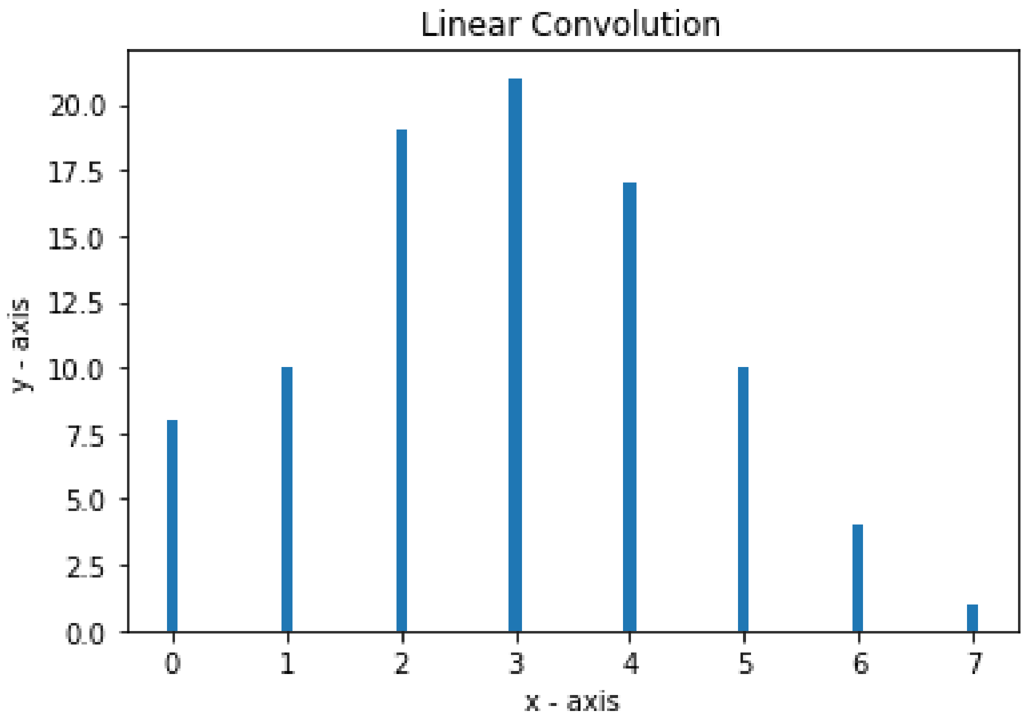
```python
#Question 1
#x(n) = [2,1,3,2,1], h(n) = [4,3,2,1]

import numpy as np

import matplotlib.pyplot as plt

# Creating two numpy One-Dimensional
# array using the array() method
arr1 = np.array([2, 1, 3, 2, 1])
arr2 = np.array([4, 3, 2, 1])

# Display the arrays
print("Array1:->\n", arr1)
print("\nArray2:->\n", arr2)


# To return the discrete linear convolution
# of two one-dimensional sequences,
# we use the numpy.convolve() method in Python Numpy
print("\nResult:->\n", np.convolve(arr1, arr2))


arr_final = np.convolve(arr1,arr2)

size = [0,1, 2, 3, 4, 5, 6, 7]

# plotting a bar chart
plt.bar(size, arr_final ,width = 0.1)

# naming the x-axis
plt.xlabel('x - axis')
# naming the y-axis
plt.ylabel('y - axis')
# plot title
plt.title('Linear Convolution')

# function to show the plot
plt.show()
```

