|  |  |
| --- | --- |
| **Ex. No. 9**  **Date:19.06.2021** | **LIST AND TUPLES – LEVEL 1** |

**AIM:**

To write a simple python program using List and Tuple.

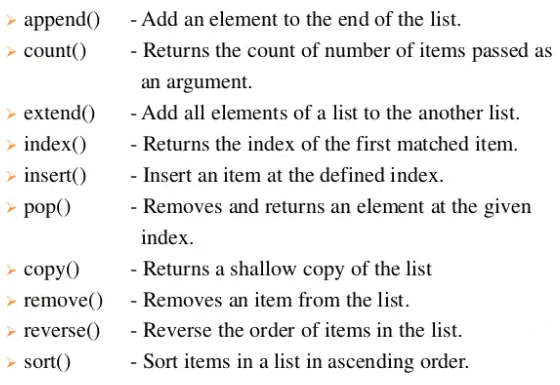
**PROGRAMMING BASE:**

**LIST**

Lists are Python’s most flexible ordered collection object type and it is mutable.

* Lists are sequences of values
* Values need not be of uniform type
* Lists may be nested
* Can access value at a position, or a slice
* Lists are mutable, can update in place
* Assignment does not copy the value
* Use full slice to make a copy of a list

**List Methods**

****

**TUPLE**

* Tuples are the sequences of different types of values.
* Elements are separated by commas inside the parentheses and are assigned to a variable to create a tuple.
* Tuples can be created with or without parenthesis.
* Nested tuples can be created.
* To complete the assignment of the tuple, a final comma must be added after the element.
* In order to access the values in a tuple, it is necessary to use the index number enclosed in square brackets along with the name of the tuple.

**Basic Tuple Operations**

|  |  |  |
| --- | --- | --- |
| **Python Expression** | **Results** | **Operation** |
| len((1, 2, 3)) | 3 | Length |
| (1, 2, 3) + (4, 5, 6) | (1, 2, 3, 4, 5, 6) | Concatenation |
| ('Hi!',) \* 4 | ('Hi!', 'Hi!', 'Hi!', 'Hi!') | Repetition |
| 3 in (1, 2, 3) | True | Membership |
| for x in (1, 2, 3): print x, | 1 2 3 | Iteration |
| L[2] | 'SPAM!' | Indexing-Offsets start at zero |
| L[-2] | 'Spam' | Slicing-Negative: count from the right |
| L[1:] | ['Spam', 'SPAM!'] | Matrixes-Slicing fetches sections |

**Built-in Tuple Functions**

len(tuple) ->Gives the total length of the tuple.

max(tuple) ->Returns item from the tuple with max value.

min(tuple) ->Returns item from the tuple with min value.

tuple(seq) ->Converts a list into tuple.

cmp(tuple1, tuple2)-> Compares elements of both tuples.

**PROGRAMS:**

**a) List Operations**

**Description:**

To perform various operations in List data structure.

Comment statements should be included in the necessary places.

Define function for each operations.

Generate the following choice:

1. Insert a new value

2. Insert a new value at a specific location

3. Remove the element from the list

4. Remove the element at the specific location

5. Display the list

6. Sort the list

7. Reverse the list

8. End

Repeat the choice until the choice is 8.

**Program Template:**

# 1. Insert a new value

definsert\_new(a,val):

# 2. Insert a new value at a specific location

definsert\_pos(a,pos,val):

# 3. Remove the element from the list

defremove\_element(a,val):

# 4. Remove the element at the specific location

defremove\_pos\_ele(a,pos):

# 5. Display the list

def display(a):

# 6. Sort the list

def sorting(a):

# 7. Reverse the list

def reversing(a);

# Main Program

a = []

while(True):

# Generate the choice here

**Program:**

'''Name : R.sridevi

Roll number: 20UIT021

Program Name: List Operations'''

# 1. Insert a new value

def insert\_new(a,val):

a.append(val)

