Introduction to List

Course Title: Programming Language II
Course Code: CSE 111
Semester: Summer 2020
Lecture – 9

Last Lecture

- String
- Indexing
- Mutability of String
- Basic String operations
  - Concatenation
  - Deletion
  - Repetition
  - Slicing

Today’s Lecture

- Introduction to Lists
- List Manipulation
  - Creation
  - Indexing
  - Adding, accessing and removing elements
  - Mutability
  - Slicing
  - Concatenation

Concept of List

- List is a sequence of values called items or elements.
- The elements can be of any data type.
- Each value has a location (an index).
- Indexes range from 0 to n-1 where n is the length of the list.
- List is mutable, meaning, their elements can be changed.
- Values are enclosed in [], myList = [1, 2, 3].

Creating a List

- A list is created by placing all the elements inside a square bracket [ ], separated by commas.
- It can have any number of items and they may be of different types (integer, float, string etc.).

- Creating without constructor -

```
# empty list
my_list = []

# list of integers
my_list = [1, 2, 3]
```

```
# list with mixed data types
my_list = [1, "Abc", 4.5]

# nested list
my_list = ["Hello", [7, 8, 9]]
```

Creating a List(continued)

- Using list constructor –

```
# empty list
my_list = list()
# list of integers
my_list = list([1, 2, 3])
```

- List length – ***len()*** function

```
>>> a = [1,2,3]
>>> print(len(a))

Output will be 3
```

Indexing of List

- Just like strings, list can also be accessed using indexing method.
- Range: Index starts at 0 and ends at n-1.
- Type: Index must be an integer.
- Index out of range will give IndexError. Different type will give type error.
- Two indexing technique: Positive and Negative
- Example:

a = [11, 12, 13, 14, 15]

Pos. Index →	0	1	2	3	4
Items →	11	12	13	14	15
Neg. Index →	-5	-4	-3	-2	-1

Access items from a list

- Use the index operator [] to access an item in a list
- The index starts at 0 just like with a string and goes up to n-1 if there are n elements in the list
- They also can use negative subscripts, -1 is the last element on the right, -n on the left end

```
>>> a = [11, 12, 13, 14, 15]
>>> print(a[2])   #Output will be 13
>>> print(a[-2])   #Output will be 14
```

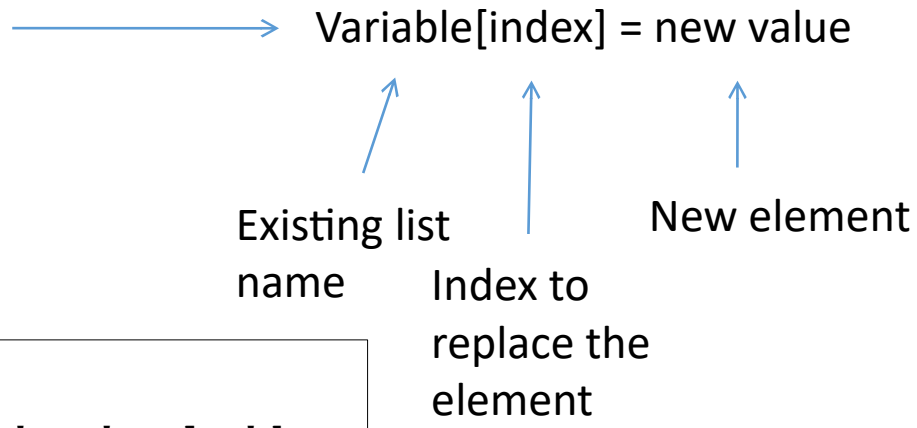
Access items from a list

```
# This is acceptable
>>> players_list = ['Ronaldo', 'Messi', 'Neymar', 'Bale']
>>> print(players_list[1])  #Positive Indexing
Output: Messi
>>> print(players_list[0])  #Positive Indexing
Output: Ronaldo
>>> print(players_list[-1])  #Negative Indexing
Output: Bale
```

```
# This is not acceptable
>>> print(players_list[5])
Output: IndexError: list index out of range
>>> print(players_list[4.0])
Output: TypeError: list integer must be integer or slices
```

