List

Topics to cover

- Defining List
- Indexing, slicing, check membership
- length (len), compare (cmp), max, min, convert list to tuple and string
- copy, shallow copy
- Multi-dimension
- list to string using join
- string to list using split
- Methods
  - append
  - insert
  - count
  - extend
  - index
  - pop
  - remove
  - reverse
  - sort (reverse=True)
  - sorted

tuples vs lists

1. tuples are immutable i.e. cannot be changed unlike lists
2. tuples use parentheses, whereas lists use square brackets

Declaring a List

In [1]:
```
cs101_students = []        # create empty list
cs101_students = list()    # create empty list
cs101_students = ['Sanam','Sachin','Ajit']
print (cs101_students)
print (type(cs101_students))
```

```
['Sanam', 'Sachin', 'Ajit']
<class 'list'>
```

Accessing Elements of list

- Using slicing and indexing - as we did in string manipulation

In [2]:
```
cs101_students = ['Sanam','Sachin','Ajit']
print (cs101_students[0])
print (cs101_students[0:2])
print (cs101_students[-2:])  # negative indexing
```

```
Sanam
['Sanam', 'Sachin']
['Sachin', 'Ajit']
```

Updating List - using index

- You can replace old element with new
- New element can not be added to list

In [3]:
```
cs101_students = ['Sanam','Sachin','Ajit']
print (cs101_students)
cs101_students[1] = 'Piyush'
print (cs101_students)
# cs101_students[3] = 'Piyush'           # Not valid operation - Give Index Error
```

```
['Sanam', 'Sachin', 'Ajit']
['Sanam', 'Piyush', 'Ajit']
```

Updating List - using append() method

- append method takes one input as parameter
- add new element to the end of list

In [4]:
```
cs101_students = ['Sanam','Sachin','Ajit']
print (cs101_students)
cs101_students.append('Piyush')
print (cs101_students)
cs101_students.append('Deepa')
print (cs101_students)
```

```
['Sanam', 'Sachin', 'Ajit']
['Sanam', 'Sachin', 'Ajit', 'Piyush']
['Sanam', 'Sachin', 'Ajit', 'Piyush', 'Deepa']
```

Insertion of element

- `listname.insert(index,element)`
  - `listname` is name of list
  - `index` is position where insertion needs to be done

In [5]:
```
cs101_students = ['Sanam','Sachin','Ajit']
cs101_students.insert(1,"Piyush")          # Insert "Piyush" at index 1
print (cs101_students)
cs101_students.insert(1,[1,2,3])          # Insert list [1,2,3] at index 1
print (cs101_students)
```

```
['Sanam', 'Piyush', 'Sachin', 'Ajit']
['Sanam', [1, 2, 3], 'Piyush', 'Sachin', 'Ajit']
```

Extend list

- Adding multiple element to the end of list

In [6]:
```
cs101_students = ['Sanam','Sachin','Ajit']
cs102_students = ['Deepa','Rutuja']

print (cs101_students)
cs102_students.extend(cs101_students)
print (cs102_students)
```

```
['Sanam', 'Sachin', 'Ajit']
['Deepa', 'Rutuja', 'Sanam', 'Sachin', 'Ajit']
```

Basic List operation

- len
- min
- max
- Repetition
- Membership
- Iteration

In [7]:
```
cs101_students = ['Sanam','Sachin','Ajit']

cs101_marks = 4*[96,]

print ("length of list: ", len(cs101_students))
print ("min. value: ", min(cs101_students))
print ("max. value: ", max(cs101_students))
print ("repeat same element in list: ", cs101_marks)
```

```
length of list:  3
min. value:  Ajit
max. value:  Sanam
repeat same element in list:  [96, 96, 96, 96]
```

In [8]:
```
# check membership of element
cs101_students = ['Sanam','Sachin','Ajit']

print ('Sanam' in cs101_students)   # return true as Sanam is present
print ('sanam' in cs101_students)   # return false as sanam(case-sensitive) is not present
```

```
True
False
```

In [9]:
```
cs101_students = ['Sanam','Sachin','Ajit']

## Iterate through all the elements of list
print ("Printing all elements of list")
for i in cs101_students:
    print (i)
```

```
Printing all elements of list
Sanam
Sachin
Ajit
```

Counting element in list

- `listname.count(element)`
  - `listname` is name of list
  - return number of time "element" is present in "listname"

In [10]:
```
cs101_students = ['Sanam','Sachin','Ajit','Ajit']
print ("Ajit is present %d time(s)" %cs101_students.count("Ajit"))
print ("Deepa is present %d time(s)" %cs101_students.count("Deepa"))
```

```
Ajit is present 2 time(s)
Deepa is present 0 time(s)
```

Finding index of element in list

- `listname.index(element)`
  - `listname` is name of list
  - return index of first occurrence of `element` in `listname`
  - If element is not present in list, error is raised