# 2. Insert a new value at a specific location

def insert\_pos(a,pos,val):

a[pos]=val

# 3. Remove the element from the list

def remove\_element(a,val):

a.remove(val)

# 4. Remove the element at the specific location

def remove\_pos\_ele(a,pos):

del a[pos]

# 5. Display the list

def display(a):

print(a)

# 6. Sort the list

def sorting(a):

print(sorted(a))

# 7. Reverse the list

def reversing(a):

a.reverse()

print(a)

# Main Program

a = []

n=int(input('Enter the number of elements in list : '))

print('Enter the list elements...')

for i in range(n):

element = a.append(int(input()))

while(True):

# Generate the choice here

print('''

C H O I C E S

=============

1. Insert a new value

2. Insert a new value at a specific location

3. Remove the element from the list

4. Remove the element at the specific location

5. Display the list

6. Sort the list

7. Reverse the list

8. End''')

choice = input('\nEnter your choice : ')

if choice == '1':

val=int(input('Enter value to be inserted : '))

insert\_new(a,val)

elif choice == '2':

pos = int(input('Enter the position to insert the value : '))

val = int(input('Enter the value to be inserted : '))

insert\_pos(a,pos,val)

elif choice == '3':

val = int(input('Enter the value to be removed : '))

remove\_element(a,val)

elif choice == '4':

pos = int(input('Enter the position of the value to be removed : '))

remove\_pos\_ele(a,pos)

elif choice == '5':

display(a)

elif choice =='6':

sorting(a)

elif choice == '7':

reversing(a)

elif choice == '8':

break

else:

print('\tPlease give valid choice..!')

**Output:**

Enter the number of elements in list:3

Enter the list elements:

898

10

24

C H O I C E S

=============

1. Insert a new value

2. Insert a new value at a specific location

3. Remove the element from the list

4. Remove the element at the specific location

5. Display the list

6. Sort the list

7. Reverse the list

8. End

Enter your choice: 8

**b) Sort the list of tuples in ascending order based on last element**

**Description:**

Write a Python program to get a list, sorted in increasing order by the last element in each tuple from a given list of non-empty tuples.

Get the maximum limit from the user.

Get 2 tuples for each list element.

Example:

In the below example there are list with 4 tuples each has 2 elements.

[(2,1), (3,4), (1,2), (5,3)]

Sample Input

4

2

1

3

4

1

2

5

3

Sample Output

Before Sorting

[(2,1), (3,4), (1,2), (5,3)]

After Sorting

[(2,1), (1,2), (5,3), (3,4)]

Sample Input

0

Sample Output

Sorry no elements

Sample Input

-9

Sample Output

Not possible

**Program:**

'''Name : R.sridevi

Roll Number : 20UIT021

Program Name : Sort the list of tuples in as'''

max=int(input())

if(max!=0):

if(max>0):

list=[]

for i in range(max):

list1=[]

for j in range(2):

ip=int(input())

list1.append(ip)

list1=tuple(list1)

list.append(list1)

print('Before Sorting')

print(list)

print('After Sorting')

#print the list After Sorting

for i in range((max)):

for j in range(i,max):

if(list[i][1]>list[j][1]):

list[i],list[j]=list[j],list[i]

print(list)

else:

print('Not possible')

else:

print('Sorry no elements')

**Test Cases:**

|  |  |  |
| --- | --- | --- |
| **Test Case No.** | **Input** | **Expected Output** |
| 1 | 5  2  1  3  4  1  2  5  3 | Before Sorting  [(2, 1), (3, 4), (1, 2), (5, 3)]  After Sorting  [(2, 1), (1, 2), (5, 3), (3, 4)] |
| 2 | 0 | Sorry no elements |
| 3 | -9 | Not possible |
| **Total Test Cases** | | **3** |
| **Number of Test Cases Passed** | | **3** |

**RESULT:**

Thus, the Python programs are executed successfully.