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

AIM: The aim of this experiment is to study mathematical operation and correlation and measure the degree of similarity between two signals.

Objective

- Write a function to find Correlation Operation
- Calculate the correlation of DT signals and verify the results using a mathematical formula

Problem Definition

1. Find the autocorrelation of the input signal and find the significance of value of output signal at n=0. Let y[n] = x[n] O x[n] Classify the resultant signal(Even / Odd). Calculate the energy of the signal . Q. What is the significance of value of y[0].

2. Find auto correlation of delayed input signal. Let p[n]= x[n-1] O x[n-1]. Compare the resultant signal p[n] with y[n]. Give your conclusion.

3. Find cross correlation of input signal and delayed input signal q[n] = x[n] 0 x[n-1]. Compare the resultant signal q[n] with p[n] and y[n] Give your conclusion.

4. Find cross correlation of input signal and scaled input signal. Let s[n] = x[n] O a x[n-2] where “a” is any constant. Compare the resultant signals. Give your conclusion

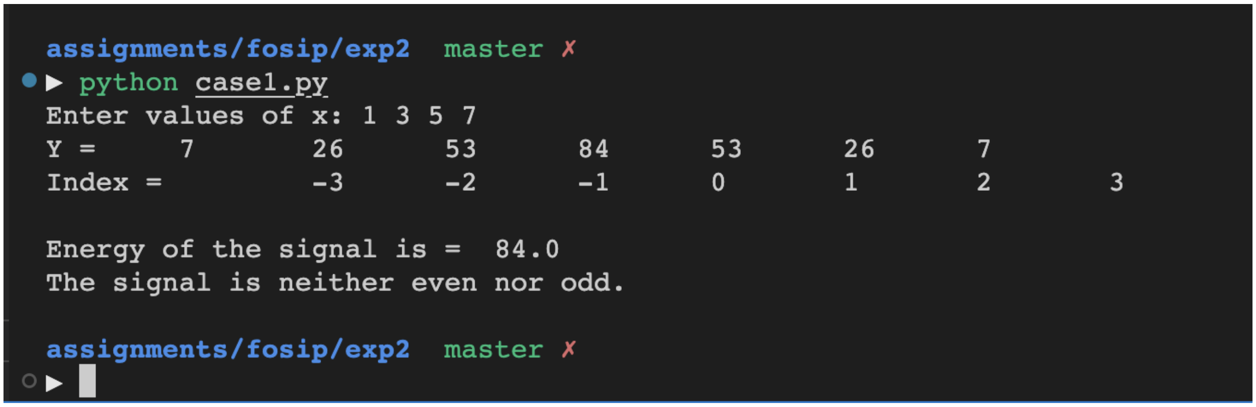
Input Specifications :

