Elementary Signals

to plot elementary signals in both continuous and discrete domain.

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

t1=np.arange(-10,10,0.01)
t2=np.arange(-10,10,1)
```

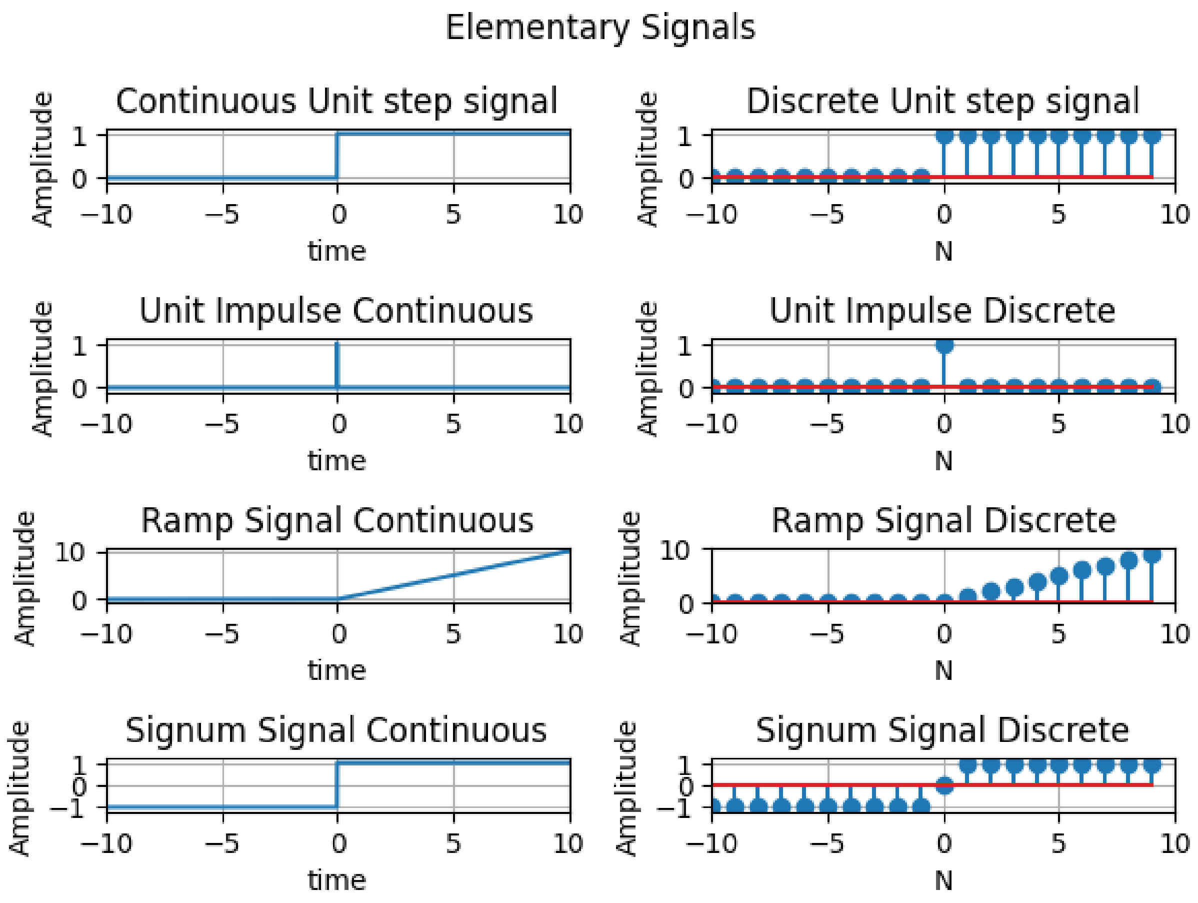
TASK 1:

- To plot continuous and discrete unit functions/signals
- To plot continuous and discrete unit impulse functions/signals
- To plot continuous and discrete ramp functions/signals
- To plot continuous and discrete signum functions/signals

```
#for continuous unit signal
unit_step_continuous=np.zeros_like(t1)
unit_step_continuous[t1>=0]=1
plt.subplot(4,2,1)
plt.suptitle("Elementary Signals")
plt.title("Continuous Unit step signal")
plt.plot(t1,unit_step_continuous)
plt.xlabel('time')
plt.ylabel('Amplitude')
plt.grid()
# plt.axhline(y=0,color="black",linestyle="--")
# plt.axvline(x=0,color="black",linestyle="--")
plt.xlim(-10,10)
plt.ylim(-0.1,1.1)

# for discrete unit signal
unit_step_discrete=np.zeros_like(t2)
unit_step_discrete[t2>=0]=1
plt.subplot(4,2,2)
plt.title("Discrete Unit step signal")
plt.stem(t2,unit_step_discrete)
plt.grid()
plt.xlabel("N")
plt.ylabel("Amplitude")
plt.xlim(-10,10)
plt.ylim(-0.1,1.1)
```

