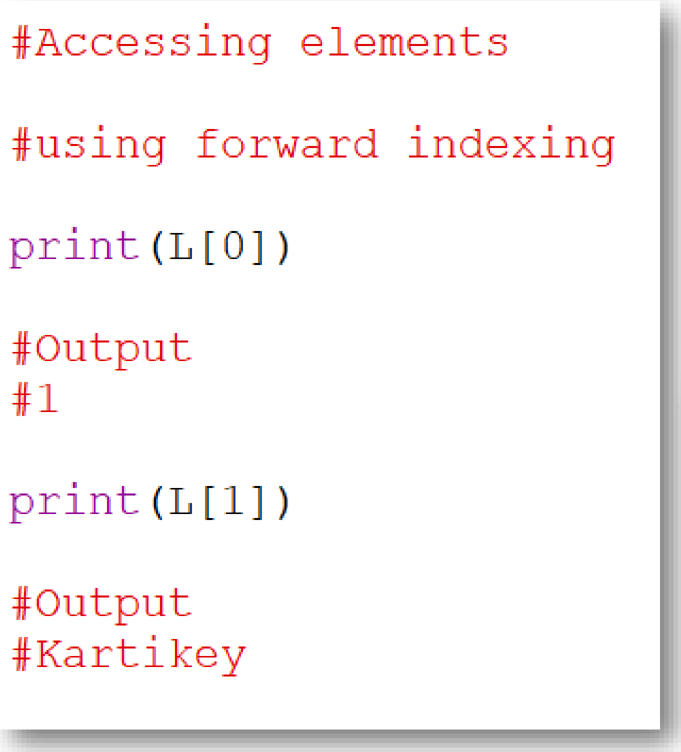
Chapter 8:

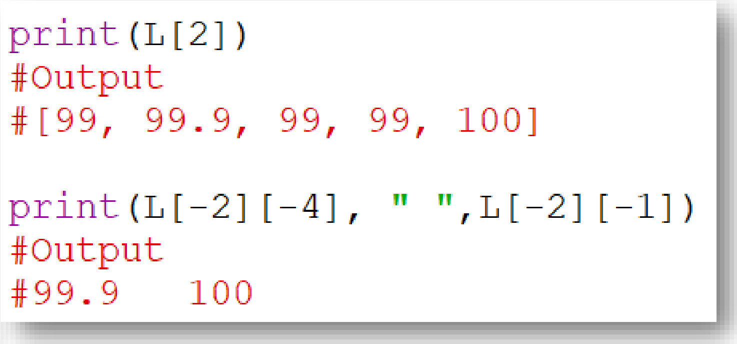
Data Structures -1 Linear Lists (Basics) (B)

