NAME: Omkar Gajanan Phansopkar
CLASS: CSE(DS) D3
UID_NO: 2022701007
SUBJECT: FOSIP
EXPT-1 Discrete Convolution

AIM: The aim of this experiment is to study mathematical operations such as Linear Convolution, Circular Convolution, and Linear Convolution using Circular Convolution

---

Objective

- Develop a function to find Linear Convolution and Circular Convolution
- Calculate Linear convolution, Circular convolution, and Linear Convolution using Circular Convolution and verify the results using mathematical formulation.
- Conclude on aliasing effect in Circularconvolution

Problem Definition

- Find Linear Convolution and Circular Convolution of L point sequence x[n] and M point sequence h[n].

- Find Linear Convolution of L point sequence x[n] and M point sequence h[n] using Circular Convolution.

- Give your conclusion about No of values in Linearly Convolved signal, Aliasing effect in Circular Convolution.

Input Specifications :
1. Length of first Signal L and signal values.
2. Length of second Signal M and signal values.

Linear Convolution: Linear convolution is the standard convolution operation used to combine two functions or sequences, typically in a finite or infinite domain. It is defined using the convolution integral or sum, depending on whether you are working with continuous or discrete data, respectively.
Properties: Linear convolution obeys the commutative property, associative property, and distributive property. Linear convolution is generally used for linear time-invariant systems and filtering operations.

Applications: Signal processing for filtering, convolutional neural networks (CNNs) in deep learning, solving linear differential equations, and various other engineering and scientific applications.

Circular convolution: Circular Convolution, on the other hand, is a different type of convolution operation that is used when working with
periodic signals or functions. It assumes that the signals wrap around, creating a circular or periodic effect.
Properties: Circular convolution does not necessarily obey the commutative or associative properties. It is used for circular or periodically shifting signals and is especially relevant in applications involving circular or periodic data, like signal processing in communications or cyclic feature extraction.
Applications: Signal processing for cyclic or periodic signals, such as in communication systems (e.g., modulation and demodulation), circular convolution theorem in the frequency domain.

Difference:
● Linear convolution deals with non-periodic signals or functions over a finite or infinite domain, while circular convolution is used for periodic signals.
● Linear convolution follows standard mathematical properties, whereas circular convolution may not obey some of these properties.
● Linear convolution is widely used in various fields, while circular convolution is specific to applications involving periodic data.

- **Linear**

```
x = list(map(int, input("Enter values of x: ").split()))
h = list(map(int, input("Enter values of h: ").split()))
h.reverse()

for i in range(len(h) - 1):
x.insert(0, 0)
for i in range(len(x) - 1):
h.append(0)
arr = []
sum = -1
```

```
print("\n---------Values after padding--------")
print(x)
print(h)
print()
while (sum != 0):
sum = 0
for i in range(len(x)):
sum += x[i]*h[i]
h.pop()
h.insert(0, 0)
arr.append(sum)
arr.pop()
for i in range(len(arr)):
print(f"Y({i}) = ", arr[i])
```

```
assignments/fosip/exp1  master ✗
▶ python disc_conv.py
Enter values of x: 8 15 7 9 10 11
Enter values of h: 3 4 7

---------Values after padding--------
[0, 0, 8, 15, 7, 9, 10, 11]
[7, 4, 3, 0, 0, 0, 0, 0, 0, 0]

Y(0) =  24
Y(1) =  77
Y(2) =  137
Y(3) =  160
Y(4) =  115
Y(5) =  136
Y(6) =  114
Y(7) =  77

assignments/fosip/exp1  master ✗
```

- **Circular**

```
x = list(map(int, input("Enter values of x: ").split()))
h = list(map(int, input("Enter values of h: ").split()))
m = max(len(x), len(h))
for i in range(len(x), m):
x.insert(0, 0)
for i in range(len(h), m):
h.append(0)
a = h[1:]
a.reverse()
h[1:] = a
print("\n---------Values after padding--------")
print(x)
print(h)
print()
arr = []
sum = -1
i = 0
while (i < m):
sum = 0
for j in range(m):
sum += x[j]*h[j]
a = h.pop()
h.insert(0, a)
arr.append(sum)
print(f"Y({i}) = ", sum)
i += 1
```

```
assignments/fosip/exp1  master ✗
▶ python disc_conv_circ.py
Enter values of x: 8 15 7 9 10 11
Enter values of h: 3 4 7

---------Values after padding--------
[8, 15, 7, 9, 10, 11]
[3, 0, 0, 0, 7, 4]

Y(0) =  138
Y(1) =  154
Y(2) =  137
Y(3) =  160
Y(4) =  115
Y(5) =  136

assignments/fosip/exp1  master ✗
```