```python
#for continuous unit impulse signal
unit_impulse_signal_continuous=np.zeros_like(t1)
unit_impulse_signal_continuous[np.isclose(t1,0)]=1
plt.subplot(4,2,3)
plt.plot(t1,unit_impulse_signal_continuous)
plt.title("Unit Impulse Continuous")
plt.grid()
plt.xlabel("time")
plt.ylabel("Amplitude")
plt.xlim(-10,10)
plt.ylim(-0.1,1.1)

# for discrete unit impulse signal
unit_impulse_signal_discrete=np.zeros_like(t2)
unit_impulse_signal_discrete[t2==0]=1
plt.subplot(4,2,4)
plt.stem(t2,unit_impulse_signal_discrete)
plt.title("Unit Impulse Discrete")
plt.grid()
plt.xlabel("N")
plt.ylabel("Amplitude")
plt.xlim(-10,10)
plt.ylim(-0.1,1.1)

#for continuous ramp signal

ramp_signal_continuous=np.zeros_like(t1)
ramp_signal_continuous=t1*(t1>=0).astype(int)
plt.subplot(4,2,5)
plt.plot(t1,ramp_signal_continuous)
plt.title("Ramp Signal Continuous")
plt.grid()
plt.xlabel("time")
plt.ylabel("Amplitude")
plt.xlim(-10,10)



#for discrete ramp signal

ramp_signal_discrete=np.zeros_like(t2)
ramp_signal_discrete=t2*(t2>=0).astype(int)
plt.subplot(4,2,6)
plt.stem(t2,ramp_signal_discrete)
plt.title("Ramp Signal Discrete")
plt.grid()
plt.xlabel("N")
plt.ylabel("Amplitude")
plt.xlim(-10,10)
plt.ylim(0,10)
```

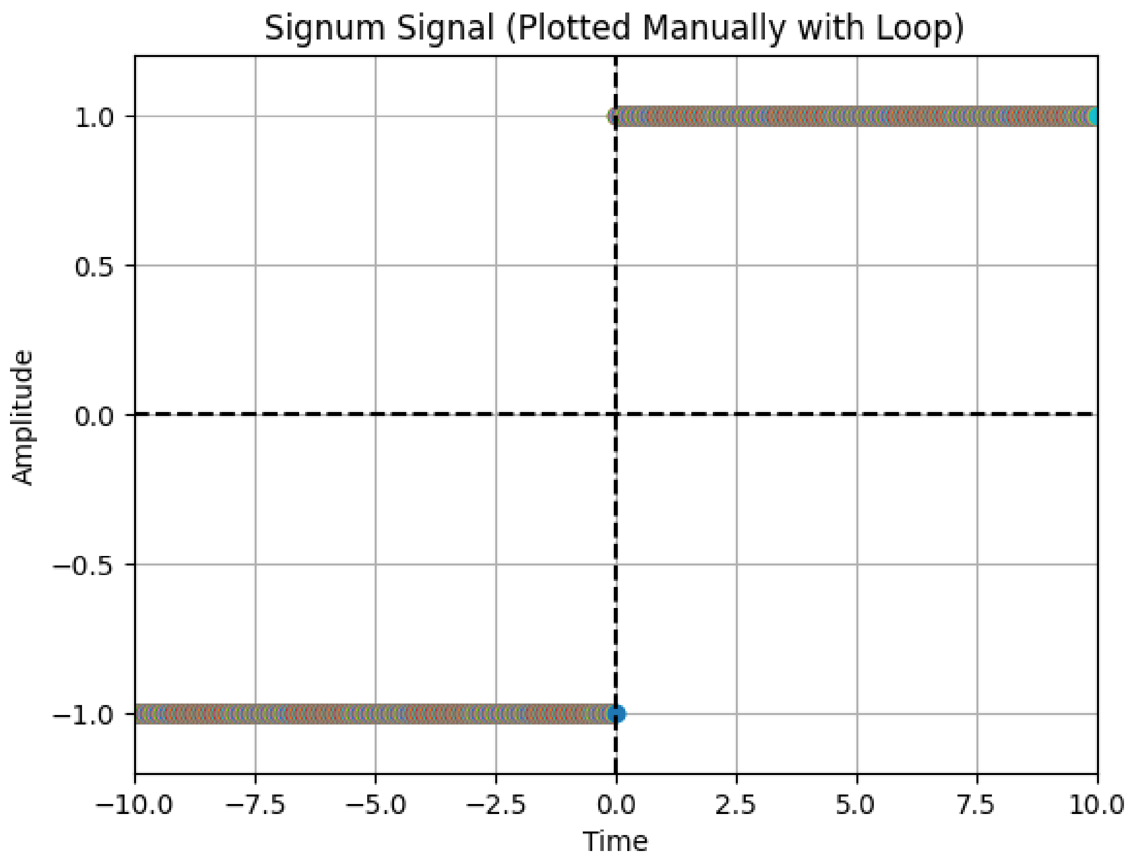
```python