By: Jagdish Devrani
Lecturer in Computer Science
Ahlcon Public Shool

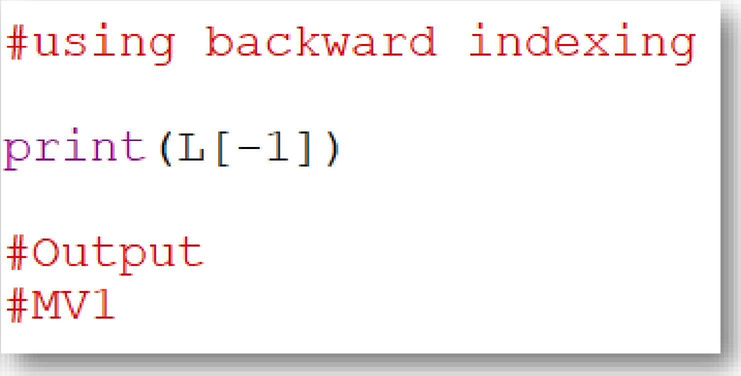
Index : 1



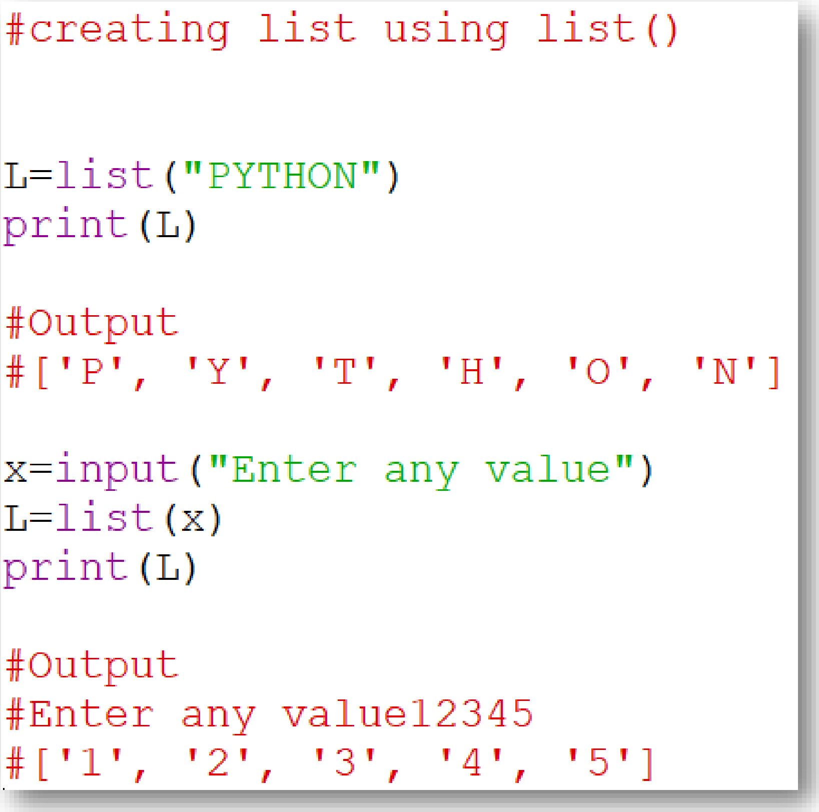
```
#Accessing elements

#using forward indexing

print(L[0])

#Output
#1

print(L[1])

#Output
#Kartikey
```



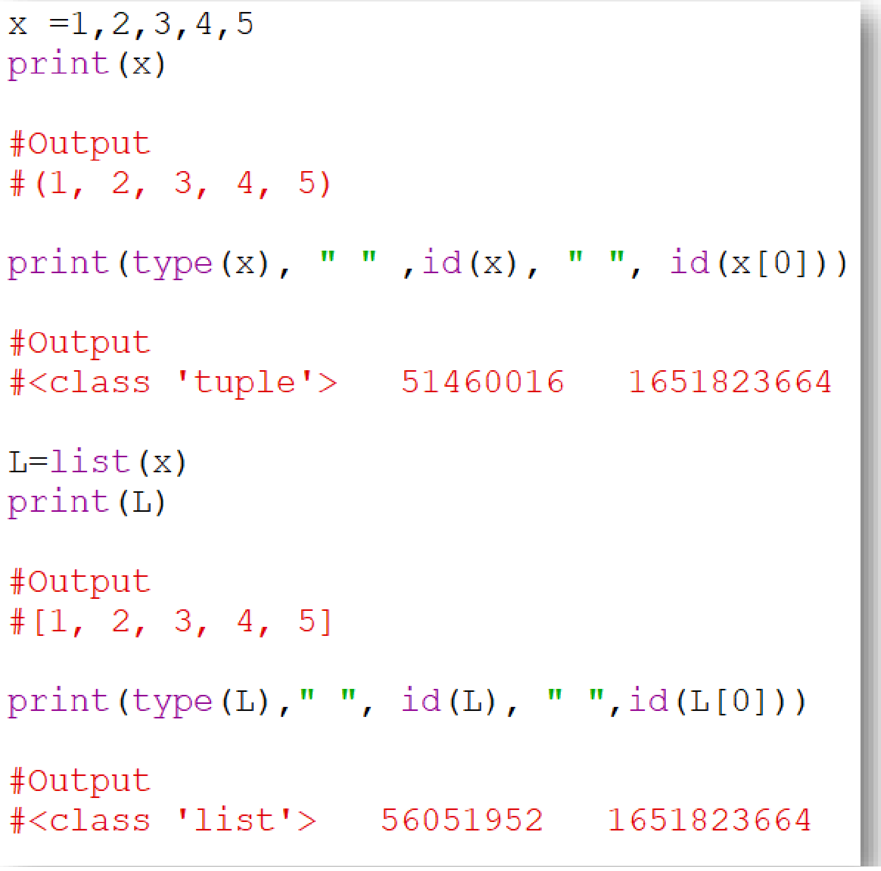
```
print(L[2])
#Output
#[99, 99.9, 99, 99, 100]

print(L[-2][-4], " ",L[-2][-1])
#Output
#99.9   100
```