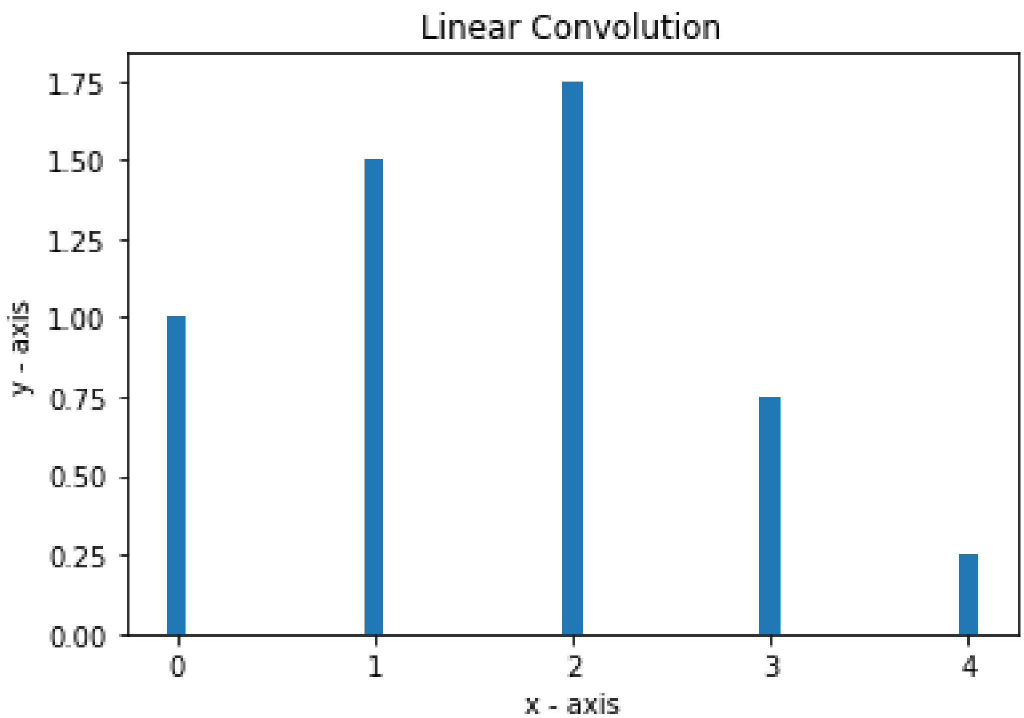
```
Array1:->
 [2 1 3 2 1]

Array2:->
 [4 3 2 1]

Result:->
 [ 8 10 19 21 17 10  4  1]
```




```
#Question 2
#x(n) = [1,1,1], h(n) = [1,0.5,0.25]

import numpy as np

# Creating two numpy One-Dimensional
# array using the array() method
arr1 = np.array([1,1,1])
arr2 = np.array([1,0.5,0.25])

# Display the arrays
print("Array1:->\n", arr1)
print("\nArray2:->\n", arr2)


# To return the discrete linear convolution
# of two one-dimensional sequences,
# we use the numpy.convolve() method in Python Numpy
print("\nResult:->\n", np.convolve(arr1, arr2))

arr_final = np.convolve(arr1,arr2)

size = [0,1, 2, 3, 4]


# plotting a bar chart
plt.bar(size, arr_final ,width = 0.1)
```

```
# naming the x-axis
plt.xlabel('x - axis')
# naming the y-axis
plt.ylabel('y - axis')
# plot title
plt.title('Linear Convolution')

# function to show the plot
plt.show()
```

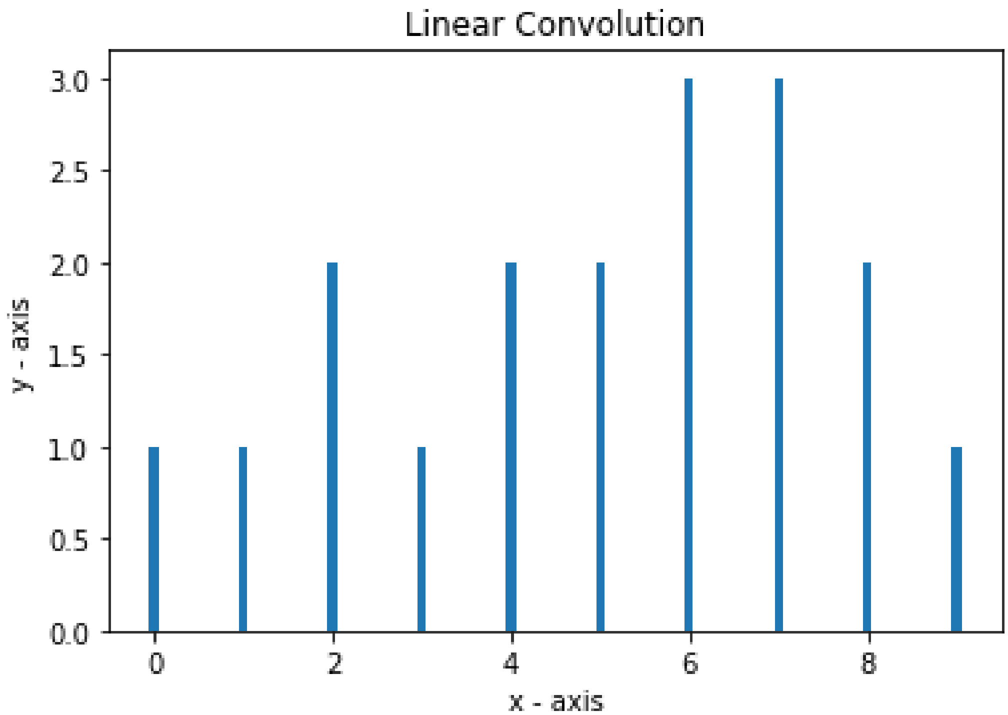
```
Array1:->
 [1 1 1]

Array2:->
 [1.   0.5  0.25]

Result:->
 [1.   1.5  1.75 0.75 0.25]
```




```
#Question 3
#x(n) = [1,0,1,0,1,1,1,1], h(n) = [1,1,1]

import numpy as np

# Creating two numpy One-Dimensional
# array using the array() method
arr1 = np.array([1,0,1,0,1,1,1,1])
arr2 = np.array([1,1,1])
```

```python
# Display the arrays
print("Array1:->\n", arr1)
print("\nArray2:->\n", arr2)


# To return the discrete linear convolution
# of two one-dimensional sequences,
# we use the numpy.convolve() method in Python Numpy
print("\nResult:->\n", np.convolve(arr1, arr2))

arr_final = np.convolve(arr1,arr2)

size = [0,1, 2, 3, 4, 5, 6, 7, 8, 9]

# plotting a bar chart
plt.bar(size, arr_final ,width = 0.1)

# naming the x-axis
plt.xlabel('x - axis')
# naming the y-axis
plt.ylabel('y - axis')
# plot title
plt.title('Linear Convolution')

# function to show the plot
plt.show()
```