# for continuous signum function/signal
signum_signal_continous=np.sign(t1)
plt.subplot(4,2,7)
plt.plot(t1,signum_signal_continous)
plt.title("Signum Signal Continuous")
plt.grid()
plt.xlabel("time")
plt.ylabel("Amplitude")
plt.xlim(-10,10)
plt.ylim(-1.2,1.2)


# for discrete signum function/signal
signum_signal_discrete=np.sign(t2)
plt.subplot(4,2,8)
plt.stem(t2,signum_signal_discrete)
plt.title("Signum Signal Discrete")
plt.grid()
plt.xlabel("N")
plt.ylabel("Amplitude")
plt.xlim(-10,10)
plt.ylim(-1.2,1.2)




plt.tight_layout()
plt.show()
```



TASK 2:

- to plot signum using loop

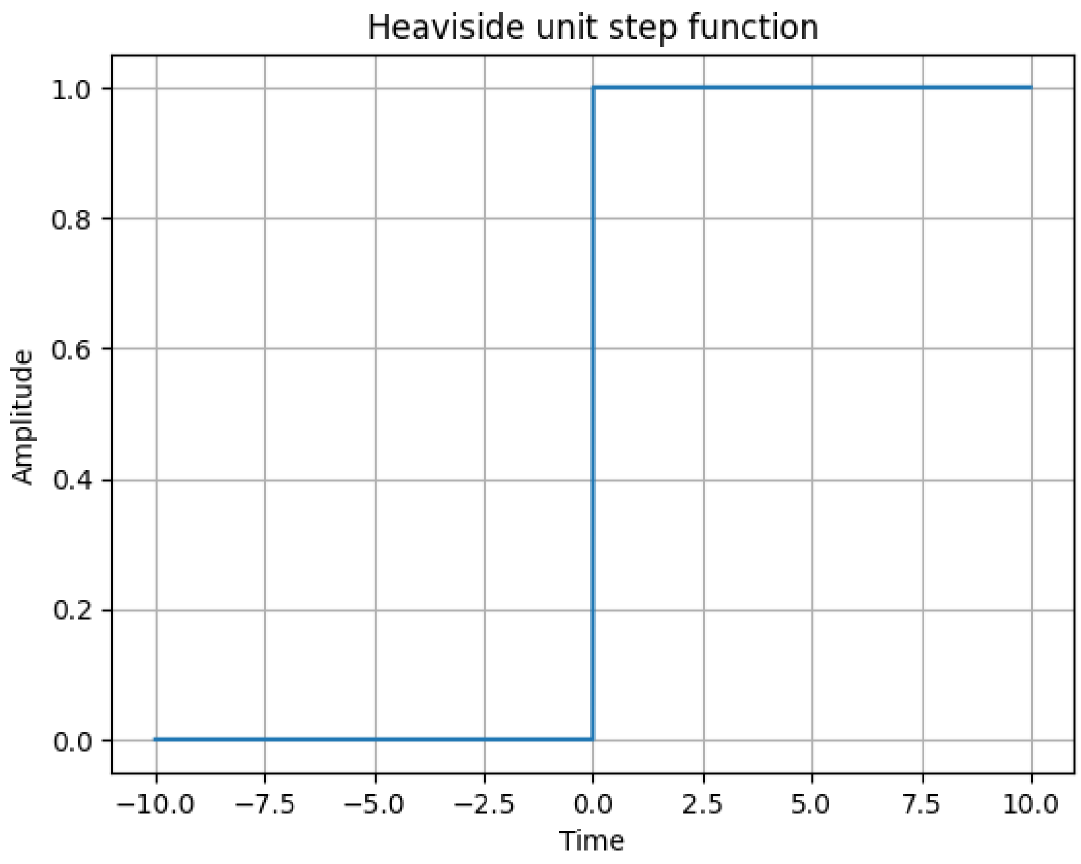
```
for value in t1:
    if value<0:
        plt.plot(value,-1,'o')
    elif value>0:
        plt.plot(value,1,'o')
    elif np.isclose(value,0,'o'):
        plt.plot(value,0)
    else:
        print(f"Not a number {value} ")



plt.title("Signum Signal (Plotted Manually with Loop)")
plt.xlabel("Time")
plt.ylabel("Amplitude")
plt.grid(True)
plt.axhline(0, color='black', linestyle='--')
plt.axvline(0, color='black', linestyle='--')
plt.xlim(-10, 10)
plt.ylim(-1.2, 1.2)
plt.show()
```



TASK 3:

- to explore the use of heaviside

```
unit_step_signal_continuous_heaviside=np.heaviside(t1,0.5)
plt.plot(t1,unit_step_signal_continuous_heaviside)
plt.title("Heaviside unit step function")
plt.grid()
plt.xlabel("Time")
plt.ylabel("Amplitude")
plt.show()
```


Heaviside unit step function
Amplitude
1.0
0.8
0.6
0.4
0.2
0.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
Time