# NATIONAL INSTITUTE OF TECHNOLOGY SURATHKAL

# MANGALORE, KARNATAKA-575025

## LAB ASSIGNMENT :-06


NAME :- CHIKKERI CHINMAYA

ROLL NO :- 211IT017

COURSE :- B.TECH (INFORMATION TECHNOLOGY)

SUBJECT :- IT204 (SIGNALS AND SYSTEMS LAB)

SUBMITTED TO :-

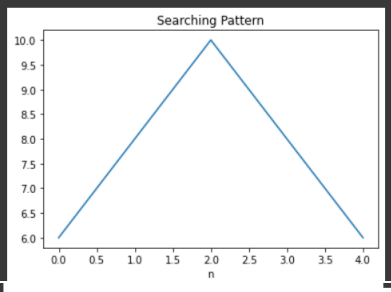
- REVANESHA M SIR

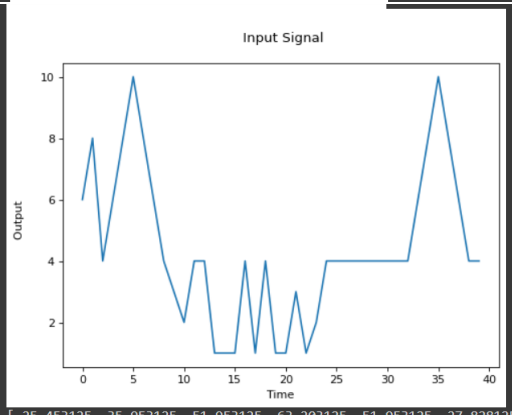
**Q1: Use the concepts learnt so far in Signals and Systems to perform pattern matching on the given signal. Given a complex input signal and a search pattern, determine if the pattern exists within the signal. Write a program to implement this functionality for both single match and multiple match scenarios for**

a) Simple input signal (as shown in reference example)

```
[ ] import matplotlib.pyplot as plt
    import numpy as np
    from matplotlib.pyplot import figure


    area = np.arange(0,5,1);

    subset = np.array([6,8,10,8,6])

    plt.plot(area,subset)
    plt.title(' Searching Pattern ')
    plt.xlabel('n')
    plt.show()

    area2 = np.arange(1,41,1)
    some_data = np.array([6,8,4,6,8,10,8,6,4,3,2,4,4,1,1,1,4,1,4,1,1,3,1,2,4,4,4,4,4,4,4,4,4,6,8,10,8,6,4,4])
    figure(figsize=(7, 5), dpi=80)
    plt.title('\n Input Signal\n')
    plt.xlabel('Time')
    plt.ylabel('Output  \n')
    curve, = plt.plot(some_data)
    plt.show()

    mean = np.mean(some_data)
    some_data_normalised = some_data - mean
    subset_normalised = subset - mean
    correlated = np.correlate(some_data_normalised, subset_normalised)
    print(correlated)
```

```
    max_index = np.argmax(correlated)
    a = correlated[max_index]
    i, = np.where(correlated == a)
    print(i+1)
```



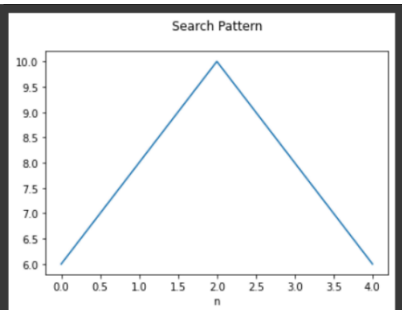



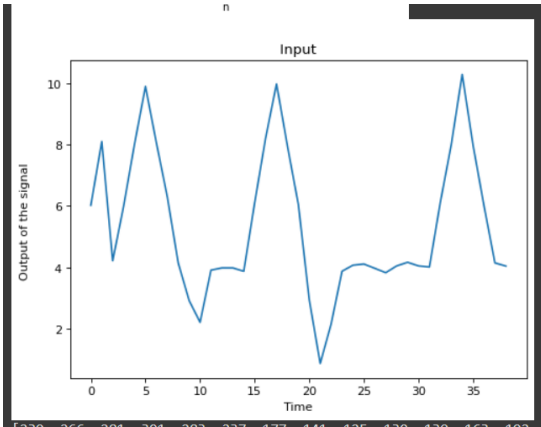

```
[ 25.453125  35.953125  51.953125  63.203125  51.953125  27.828125
   0.828125 -15.671875 -20.921875 -19.796875 -25.046875 -38.671875
 -44.671875 -43.546875 -32.671875 -32.671875 -32.671875 -40.296875
 -42.296875 -41.546875 -38.671875 -29.796875 -18.171875  -9.296875
  -6.046875  -6.046875  -6.046875  -6.046875  -6.046875  -2.796875
   7.703125  29.453125  51.953125  63.203125  51.953125  29.453125]
[ 4 34]
```

b) Noisy input signal ( ex: x = x+ np.random.normal(0,0.1,len(x)), where x is the original input signal)

```
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.pyplot import figure
area = np.arange(0,5,1);
subset = np.array([6,8,10,8,6])


plt.plot(area,subset)
plt.title(' Search Pattern \n')
plt.xlabel('n')
plt.show()


area2 = np.arange(1,41,1)
data = np.array([6,8,4,6,8,10,8,6,4,3,2,4,4,4,4,6,8,10,8,6,3,1,2,4,4,4,4,4,
4,4,4,6,8,10,8,6,4,4])
data = data+ np.random.normal(0,0.1,len(data))
figure(figsize=(7, 5), dpi=80)
plt.title('\n Input ')
plt.xlabel('Time')
plt.ylabel('Output of the signal \n')
curve, = plt.plot(data)
plt.show()
normalised = data
subset_normalised = subset
correlated = np.correlate(normalised, subset_normalised)
correlated = np.around(correlated)
print(correlated)
```

```
max_index = np.argmax(correlated)
MAX = correlated[max_index]
error = 2*a*0.1
```




```
[239. 266. 281. 301. 283. 237. 177. 141. 125. 130. 139. 163. 192. 241.
 282. 302. 275. 213. 145. 109. 101. 118. 141. 151. 152. 152. 152. 153.
 167. 194. 243. 283. 303. 283. 243.]
```