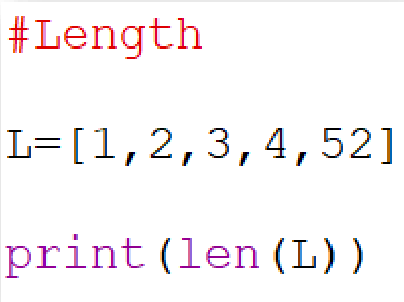
```
#using backward indexing

print(L[-1])

#Output
#MV1
```



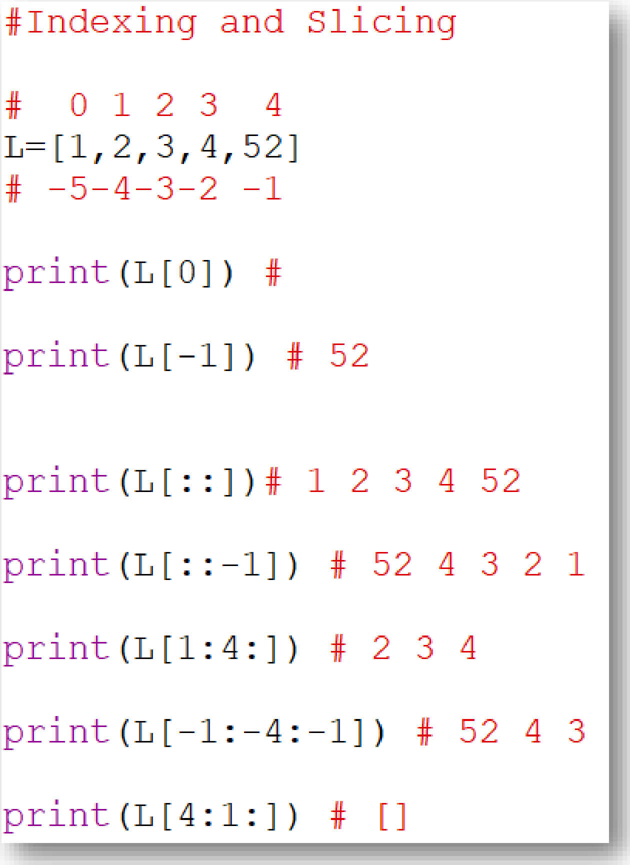
```
#creating list using list()


L=list("PYTHON")
print(L)

#Output
#['P', 'Y', 'T', 'H', 'O', 'N']

x=input("Enter any value")
L=list(x)
print(L)

#Output
#Enter any value12345
#['1', '2', '3', '4', '5']
```