- **Linear using circular**

```
x = list(map(int, input("Enter values of x: ").split()))
h = list(map(int, input("Enter the values of h: ").split()))
m = max(len(x), len(h)) + min(len(x), len(h)) - 1
for i in range(len(x), m):
x.insert(i, 0)
for i in range(len(h), m):
h.append(0)
a = h[1:]
a.reverse()
h[1:] = a
arr = []
sum = -1
```

```python
i = 0
print("\n---------Values after padding--------")
print(x)
print(h)
print()
while (i < len(h)):
sum = 0
for j in range(len(x)):
sum += x[j]*h[j]
a = h.pop()
h.insert(0, a)
arr.append(sum)
print(f"Y({i}) = ", sum)
i += 1
```

```
assignments/fosip/exp1  master ✗
▶ python disc_conv_circ_us_linear.py
Enter the values of x: 8 15 7 9 10 11
Enter the values of h: 3 4 7

---------Values after padding--------
[8, 15, 7, 9, 10, 11, 0, 0]
[3, 0, 0, 0, 0, 0, 7, 4]

Y(0) =  24
Y(1) =  77
Y(2) =  137
Y(3) =  160
Y(4) =  115
Y(5) =  136
Y(6) =  114
Y(7) =  77

assignments/fosip/exp1  master ✗
▶
```

Audio Signal Filtering

Filter the Audio Signal Captured in the presence of noise and improve the quality of sound

1. Record Audio Password in the presence of noise ==> x[n].
2. Play the recorded signal x[n] and observe the quality of sound.
3. Design FIR Low Pass Filter using MATLAB filter design Tool. Take Fpass = 4000Hz. Fstop = 6000Hs Fs = 44000
4. Filter the audio signal x[n] i.e. Perform Linear Convolution of x[n] and h[n] ==>. Y[n]
5. Play the filtered signal [n] and observe the quality of sound

```
import numpy as np
import matplotlib.pyplot as plt
import soundfile as sf

custom_filter = np.array([-0.0003, 0.0000, 0.0003, 0.0005, 0.0006, 0.0005, 0.0002, -0.0003,
-0.0008, -0.0011,
-0.0012, -0.0008, 0.0000, 0.0010, 0.0019, 0.0024, 0.0020, 0.0008, -0.0009, -0.0028,
-0.0040, -0.0041, -0.0027, 0.0000, 0.0033, 0.0060, 0.0072, 0.0060, 0.0025, -0.0027,
-0.0079, -0.0113, -0.0114, -0.0074, 0.0000, 0.0089, 0.0165, 0.0197, 0.0167, 0.0069,
-0.0077, -0.0233, -0.0347, -0.0367, -
0.0255, 0.0000, 0.0375, 0.0817, 0.1252, 0.1599,
0.1791, 0.1791, 0.1599, 0.1252, 0.0817, 0.0375, 0.0000, -0.0255, -0.0367, -0.0347,
-0.0233, -0.0077, 0.0069, 0.0167, 0.0197, 0.0165, 0.0089, 0.0000, -0.0074, -0.0114,
-0.0113, -0.0079, -0.0027, 0.0025, 0.0060, 0.0072, 0.0060, 0.0033, 0.0000, -0.0027,
-0.0041, -0.0040, -0.0028, -0.0009, 0.0008, 0.0020, 0.0024, 0.0019, 0.0010, 0.0000,
-0.0008, -0.0012, -0.0011, -0.0008, -0.0003, 0.0002, 0.0005, 0.0006, 0.0005, 0.0003, 0.0000])

def custom_convolve(signal, custom_filter=custom_filter):
signal_len = len(signal)
filter_len = len(custom_filter)
output = np.zeros(signal_len)
```

```
for i in range(signal_len):
for j in range(filter_len):
if i - j >= 0:
output[i] += signal[i - j] * custom_filter[j]

return output



input_audio, sample_rate = sf.read('input_audio.wav')
print("Sample Rate:", sample_rate)

filtered = custom_convolve(input_audio)
filtered = filtered / np.max(np.abs(filtered))

sf.write('filtered.wav', filtered, sample_rate)
print("Convolution completed and saved to 'filtered.wav'")
```

```
assignments/fosip/exp1  master ✗
● ▶ python audio.py
Sample Rate: 48000
Convolution completed and saved to 'filtered.wav'

assignments/fosip/exp1  master ✗
```

Input audio file -
https://drive.google.com/file/d/1zlQlQ7semRdq5WOHxGuwcVfP87vh-Wkb/view?usp=sharing

Filtered audio file -
https://drive.google.com/file/d/1WOdgkoQ3GrFmcawaunFMynRhQQIZh3HJ/view?usp=sharing

CONCLUSION: In this experiment, we implemented Linear, circular and Linear using circular convolution and audio filtering using Linear convolution