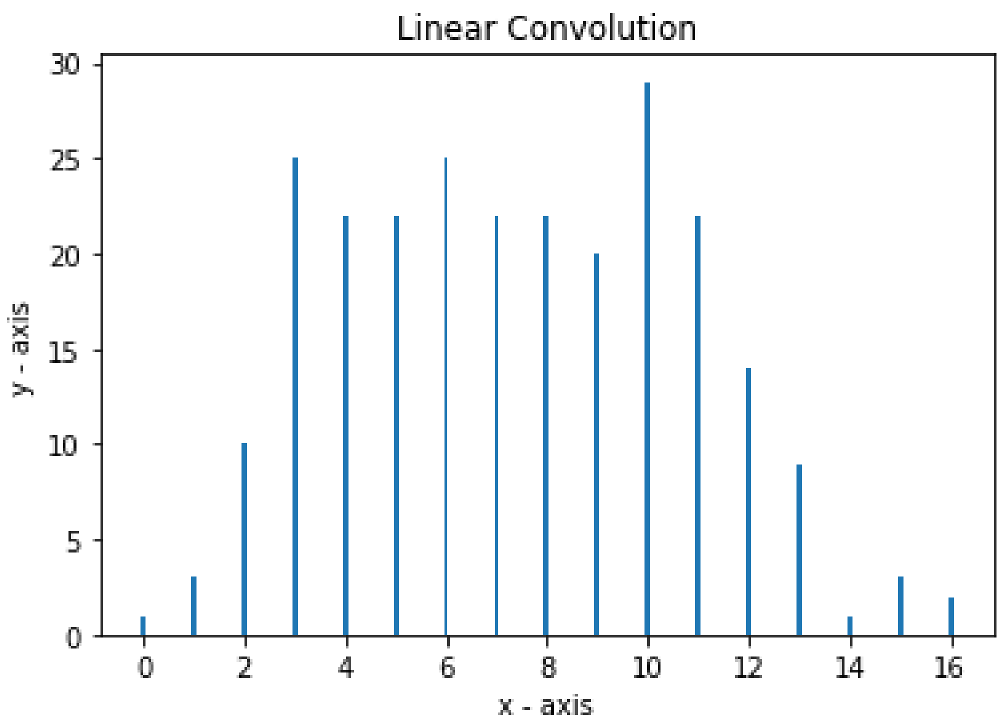
```
Array1:->
 [1 0 1 0 1 1 1 1]

Array2:->
 [1 1 1]

Result:->
 [1 1 2 1 2 2 3 3 2 1]
```




```
#Question 4
#x(n) = [1,3,2,1,2,2,1,1,3,2], h(n) = [1,0,8,0,4,0,0,1]

import numpy as np

# Creating two numpy One-Dimensional
# array using the array() method
arr1 = np.array([1,3,2,1,2,2,1,1,3,2])
arr2 = np.array([1,0,8,0,4,0,0,1])

# Display the arrays
print("Array1:->\n", arr1)
print("\nArray2:->\n", arr2)


# To return the discrete linear convolution
# of two one-dimensional sequences,
# we use the numpy.convolve() method in Python Numpy
print("\nResult:->\n", np.convolve(arr1, arr2))

arr_final = np.convolve(arr1,arr2)

size = [0,1, 2, 3, 4, 5, 6, 7, 8,9,10,11,12,13,14,15,16]

# plotting a bar chart
plt.bar(size, arr_final ,width = 0.1)

# naming the x-axis
```

```
plt.xlabel('x - axis')
# naming the y-axis
plt.ylabel('y - axis')
# plot title
plt.title('Linear Convolution')

# function to show the plot
plt.show()
```