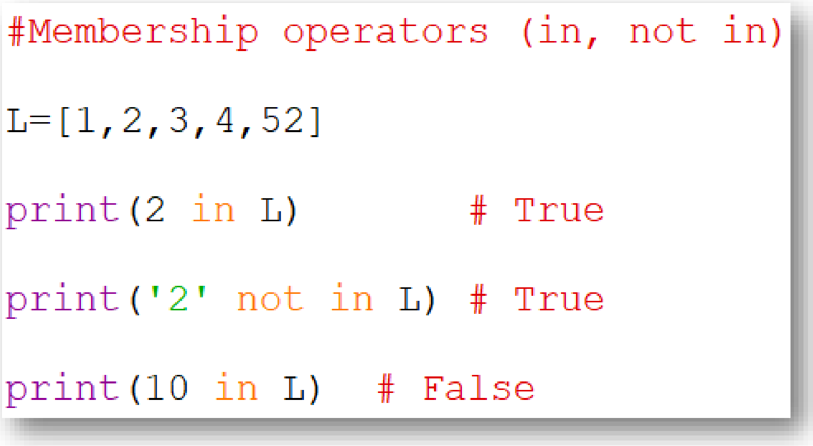
```
x =1,2,3,4,5
print(x)

#Output
#(1, 2, 3, 4, 5)

print(type(x), " " ,id(x), " ", id(x[0]))

#Output
#<class 'tuple'>   51460016   1651823664

L=list(x)
print(L)

#Output
#[1, 2, 3, 4, 5]

print(type(L)," ", id(L), " ",id(L[0]))

#Output
#<class 'list'>   56051952   1651823664
```



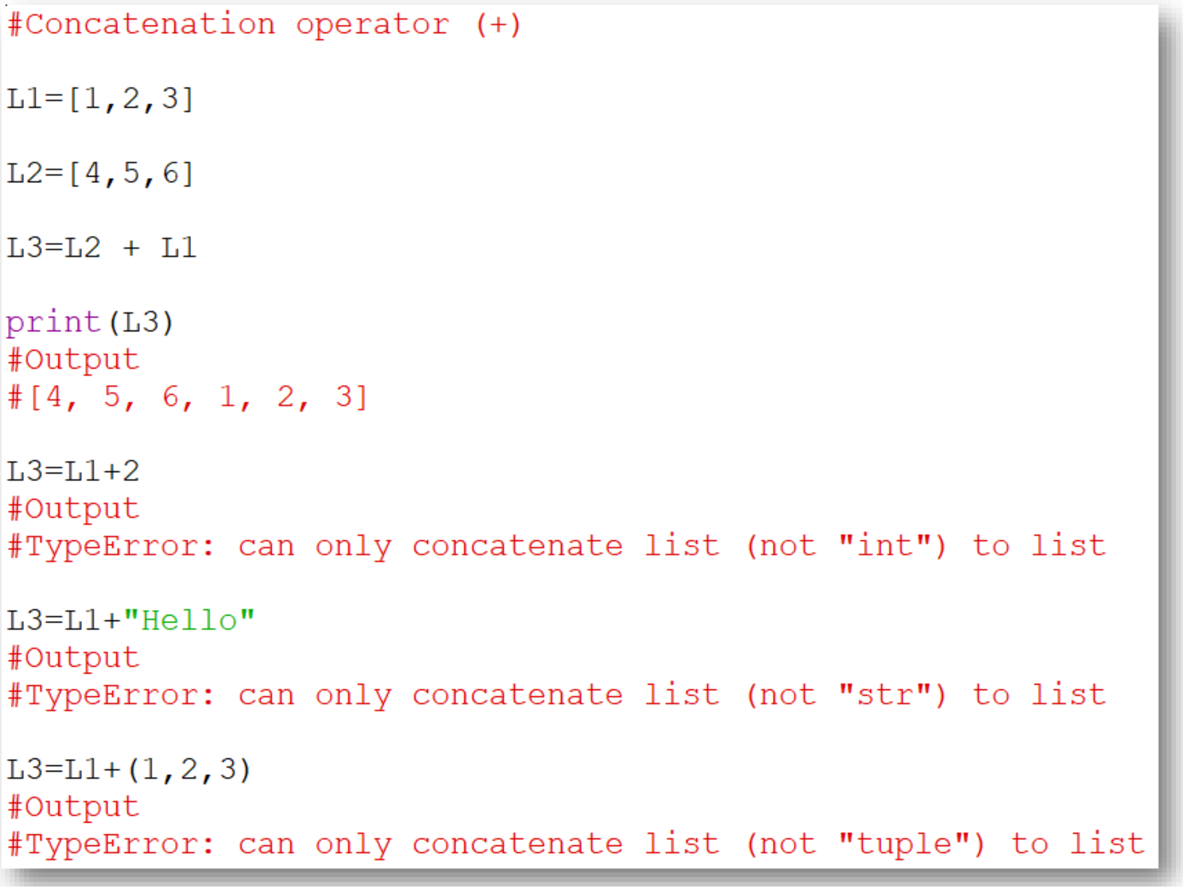
```
#Length

L=[1,2,3,4,52]

print(len(L))
```