In [11]:
```
cs101_students = ['Sanam','Sachin','Ajit','Ajit']
print ("Sachin is present at %d position" %cs101_students.index("Sachin"))
#print ("Deepa is present %d time" %cs101_students.index("Deepa"))   # Gives value error as element is not present in lis
```

```
Sachin is present at 1 position
```

pop

- pop (extract and remove element)
- `listname.pop(index)`
  - `listname` is name of list
  - `index` is position of element which has to be removed. If it is not specified, last element is removed
  - return element which is removed
  - list is updated, with last element removed

In [12]:
```
cs101_students = ['Sanam','Sachin','Ajit','Deepa']
print ("Element pop is: ",cs101_students.pop())   # pop element at last index
print ("Updated list is: ",cs101_students)

cs101_students = ['Sanam','Sachin','Ajit','Deepa']
print ("Element pop is: ",cs101_students.pop(1))  # pop element at index 1
print ("Updated list is: ",cs101_students)
```

```
Element pop is:  Deepa
Updated list is:  ['Sanam', 'Sachin', 'Ajit']
Element pop is:  Sachin
Updated list is:  ['Sanam', 'Ajit', 'Deepa']
```

remove

- remove (remove element)
- `listname.remove(element)`
  - `listname` is name of list
  - `element` is one which has to be removed. If element is not present in list, gives value error
- How remove is different from pop - does not return anything

In [13]:
```
cs101_students = ['Sanam','Sachin','Ajit','Deepa','Ajit']
print ("Element removed is: ",cs101_students.remove('Ajit'))      # remove method returns nothing
print ("Updated list is: ",cs101_students)                         # It will remove the first matching element
#cs101_students.remove('Saurav')       # Element 'Saurav' is not present - hence this will give value error
```

```
Element removed is:  None
Updated list is:  ['Sanam', 'Sachin', 'Deepa', 'Ajit']
```

reverse

- reverse ordering of list in place
- return nothing

In [14]:
```
cs101_students = ['Sanam','Sachin','Ajit','Deepa']
#reversed_list = cs101_students
rev = cs101_students.reverse()              # method reverse returns nothing
print (cs101_students)                        # List is updated in reverse order
print (rev)
```

```
['Deepa', 'Ajit', 'Sachin', 'Sanam']
None
```

sort

- sort list in place
- return nothing
- sorting order can be reversed using reverse parameter by setting `reverse=True`

In [15]:
```
cs101_students = ['Sanam','Sachin','Ajit','Deepa']
#reversed_list = cs101_students
rev = cs101_students.sort()                 # method sort returns nothing
print (cs101_students)                      # List is updated in sorted order
print (rev)

cs101_students.sort(reverse=True)           # reverse sorting
print (cs101_students)
```

```
['Ajit', 'Deepa', 'Sachin', 'Sanam']
None
['Sanam', 'Sachin', 'Deepa', 'Ajit']
```

Copying a list

- copying one list to another list

In [16]:
```
my_list = [1,2,3,4]
new_list = my_list

print (my_list)
print (new_list)

my_list[0] = 9

print (my_list)
print (new_list)       #Both the list refer to the same list
```

```
[1, 2, 3, 4]
[1, 2, 3, 4]
[9, 2, 3, 4]
[9, 2, 3, 4]
```

Conversion to other data-structure

- it is possible to convert List to Tuple or String and vice-versa

In [17]:
```
cs101_students = ['Sanam','Sachin','Ajit','Deepa']

## List to tuple
tup = tuple(cs101_students)
print ("List to Tuple: ",tup, type(tup))

## Tuple to List
lst = list(tup)
print ("Tuple to list: ",lst, type(lst))

##List to String
string = " ".join(cs101_students)
print ("List to String: ", string, type(string))

# ##String to List
str_lst = list(string.split( ))
print ("String to List: ",str_lst, type(str_lst))
```

```
List to Tuple:  ('Sanam', 'Sachin', 'Ajit', 'Deepa') <class 'tuple'>
Tuple to list:  ['Sanam', 'Sachin', 'Ajit', 'Deepa'] <class 'list'>
List to String:  Sanam Sachin Ajit Deepa <class 'str'>
String to List:  ['Sanam', 'Sachin', 'Ajit', 'Deepa'] <class 'list'>
```

Multi-dimension list

- List of list
- List of tuple
- List of tuple, list and string

In [18]:
```
roll = (1,2,3,4,5)
name = ('Sanam', 'Sachin', 'Ajit', 'Deepa','Piyush')
marks = [10,5.5,8,9,8.5]
subject = ['Electronics','C-Prog', 'Data-structure','Digital Circuits']

information = [0]

information.insert(5,name)
information.insert(2,name)
information.insert(0,marks)
information.insert(4,subject)

print (information)
print (information[0])
print (information[2][2])
print (information[4][3])
```

```
[[10, 5.5, 8, 9, 8.5], 0, ('Sanam', 'Sachin', 'Ajit', 'Deepa', 'Piyush'), ('Sanam', 'Sachin', 'Ajit', 'Deepa', 'Piyus
h'), ['Electronics', 'C-Prog', 'Data-structure', 'Digital Circuits']]
[10, 5.5, 8, 9, 8.5]
Ajit
Digital Circuits
```