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

Case1

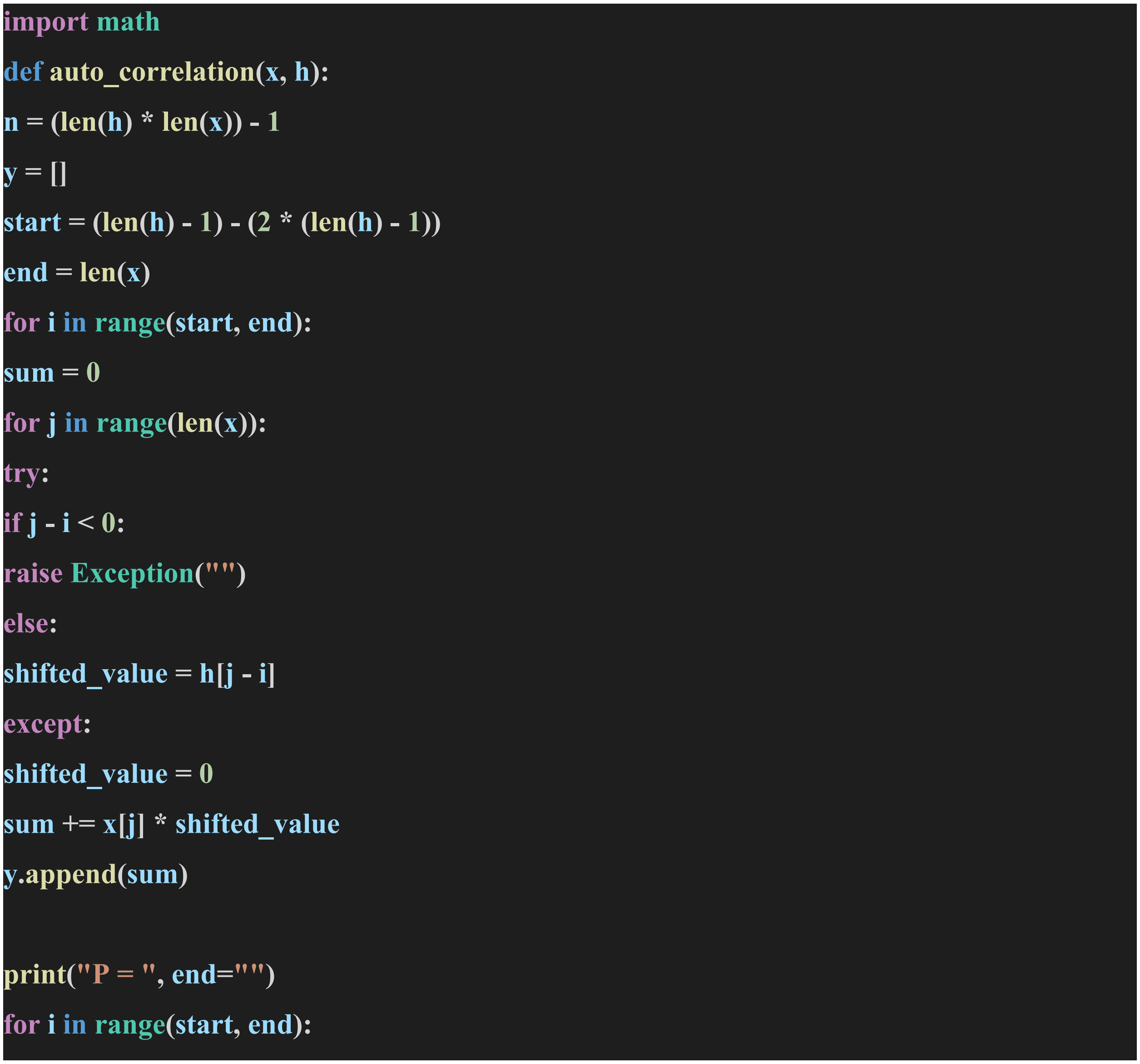
```
import math



def is_even(signal):
# Check if the signal is even
for n in range(len(signal)):
if signal[n] != signal[-n - 1]:
return False
return True



def is_odd(signal):
# Check if the signal is odd
for n in range(len(signal)):
if signal[n] != -signal[-n - 1]:
return False
return True



def auto_correlation(x, h):
n = (len(h) * len(x)) - 1
y = []
start = (len(h) - 1) - (2 * (len(h) - 1))
end = len(x)
for i in range(start, end):
sum = 0
for j in range(len(x)):
try:
if j - i < 0:
raise Exception("")
```

```
else:
shifted_value = h[j - i]
except:
shifted_value = 0
sum += x[j] * shifted_value
y.append(sum)
print("Y = ", end="")
for i in range(start, end):
print(f"\t{y[i + end - 1]}", end="")
print()
print("Index = ", end="")
for i in range(start, end):
print(f"\t{i}", end="")
print("\n")
return y


x = list(map(int, input("Enter values of x: ").split()))
y = auto_correlation(x, x)
energy = 0
for i in range(len(x)):
energy += math.pow(x[i], 2)

print("Energy of the signal is = ", energy)
if is_even(x):
print("The signal is even.")
elif is_odd(x):
print("The signal is odd.")
else:
print("The signal is neither even nor odd.")
```



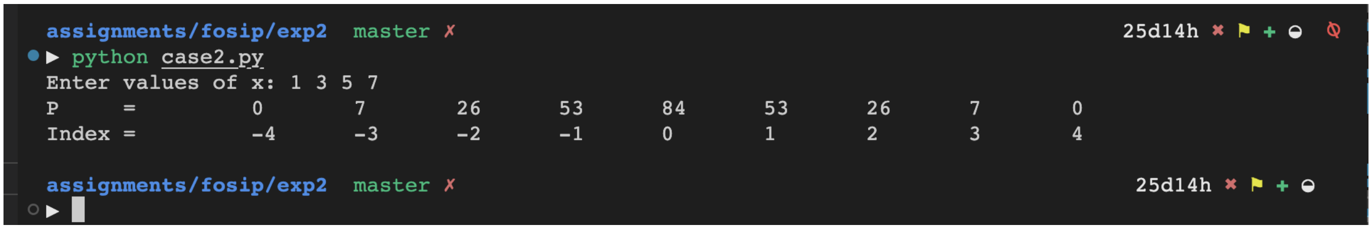
```
assignments/fosip/exp2  master ✗
● ▶ python case1.py
Enter values of x: 1 3 5 7
Y =     7       26      53      84      53      26      7
Index =         -3      -2      -1      0       1       2       3

Energy of the signal is =  84.0
The signal is neither even nor odd.

assignments/fosip/exp2  master ✗
○ ▶ █
```