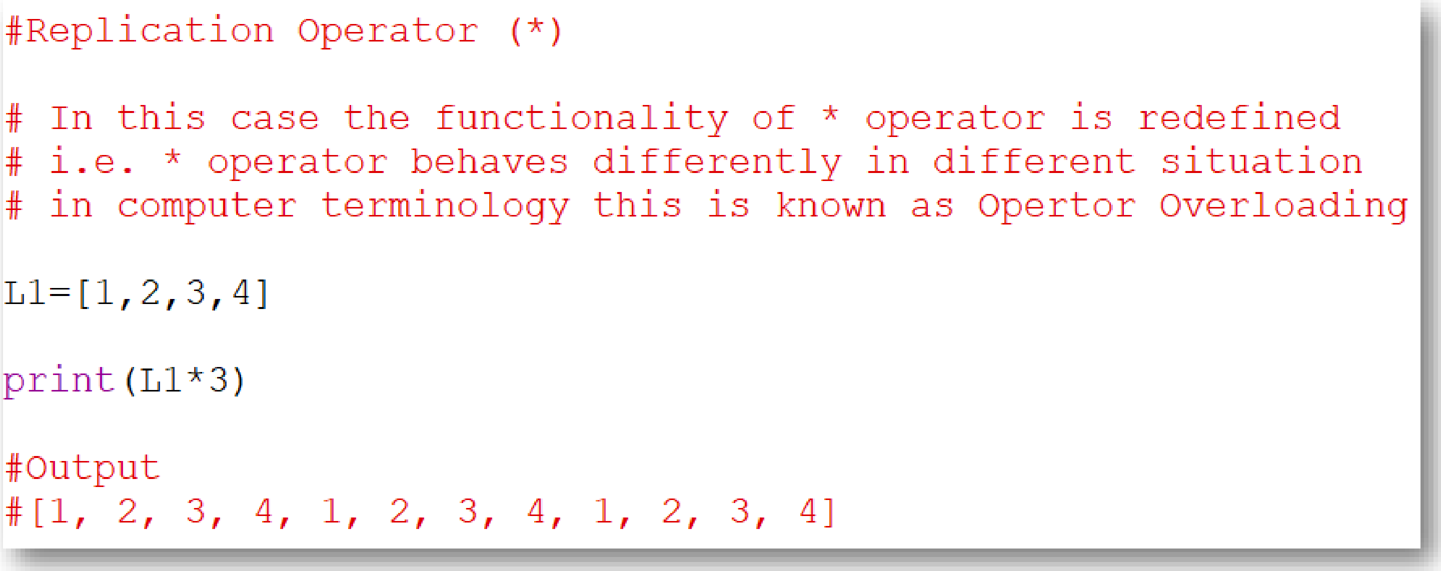
```
#Indexing and Slicing

#   0 1 2 3  4
L=[1,2,3,4,52]
#  -5-4-3-2 -1

print(L[0]) #

print(L[-1]) # 52


print(L[::])# 1 2 3 4 52

print(L[::-1]) # 52 4 3 2 1

print(L[1:4:]) # 2 3 4

print(L[-1:-4:-1]) # 52 4 3

print(L[4:1:]) # []
```



