# Lists and Strings in Python

### An array expressed as a LIST in python:

L = [23, 42, 60, 10, 54, 73]

```
L = [ 23, 42, 60, 10, 54, 73 ]
for j in range ( 0 , 5):
print j, L[ j ]
```

0 23

1 42

2 60

3 10

4 54

len(array) function gives the length of the array

The statement containing simply the name of the array prints all the elements of the array

```
Marks = [ 45, 34, 67, 70]
for j in range ( len(Marks)):
Marks[ j ] = Marks[ j ] + 5
Marks
```

[49, 39, 72, 75]

#### A different way to add elements of an array.

```
def addMarks( A ):
Total = 0
for marks in A:
   Total = Total + marks
return Total
def addMarks( A ):
Total = 0
for x in A:
   Total = Total + x
return Total
```

# SEARCHING AN ARRAY

Searching for a specific element Searching for largest element Searching for smallest element

# LINEAR SEARCH function to search if a specific element is present in the array.

If it is present, return the index of the element. If it is not present, return -1.

```
def Linsearch( A, x):
    for j in range (len(A)):
        if (A[j] == x):
            return j
    return -1
L = [ 23, 42, 60, 10, 70, 50 ]
Found = Linsearch ( L, 10)
print 'location = ', Found

location = 3
```

#### Searching for the largest element

```
def Largest( A ):
large = A[ 0 ]
for j in range (len(A)):
    if (A[ j ] > large):
        large = A[ j ]
return large
```

```
L = [ 23, 42, 60, 10, 70, 50 ]
Found_large = Largest ( L)
print 'largest element = ', Found_large
```

Largest element = 70

# List operations & methods

#### + Operator

```
L1 = [ 1, 4, 5 ]

L2 = [ 3, 7, 9 ]

L3 = L1 + L2

print L3

[1, 4, 5, 3, 7, 9 ]
```

```
* Operator
L1 = [ 1, 4, 5 ]
L3 = L1 * 3
print L3
[1, 4, 5, 1, 4, 5, 1, 4, 5 ]
```

#### **Slice Operator**

```
L1[1:4]
[ 4, 5, 8 ]
L1[ :4]
[1, 4, 5, 8 ]
L1 [ 4:]
[ 4, 6, 7 ]
```

## **Updating group of elements**

#### **Append and Extend Methods**

L1 = [ 1, 4, 5 ] L1.append(9) print L1

[1, 4, 5, 9]

L2 = [ 7, 8 ] L1.extend( L2 ) print L1 [1, 4, 5, 9, 7, 8 ]

#### **Sort Method**

```
L3 = [ 1, 14, 8, 6, 9]
L3.sort()
print L3
```

[1, 6, 8, 9, 14]

#### **Creating a List with all zero elements**

```
def makeL(size):
    mylist = [ ]
    for k in range(size):
        mylist.append(0)
    return mylist
```

# Short cut for Creating a List with all zero elements

```
B = [ 0 ] * 4
print B
[0,0,0,0]
```

Now you can enter data from the user as well.

```
A = makeL(8)
for i in range(8):
A[i] = input ("value?")
```

#### A matrix is a collection of lists

$$\begin{bmatrix}
 1, 3, 4 \\
 [ 2, 5, 6 ]\\
 [ 0, 1, 5 ]
 \end{bmatrix}$$

$$\begin{bmatrix}
 1, 3, 4 \\
 2, 5, 6 \\
 0, 1, 5
 \end{bmatrix}$$

### Create a 3x3 matrix with all elements being 4

```
matrix = []
for i in range (0, 3):
   row = []
   for j in range (0, 3):
      row.append(4)
   matrix.append(row)
```

#### Create a mxn matrix with all elements having value a

def initialM(m,n,a):
Zmatrix = [ [a for row in range(n)] for col in range(m)]
return Zmatrix

### Create a 3x5 matrix with all elements being 4

$$B = [ [4] * 5] * 3$$

uction of two vec

#### **Dot product of two vectors (Inner product)**

```
def inner_prod(A, B):
    sum = 0
    for i in range(len(A)):
        sum += A[i] * B[i]
    return sum
```