Case2



```
import math
def auto_correlation(x, h):
n = (len(h) * len(x)) - 1
y = []
start = (len(h) - 1) - (2 * (len(h) - 1))
end = len(x)
for i in range(start, end):
sum = 0
for j in range(len(x)):
try:
if j - i < 0:
raise Exception("")
else:
shifted_value = h[j - i]
except:
shifted_value = 0
sum += x[j] * shifted_value
y.append(sum)

print("P = ", end="")
for i in range(start, end):
```

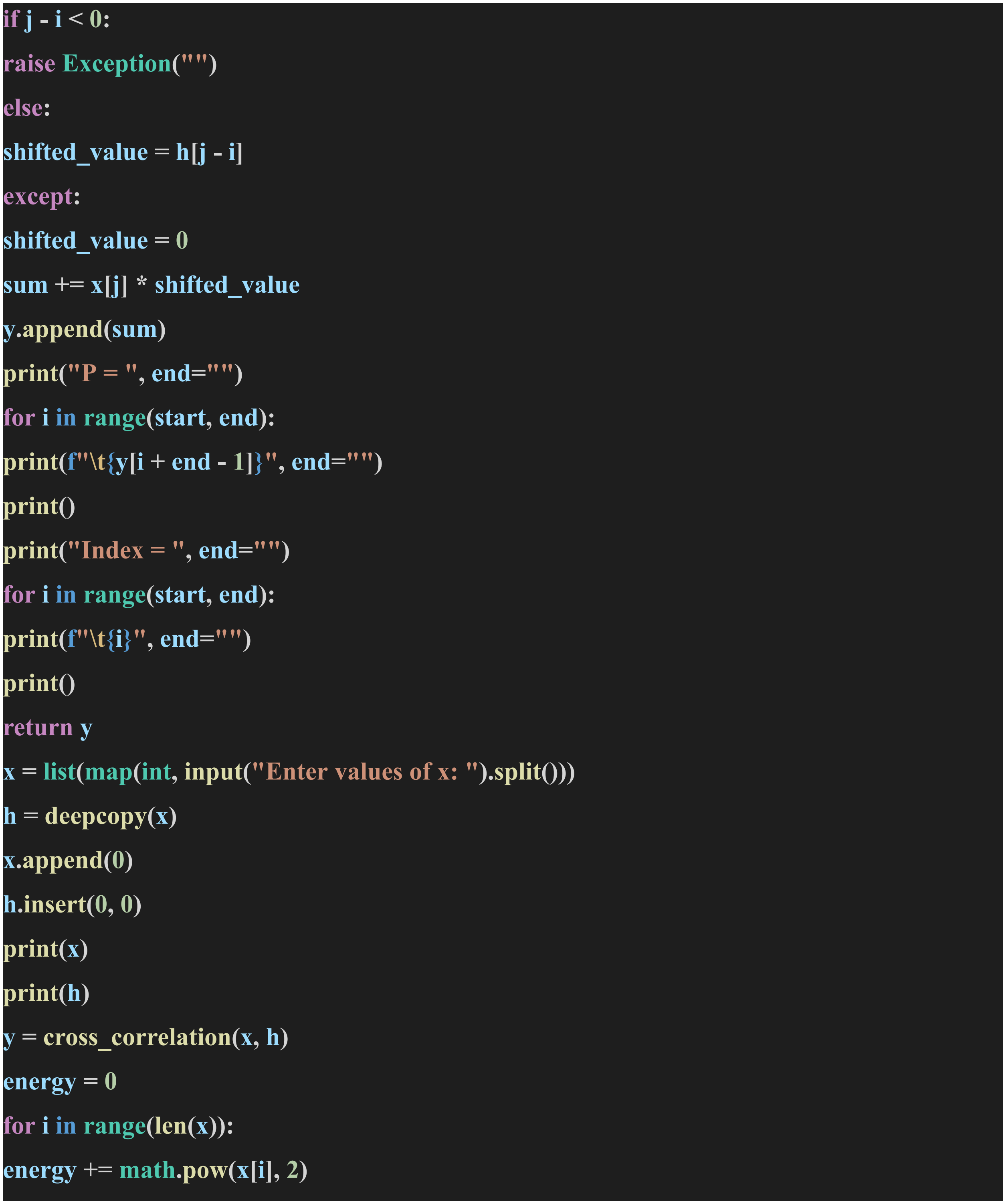
```
print(f"\t{y[i + end - 1]}", end="")
print()
print("Index = ", end="")
for i in range(start, end):
print(f"\t{i}", end="")
print()
return y
x = list(map(int, input("Enter values of x: ").split()))
x.insert(0, 0)
y = auto_correlation(x, x)
energy = 0
for i in range(len(x)):
energy += math.pow(x[i], 2)
```



