

**Advance Python Programming**

**MCA-372**

**Assignment – 04**

***BY***

**HIMANSHU HEDA (24225013)**

**SUBMITTED TO**

**Dr. Manjula Shannhog**

**SCHOOL OF SCIENCES**

**2024-25**

**Animations :**

**Star animation**

import pandas as pd

import matplotlib.animation as animation

from IPython.display import HTML

plt.ioff()

x = [1,2,3]

y = [1,2,3]

fig = plt.figure(figsize=(4,4))

ax = fig.add\_subplot(111)

ax.set\_xlim(0,4)

ax.set\_ylim(0,4)

point, = ax.plot([],[],'b\*')

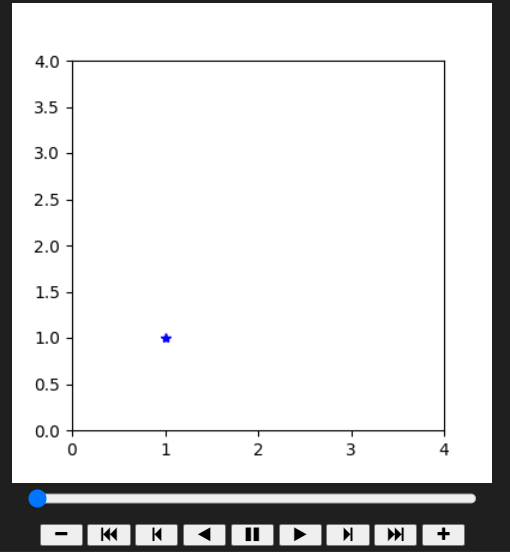
def fun(frame):

    point.set\_data([x[frame]],[y[frame]])

    return point,

a = animation.FuncAnimation(fig,fun,frames=np.arange(0,3),interval=200)

HTML(a.to\_jshtml())

****

**Circle animation**

import matplotlib.pyplot as plt

import numpy as np

import matplotlib.animation as animation

from IPython.display import HTML

plt.ioff()

fig, ax = plt.subplots()

ax.set\_xlim(0, 150)

ax.set\_ylim(0, 10)

x = np.linspace(0, 150, 100)

y = np.linspace(0, 10, 100)

point, = ax.plot([], [], 'o', markersize=10)

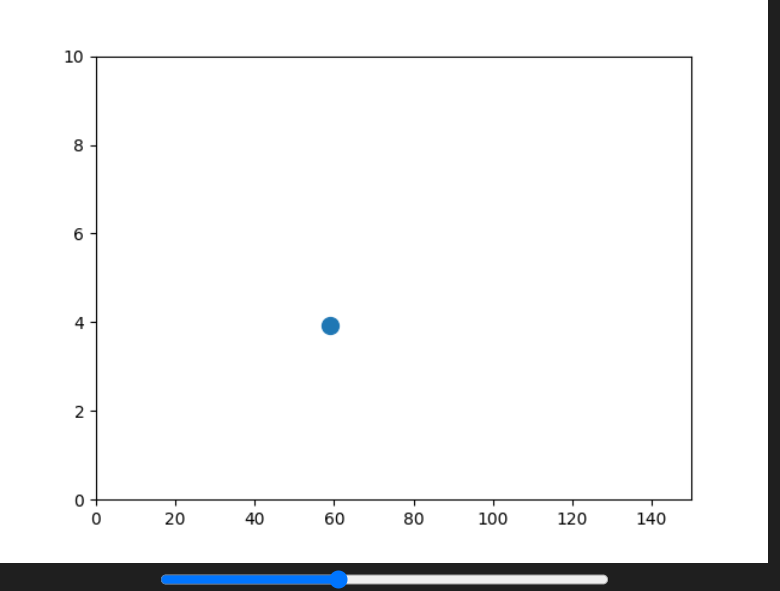
def funcani(frame):

    point.set\_data([x[frame]], [y[frame]])

    return point,

a = animation.FuncAnimation(fig, funcani, frames=np.arange(0, 100, 1), interval=20, blit=False)

HTML(a.to\_jshtml())

****

**Dot Line animation**

import matplotlib.pyplot as plt

import numpy as np

import matplotlib.animation as animation

from IPython.display import HTML

plt.ioff()

fig,ax = plt.subplots()

ax.set\_xlim(0,100)

ax.set\_ylim(0,100)

x = []

y = []

line, = ax.plot([],[],'ro',linewidth=2)

def fun\_line(frame):

    x.append(frame)