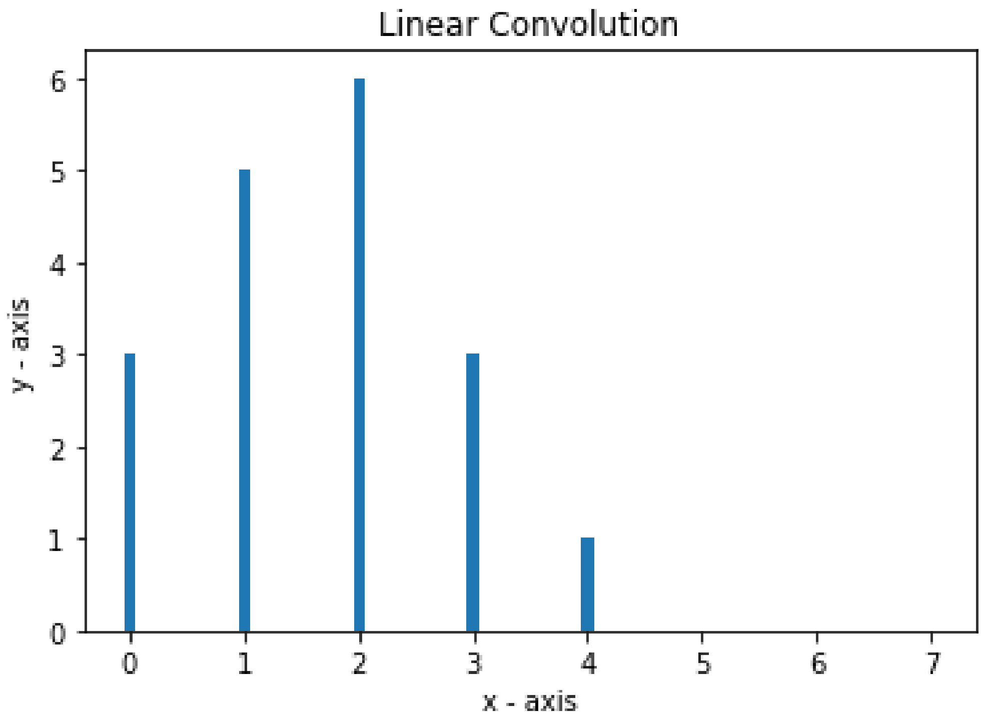
```
Array1:->
 [1 3 2 1 2 2 1 1 3 2]

Array2:->
 [1 0 8 0 4 0 0 1]

Result:->
 [ 1  3 10 25 22 22 25 22 22 20 29 22 14  9  1  3  2]
```




```
#Question 5
#x(n) = [3,2,1,0,0,0], h(n) = [1,1,1]

import numpy as np

# Creating two numpy One-Dimensional
# array using the array() method
arr1 = np.array([3,2,1,0,0,0])
arr2 = np.array([1,1,1])
```

```python
# Display the arrays
print("Array1:->\n", arr1)
print("\nArray2:->\n", arr2)


# To return the discrete linear convolution
# of two one-dimensional sequences,
# we use the numpy.convolve() method in Python Numpy
print("\nResult:->\n", np.convolve(arr1, arr2))

arr_final = np.convolve(arr1,arr2)

size = [0,1, 2, 3, 4, 5, 6, 7]


# plotting a bar chart
plt.bar(size, arr_final ,width = 0.1)

# naming the x-axis
plt.xlabel('x - axis')
# naming the y-axis
plt.ylabel('y - axis')
# plot title
plt.title('Linear Convolution')

# function to show the plot
plt.show()
```

```
Array1:->
 [3 2 1 0 0 0]

Array2:->
 [1 1 1]

Result:->
 [3 5 6 3 1 0 0 0]
```


