banana', 'orange', 'mango', 'lemon']
  fruits = sorted(fruits,reverse=True)
  print(fruits)     # ['orange', 'mango', 'lemon', 'banana']
  ```

😍 You are diligent and you have already achieved quite a lot. You have just completed day 5 challenges and you are 5 steps a head in to your way to greatness. Now do some exercises for your brain and muscles.
  ```

💻 Exercises: Day 5

Exercises: Level 1

1. Declare an empty list
2. Declare a list with more than 5 items
3. Find the length of your list
4. Get the first item, the middle item and the last item of the list
5. Declare a list called mixed_data_types, put your(name, age, height, marital status, address)
6. Declare a list variable named it_companies and assign initial values Facebook, Google, Microsoft, Apple, IBM, Oracle and Amazon.
7. Print the list using *print()*
8. Print the number of companies in the list
9. Print the first, middle and last company
10. Print the list after modifying one of the companies
11. Add an IT company to it_companies
12. Insert an IT company in the middle of the companies list
13. Change one of the it_companies names to uppercase (IBM excluded!)
14. Join the it_companies with a string '#; '
15. Check if a certain company exists in the it_companies list.
16. Sort the list using sort() method
17. Reverse the list in descending order using reverse() method
18. Slice out the first 3 companies from the list
19. Slice out the last 3 companies from the list
20. Slice out the middle IT company or companies from the list
21. Remove the first IT company from the list

22. Remove the middle IT company or companies from the list
23. Remove the last IT company from the list
24. Remove all IT companies from the list
25. Destroy the IT companies list
26. Join the following lists:
27.
```
front_end = ['HTML', 'CSS', 'JS', 'React', 'Redux']
back_end = ['Node','Express', 'MongoDB']
```
28. After joining the lists in question 26. Copy the joined list and assign it to a variable full_stack. Then insert Python and SQL after Redux.

Exercises: Level 2

1. The following is a list of 10 students ages:

```
ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]
```

- Sort the list and find the min and max age
- Add the min age and the max age again to the list
- Find the median age (one middle item or two middle items divided by two)
- Find the average age (sum of all items divided by their number)
- Find the range of the ages (max minus min)
- Compare the value of (min - average) and (max - average), use *abs()* method

1. Find the middle country(ies) in the countries list
2. Divide the countries list into two equal lists if it is even if not one more country for the first half.
3. ['China', 'Russia', 'USA', 'Finland', 'Sweden', 'Norway', 'Denmark']. Unpack the first three countries and the rest as scandic countries.

🎉 CONGRATULATIONS ! 🎉