#### **Cross product of two vectors (Outer product)**

```
def outer_prod(A, B):
    n,m = len(A),len(B)
    C = [[0 for row in range (m)] for col in range (n)]
    for i in range(n):
        for j in range(m):
        C[i][j] = A[i] * B[j]
    return C
```

luction of two vec

#### To find Transpose of a n x m size matrix

```
\begin{pmatrix}
4 & 1 & 2 \\
3 & 7 & 4 \\
5 & 0 & 8
\end{pmatrix}

\begin{pmatrix}
4 & 3 & 5 \\
1 & 7 & 0 \\
2 & 4 & 8
\end{pmatrix}
```

```
def transpose(A, n, m):
    AT= [[0 for row in range (n)] for col in range (m)]
    for i in range( n ):
        for j in range( m ):
            AT [ j ] [ i ] = A [ i ] [ j ]
    return AT
```

luction of two vec

#### To add matrices of size n x m

```
def addMat(A, B, n, m):
    C= [[0 for row in range (m)] for col in range (n)]
    for i in range( n ):
        for j in range( m ):
            C [ i ] [ j ] = A [ i ] [ j ] + B[ i ] [ j ]
        return C
```

luction of two vec

#### To multiply matrix A(nxp) with B(pxm)

It is like taking dot product of each row of A (containing p elements) with each column of B (containing p elements)

# Strings in Python

#### String is a sequence

```
fruit = 'banana'
x = fruit [2]
n = len(fruit)
print x, n
n 6
last = fruit [n-1]
print last
a
for i in range (0, len(fruit), 2):
print fruit[ i ],
b n n
```

#### **Loops with String (as with lists)**

```
fruit = 'watermelon'
for letter in fruit :
    print letter,
w a t e r m e l o n
```

```
first = 'krsd'
last = 'ing'
for letter in first :
    print letter+last
king
ring
sing
ding
```

### String slicing

```
fruit = 'watermelon'
print fruit[ :5 ]
w a t e r
```

print fruit[5:] melon

You can form new strings by adding or deleting characters, but you cannot change the original string.

#### String Search – Look for a specific letter

```
fruit = 'wetermelone'
K = 0
for i in range (len (fruit):
   if fruit[ i ] == 'e':
       K = K+1
print "number of e = ", K
K = 0
for letter in fruit:
   if letter == 'e':
       K = K+1
print "number of e = ", K
4
```

# methods in Strings

#### Method to convert a String to capital letters

```
fruit = 'Wetermelon'
capital_fruit = fruit.upper()
print capital_fruit
WETERMELON
```

Using the "find" method

```
fruit.find( 'm')
5
```

```
fruit.find( 'e', 4)
# start search for e from index 4
6
```

#### Method for checking uppercase or lower case

```
fruit = 'Wetermelon'
if fruit[0].islower():
    print "first letter is not a capital letter"
```

# Function to check if a letter is common between two words

```
def common ( word1, word2):
    for letter in word1:
        if letter in word2:
            print letter,
```

common('ring', 'sing')
i n g

### **Comparison of strings (Alphabetical Order)**

if word1 < word2:
 print word1, 'comes before', word2</pre>

#### **Exercise**

- 1. Given two words print the longer word.
- 2. Count number of common letters in two words
- 3. Count number of words in a given string.
- 4. Given a word, get another word where all 'e's are replaced by 'i'.
- 5. Given a word, print it so that the first letter is in upper case form (Capital letter)
- 6. Write a function which returns True if given word contains the letter 'e', else it returns False.

#### Lists can also consist of strings

```
G = [ 'Amit', 'Vani', 'Sona', 'Jeet' ]
for name in G:
    print name[0],
A V S J
```

```
H = [ 'Amit', 5, ['Vani', 'Sona', 'Jeet'] ]
print len( H )
3

F = [ 'Amit', [5, 7 ], ['Vani', 'Sona', 'Jeet'] ]
print len( F )
3
```

#### Lists can also consist of strings

```
G = [ 'Amit', 'Vani', 'Sona', 'Jeet' ]

for name in G:
    print name[0],
A V S J
```

```
H = [ 'Amit', 5, ['Vani', 'Sona', 'Jeet'] ]
print len( H )

While strings cannot be changed, lists can be changed.
H[0] = 10
print H
[ 10, 5, ['Vani', 'Sona', 'Jeet'] ]
```