Linear Convolution
y - axis
x - axis
0
1
2
3
4
5
6
7

SACHIN PRASANNA
211IT058

SIGNALS & SYSTEMS - ASSIGNMENT 4

1. $x(n) = \{2, 1, 3, 2, 1\}$

$h(n) = \{4, 3, 2, 1\}$

Ans

| $x(n)$ \ $h(n)$ | 4 | 3 | 2 | 1 |
|---|---|---|---|---|
| 2 | 8 | 6 | 4 | 2 |
| 1 | 4 | 3 | 2 | 1 |
| 3 | 12 | 9 | 6 | 3 |
| 2 | 8 | 6 | 4 | 2 |
| 1 | 4 | 3 | 2 | 1 |

Length of $y(n) = 5+4-1$
$= 8$

$y(n) = x(n) * h(n)$ (Adding all terms in the arrow)

So, $y(n) = \{8, 10, 19, 21, 17, 10, 4, 1\}$

2. $x(n) = \{1, 1, 1\}$

$h(n) = \{1, 0.5, 0.25\}$

Ans

| $x(n)$ \ $h(n)$ | 1 | 0.5 | 0.25 |
|---|---|---|---|
| 1 | 1 | 0.5 | 0.25 |
| 1 | 1 | 0.5 | 0.25 |
| 1 | 1 | 0.5 | 0.25 |

Length of $y(n) = 3+3-1$
$= 5$

$y(n) = x(n) * h(n)$ (Adding all terms in the arrow)

So, $y(n) = \{1, 1.5, 1.75, 0.75, 0.25\}$

3. $x(n) = \{1, 0, 1, 0, 1, 1, 1, 1\}$

$h(n) = \{1, 1, 1\}$

Ans

Length of $x(n) = 8$
Length of $h(n) = 3$
So, length of $y(n) = 8 + 3 - 1 = 10$

| $x(n)$ \ $h(n)$ | 1 | 1 | 1 |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |

$y(n) = x(n) * h(n)$ (Adding all terms in the arrow)

So, $y(n) = \{1, 1, 2, 1, 2, 2, 3, 3, 2, 1\}$

4. $x(n) = \{1, 3, 2, 1, 2, 2, 1, 1, 3, 2\}$

$h(n) = \{1, 0, 8, 0, 4, 0, 0, 1\}$

Ans

Length of $x(n) = 10$
Length of $h(n) = 8$

So, length of $y(n) = 10 + 8 - 1$
$= 17$

| x[n] \ h[n] | 1 | 0 | 8 | 0 | 4 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 8 | 0 | 4 | 0 | 0 | 1 |
| 3 | 3 | 0 | 24 | 0 | 12 | 0 | 0 | 3 |
| 2 | 2 | 0 | 16 | 0 | 8 | 0 | 0 | 2 |
| 1 | 1 | 0 | 8 | 0 | 4 | 0 | 0 | 1 |
| 2 | 2 | 0 | 16 | 0 | 8 | 0 | 0 | 2 |
| 2 | 2 | 0 | 16 | 0 | 8 | 0 | 0 | 2 |
| 1 | 1 | 0 | 8 | 0 | 4 | 0 | 0 | 1 |
| 1 | 1 | 0 | 8 | 0 | 4 | 0 | 0 | 1 |
| 3 | 3 | 0 | 24 | 0 | 12 | 0 | 0 | 3 |
| 2 | 2 | 0 | 16 | 0 | 8 | 0 | 0 | 2 |

$y[n] = x[n] * h[n]$ (Adding all term in the arrow)

So, $y[n] = \{1, 3, 10, 25, 22, 22, 25, 22, 22, 20, 29, 22, 14, 9, 1, 3, 2\}$

5. $x[n] = \{3, 2, 1, 0, 0, 0\}$

$h[n] = \{1, 1, 1\}$

Ans

length of $x[n] = 6$

length of $h[n] = 3$

So, length of $y[n] = 6 + 3 - 1$

$= 8$

P.T.O

| x(n) \ h(n) | 1 | 1 | 1 |
|---|---|---|---|
| 3 | 3 | 3 | 5 |
| 2 | 2 | 2 | 2 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 |

$$y(n) = x(n) * h(n)$$ (Adding all terms in the arrow)

So, $y(n) = \{3, 5, 6, 3, 1, 0, 0, 0\}$