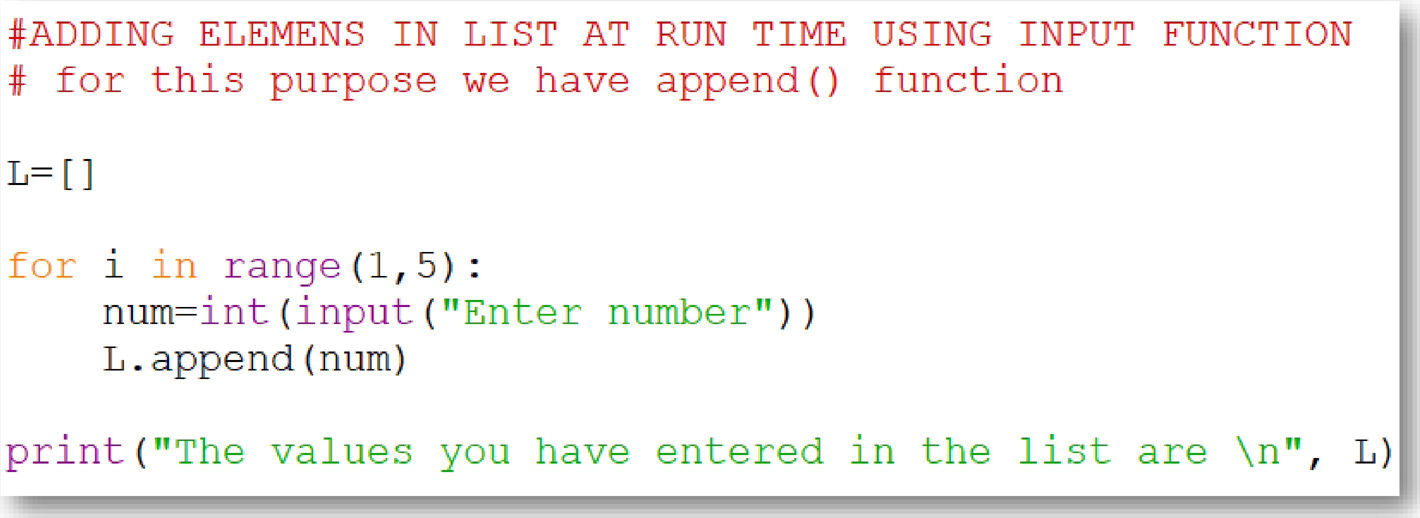
```
#Membership operators (in, not in)

L=[1,2,3,4,52]

print(2 in L)         # True

print('2' not in L) # True

print(10 in L)   # False
```



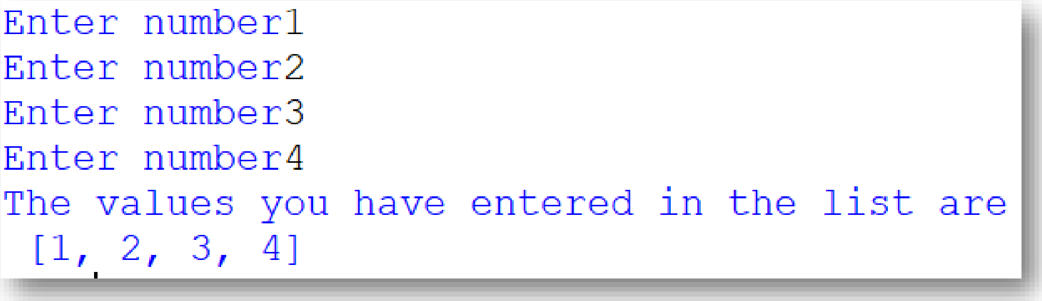
```
#Concatenation operator (+)

L1=[1,2,3]

L2=[4,5,6]

L3=L2 + L1

print(L3)
#Output
#[4, 5, 6, 1, 2, 3]

L3=L1+2
#Output
#TypeError: can only concatenate list (not "int") to list

L3=L1+"Hello"
#Output
#TypeError: can only concatenate list (not "str") to list

L3=L1+(1,2,3)
#Output
#TypeError: can only concatenate list (not "tuple") to list
```