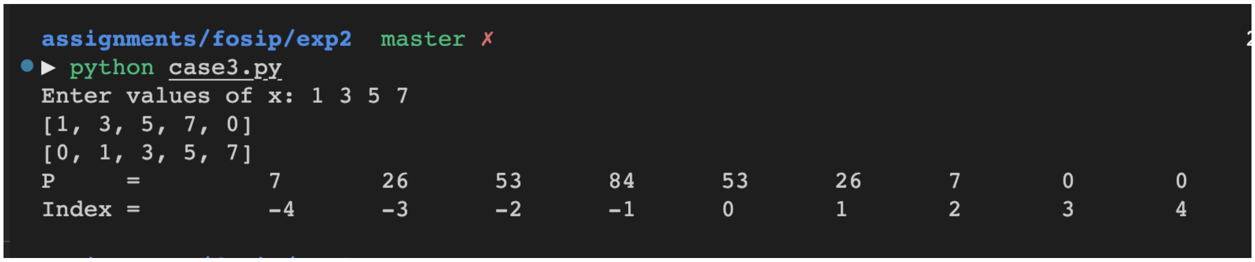
```
assignments/fosip/exp2  master ✗                                                  25d14h ✖ ⚑ ✚ ◒  ⦰
● ▶ python case2.py
Enter values of x: 1 3 5 7
P     =         0       7       26      53      84      53      26      7       0
Index =         -4      -3      -2      -1      0       1       2       3       4

assignments/fosip/exp2  master ✗                                                   25d14h ✖ ⚑ ✚ ◒
○ ▶ █
```

Case 3

```
import math
from copy import deepcopy
def cross_correlation(x, h):
n = (len(h) * len(x)) - 1
y = []
start = (len(h) - 1) - (2 * (len(h) - 1))
end = len(x)
for i in range(start, end):
sum = 0
for j in range(len(x)):
try:
```



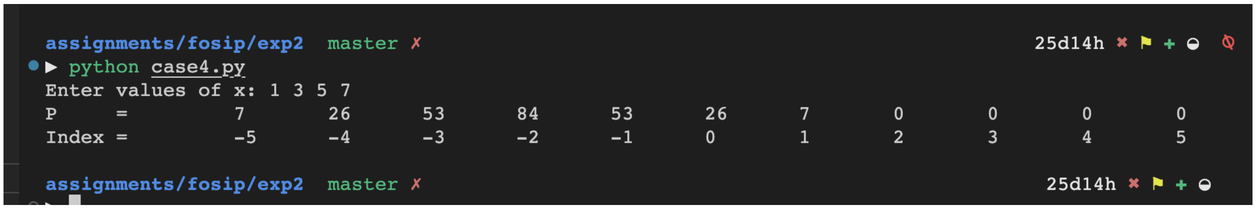
```
if j - i < 0:
raise Exception("")
else:
shifted_value = h[j - i]
except:
shifted_value = 0
sum += x[j] * shifted_value
y.append(sum)
print("P = ", end="")
for i in range(start, end):
print(f"\t{y[i + end - 1]}", end="")
print()
print("Index = ", end="")
for i in range(start, end):
print(f"\t{i}", end="")
print()
return y
x = list(map(int, input("Enter values of x: ").split()))
h = deepcopy(x)
x.append(0)
h.insert(0, 0)
print(x)
print(h)
y = cross_correlation(x, h)
energy = 0
for i in range(len(x)):
energy += math.pow(x[i], 2)
```



```
assignments/fosip/exp2  master ✗
● ▶ python case3.py
Enter values of x: 1 3 5 7
[1, 3, 5, 7, 0]
[0, 1, 3, 5, 7]
P      =        7       26      53      84      53      26      7       0       0
Index  =        -4      -3      -2      -1      0       1       2       3       4
```