List Mutability

- Lists are mutable (changeable)
- Unlike Strings items in list can easily be changed:
  - Can easily change an existing element
  - Can easily append/add new elements
  - Can easily delete/remove elements




```
# Example:
>>> players_list = ['Ronaldo', 'Messi', 'Neymar', 'Bale']
>>> print(players_list)
Output: ['Ronaldo', 'Messi', 'Neymar', 'Bale']
>>> print(players_list[2])
Output: Neymar
>>> player_list[2] = 'Mbappe' #Neymar is replaced by
Mbappe
>>> print(players_list)
Output: ['Ronaldo', 'Messi', 'Mbappe', 'Bale']
#See Neymar is no longer in the list
```

Add items to a list

- Using the ***append(value)*** method

```
>>> cars = [“Bmw”, “Audi”, “Porsche”]
>>> cars.append(“Ford”)
>>> print(cars)

Output: [“Bmw”, “Audi”, “Porsche”, “Ford”]
```

- Using the ***insert(index,value)*** method

```
>>> cars = [“Bmw”, “Audi”, “Porsche”]
>>> cars.insert(1, “Ford”)
>>> print(cars)

Output: [“Bmw”, “Ford”, “Audi”, “Porsche”]
```

Remove items from a list

- Using the ***pop(index) or pop()*** method

```
>>> cars = [“Bmw”, “Audi”, “Porsche”]
>>> cars.pop(1)
>>> print(cars)

Output: [“Bmw”, “Porsche”]
```

```
>>> cars = [“Bmw”, “Audi”, “Porsche”]
>>> cars.pop()
>>> print(cars)

Output: [“Bmw”, “Audi”]
```

- Using the ***remove(value)*** method

```
>>> cars = [“Bmw”, “Audi”, “Porsche”]
>>> cars.remove(“Bmw”)
>>> print(cars)

Output: [“Audi”, “Porsche”]
```

Remove items from a list

- Using the ***del*** keyword

```
>>> cars = [“Bmw”, “Audi”, “Porsche”]
>>> del cars[0]
>>> print(cars)

Output: [“Audi”, “Porsche”]
```

```
>>> cars = [“Bmw”, “Audi”, “Porsche”]
>>> del cars
>>> print(cars)

Output: name 'cars' is not defined
```

- Using the ***clear()*** method

```
>>> cars = [“Bmw”, “Audi”, “Porsche”]
>>> cars.clear()
>>> print(cars)

Output:[]
```

Copy a list

- Using the ***copy()*** method

```
>>> cars = [“Bmw”, “Audi”, “Porsche”]
>>> cars2 = cars.copy()
>>> print(cars2)

Output: [“Bmw”, “Audi”, “Porsche”]
```

Joining two list

- Using the ***(+)*** operator

```
>>> list1 = [1, 2, 3]
>>> list2 = [4, 5, 6]
>>> list3 = list1 + list2
>>> print(list3)

Output: [1, 2, 3, 4, 5, 6]
```

- Using the ***extend(list)*** method

```
>>> list1 = [1, 2, 3]
>>> list2 = [4, 5, 6]
>>> list1.extend(list2)
>>> print(list1)

Output: [1, 2, 3, 4, 5, 6]
```

List Slicing

- list[start:end:step]
  - start(inclusive): specifies the starting index. If not provided, starts at 0.
  - end(exclusive): specifies the ending index. If not provided, ends at last index.
  - step(optional): specifies the increment
- the return value will be a new list with the specified items

```
>>> colors = [“Black”, “White”, “Red”, “Blue”, “Green”, “Purple”, “Yellow”]
>>> clr = colors[1:4]
>>> print(clr)              #Output will be: ['White', 'Red', 'Blue']
>>> print(colors[:4])       #Output will be: ['Black', 'White', 'Red', 'Blue']
>>> print(colors[3:])       #Output will be: ['Blue', 'Green', 'Purple',
'Yellow']
>>> print(colors[-4:-1])    #Output will be: ['Blue', 'Green', 'Purple']
```

List methods

Method	Semantics
list.sort()	Sorts items of list, in place. Items must be comparable
list.reverse()	Reverses order of items in list in place
list.index(x)	Returns integer index of first (leftmost) occurrence of x in the list
list.count(x)	Returns integer count of number of occurrences of x in list

Next Lecture

- Dictionaries


Thank
You!