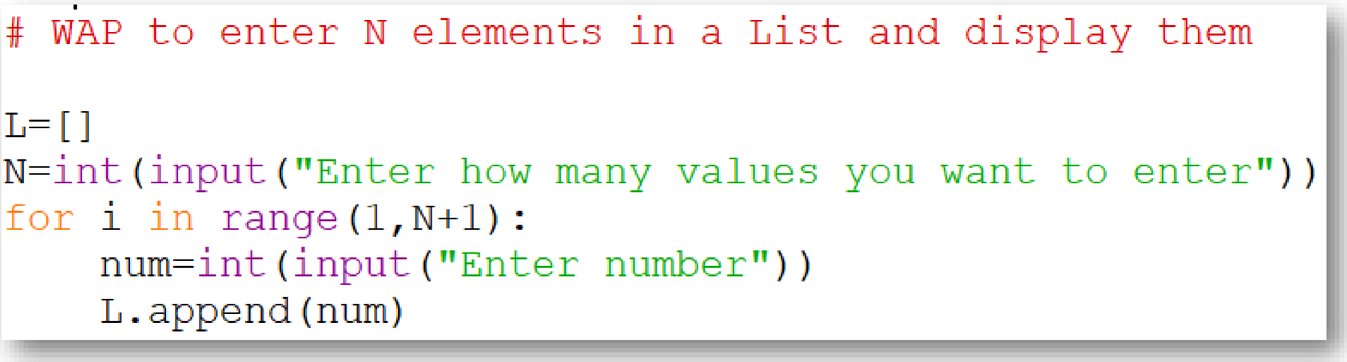
```
#Replication Operator (*)

# In this case the functionality of * operator is redefined
# i.e. * operator behaves differently in different situation
# in computer terminology this is known as Opertor Overloading

L1=[1,2,3,4]

print(L1*3)

#Output
#[1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4]
```



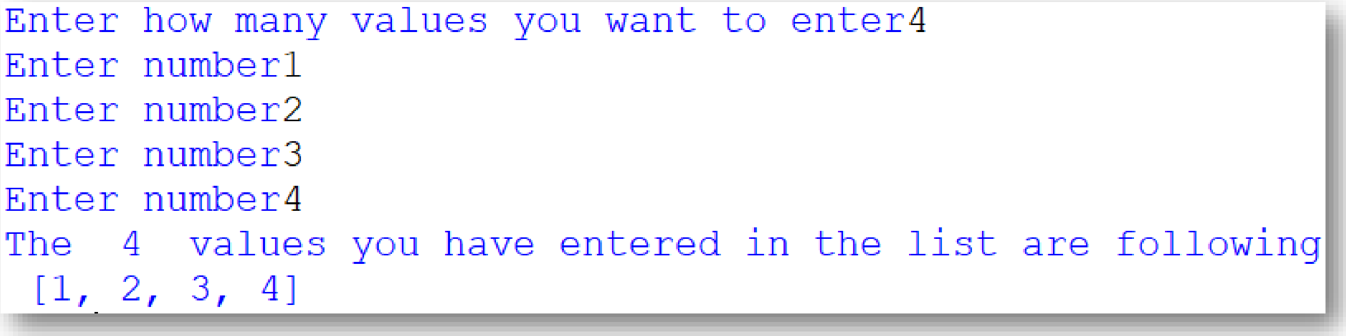
```
#ADDING ELEMENS IN LIST AT RUN TIME USING INPUT FUNCTION
# for this purpose we have append() function

L=[]

for i in range(1,5):
    num=int(input("Enter number"))
    L.append(num)

print("The values you have entered in the list are \n", L)
```