Case 4

```
import math
from copy import deepcopy



def cross_correlation(x, h):
n = (len(h) * len(x)) - 1
y = []
start = (len(h) - 1) - (2 * (len(h) - 1))
end = len(x)
for i in range(start, end):
sum = 0
for j in range(len(x)):
try:
if j - i < 0:
raise Exception("")
else:
shifted_value = h[j - i]
except:
shifted_value = 0
sum += x[j] * shifted_value
y.append(sum)
print("P = ", end="")
for i in range(start, end):
print(f"\t{y[i + end - 1]}", end="")
print()
print("Index = ", end="")
for i in range(start, end):
print(f"\t{i}", end="")
print()
return y


x = list(map(int, input("Enter values of x: ").split()))
h = deepcopy(x)
x.append(0)
x.append(0)
h.insert(0, 0)
h.insert(0, 0)
y = cross_correlation(x, h)
energy = 0
for i in range(len(x)):
energy += math.pow(x[i], 2)
```



```
assignments/fosip/exp2  master ✗                                                  25d14h ✖ ⚑ ✚ ◒  
● ▶ python case4.py
Enter values of x: 1 3 5 7
P     =        7       26      53      84      53      26      7       0       0       0       0
Index =        -5      -4      -3      -2      -1      0       1       2       3       4       5

assignments/fosip/exp2  master ✗                                                  25d14h ✖ ⚑ ✚ ◒
```

Audio Signal Filtering

Authentication using Audio Password Verification

Authenticate the user by measuring the degree of similarity between stored audio Password and Test Audio Password

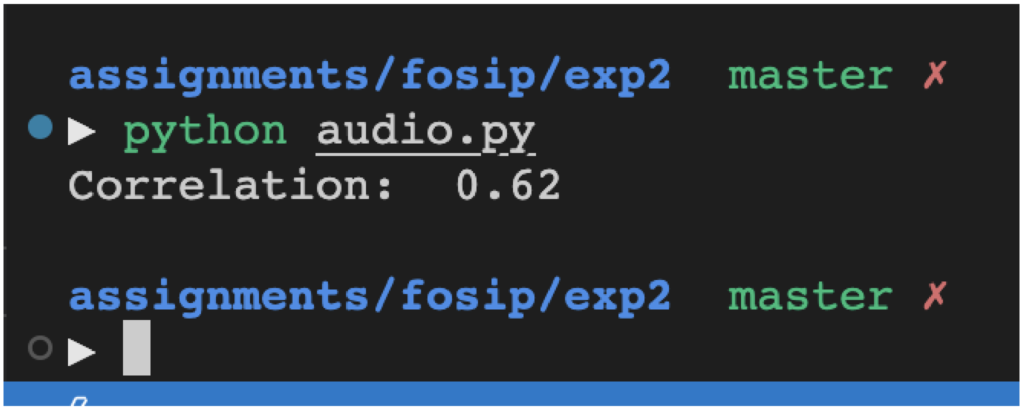
Algorithm:

1. Record Audio Password and filter the noise ==> x[n].

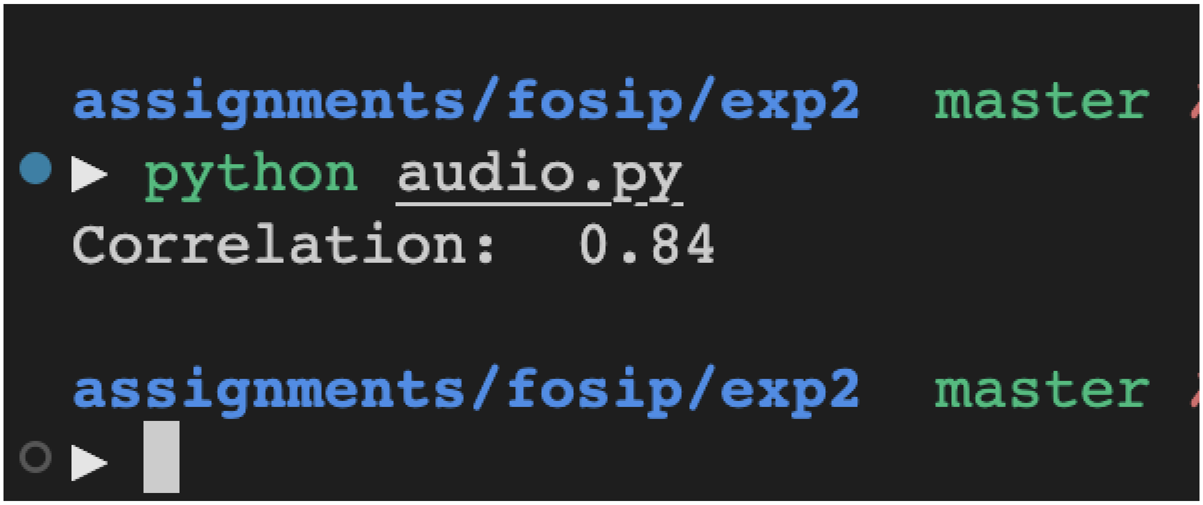
2. Play the recorded Audio signal x[n].