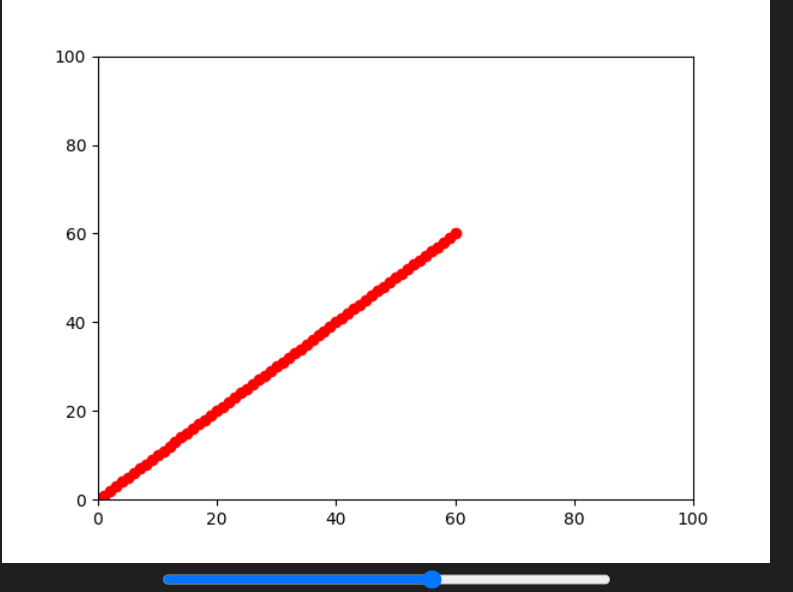
    y.append(frame)

    line.set\_data(x,y)

    return line,

a = animation.FuncAnimation(fig,fun\_line,frames=np.arange(0,100,1),interval=10)

HTML(a.to\_jshtml())

****

**Curve Graph :**

import numpy as np

import matplotlib.pyplot as plt

plt.ioff()

x = np.arange(0,200,1)

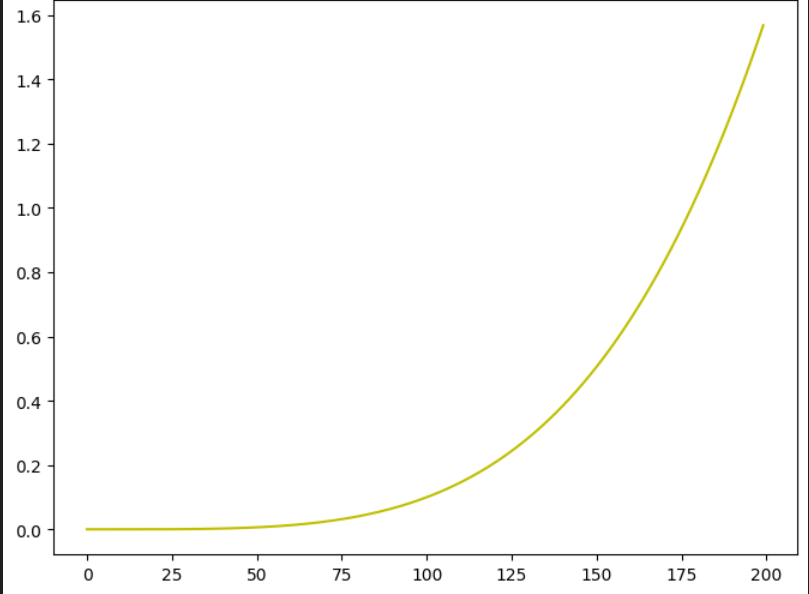
y = x\*\*4

fig = plt.figure(figsize=(8,6))

ax = fig.add\_subplot(111)

ax.plot(x,y,'y-')

plt.show()

****

**Animation curve graph**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.animation as animation

from IPython.display import HTML

plt.ioff()

x = np.arange(0, 200, 1)

y = x\*\*4

fig = plt.figure(figsize=(8, 6))

ax = fig.add\_subplot(111)

line, = ax.plot([], [], 'y-', linewidth=2)

ax.set\_xlim(0, 200)

ax.set\_ylim(0, y[-1])