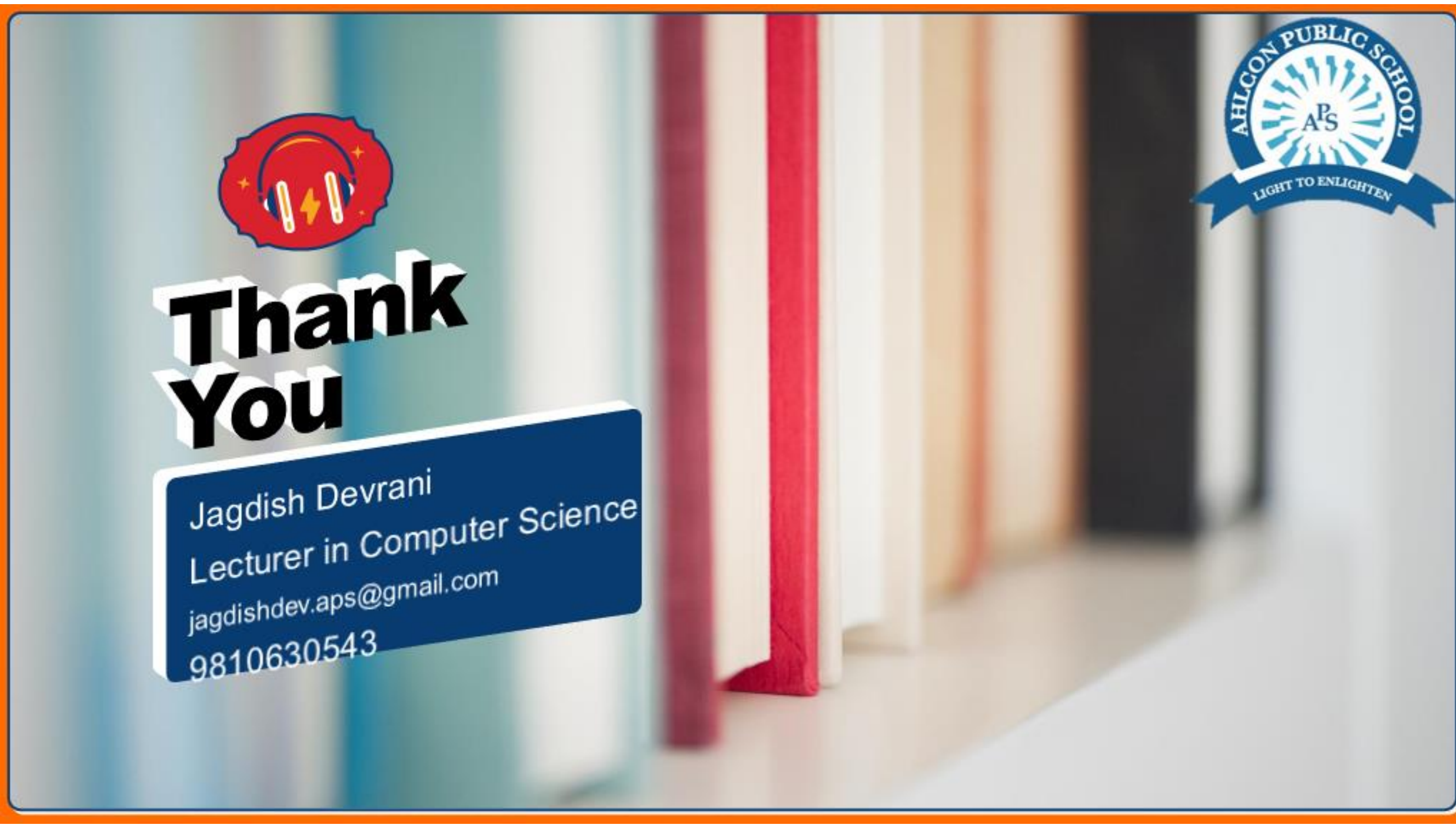
```
Enter number1
Enter number2
Enter number3
Enter number4
The values you have entered in the list are
 [1, 2, 3, 4]
```



```
# WAP to enter N elements in a List and display them

L=[]
N=int(input("Enter how many values you want to enter"))
for i in range(1,N+1):
    num=int(input("Enter number"))
    L.append(num)
```



```
Enter how many values you want to enter4
Enter number1
Enter number2
Enter number3
Enter number4
The  4  values you have entered in the list are following
 [1, 2, 3, 4]
```



Thank You

Jagdish Devrani
Lecturer in Computer Science
jagdishdev.aps@gmail.com
9810630543