3. Record Test Audio Password and filter the noise ==> y[n].
4. Play the recorded Test Audio signal y[n].
5. Calculate Coefficient of Correlation ==>. r
6. Authenticate the user by selecting appropriate Threshold value (Anything > 0.9).

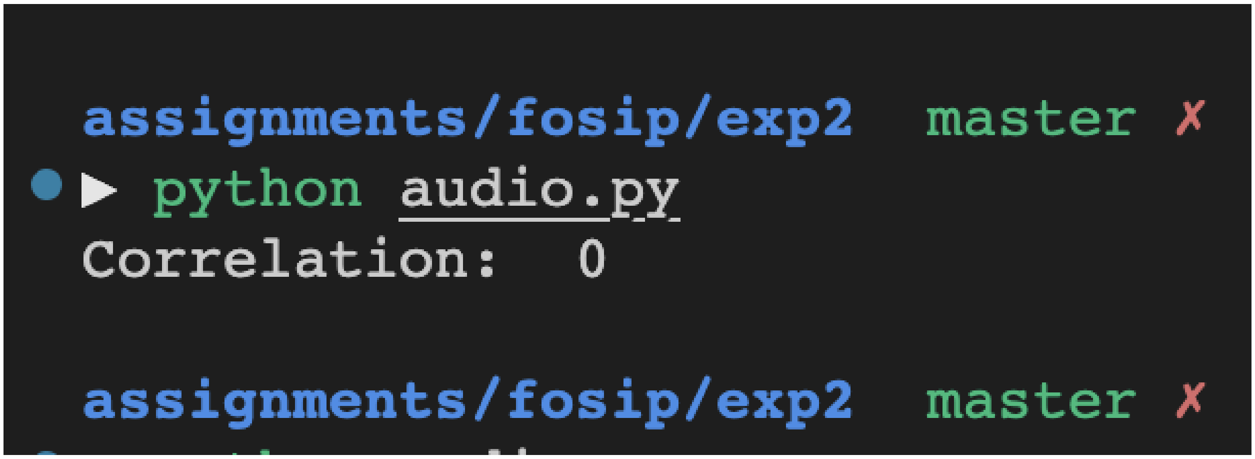
```
import numpy as np
import soundfile as sf
from exp1 import custom_convolve


def Pearson_correlation(X, Y):
corr = 0
if len(X) == len(Y):
Sum_xy = sum((X - X.mean()) * (Y - Y.mean()))
Sum_x_squared = sum((X - X.mean()) ** 2)
Sum_y_squared = sum((Y - Y.mean()) ** 2)
corr = Sum_xy / np.sqrt(Sum_x_squared * Sum_y_squared)
return corr


input_audio1, sample_rate1 = sf.read("pass1.wav")
input_audio2, sample_rate2 = sf.read("pass2.wav")
input_audio1 = custom_convolve(input_audio1)
input_audio2 = custom_convolve(input_audio2)
print("Correlation: ", Pearson_correlation(input_audio1, input_audio2))
```



```
  assignments/fosip/exp2  master ✗
● ▶ python audio.py
  Correlation:  0.62

  assignments/fosip/exp2  master ✗
○ ▶
```



```
  assignments/fosip/exp2  master
● ▶ python audio.py
  Correlation:  0.84

  assignments/fosip/exp2  master
○ ▶
```



```
  assignments/fosip/exp2  master ✗
● ▶ python audio.py
  Correlation:  0

  assignments/fosip/exp2  master ✗
```

CONCLUSION: In this experiment, we learned to perform correlation on signals and also implemented an authentication system by measuring the degree of similarity between audio passwords