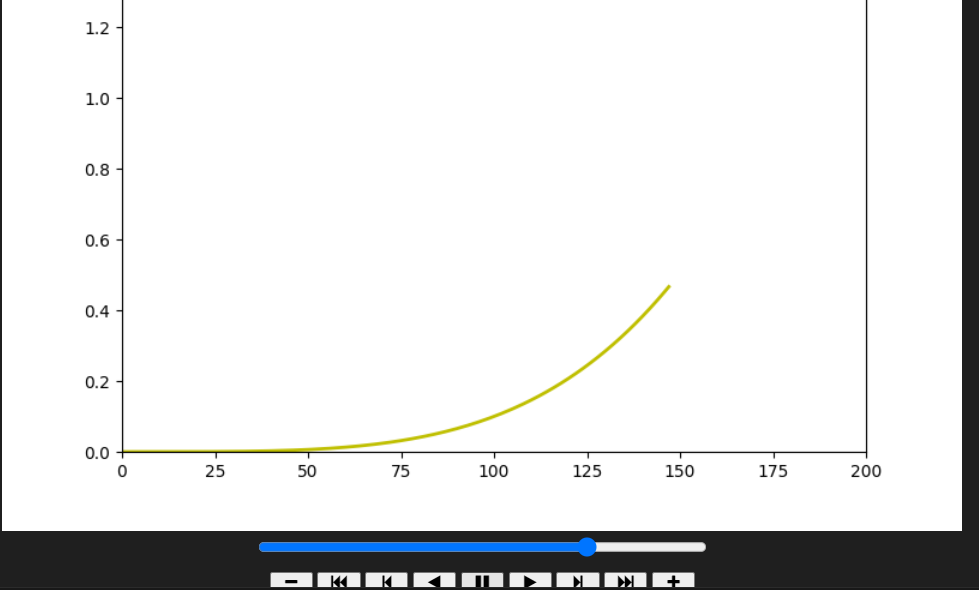
def update(frame):

    line.set\_data(x[:frame], y[:frame])

    return line,

ani = animation.FuncAnimation(fig, update, frames=len(x), interval=10, blit=True)

HTML(ani.to\_jshtml())

****

**Vertical Line animation**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.animation as animation

from IPython.display import HTML

plt.ioff()

fig, ax = plt.subplots()

ax.set\_xlim(0, 10)

ax.set\_ylim(0, 10)

line, = ax.plot([], [], 'r-', linewidth=2)

def init():

    line.set\_data([], [])

    return line,

def update(frame):

    x = [5, 5]

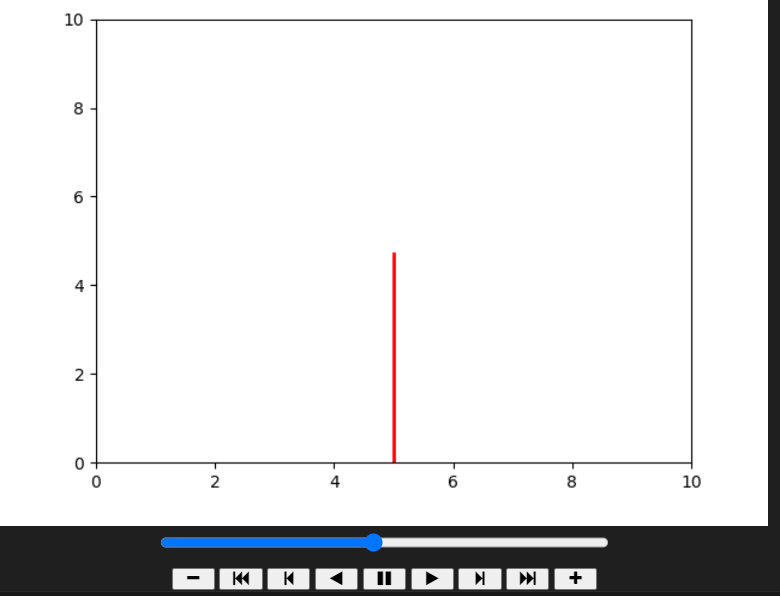
    y = [0, frame]

    line.set\_data(x, y)

    return line,

ani = animation.FuncAnimation(fig, update, frames=np.arange(0, 10, 0.1), init\_func=init, interval=50, blit=True)

HTML(ani.to\_jshtml())

****

**Horizontal Line animation**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.animation as animation

from IPython.display import HTML

plt.ioff()

fig, ax = plt.subplots()

ax.set\_xlim(0, 10)

ax.set\_ylim(0, 10)

line, = ax.plot([], [], 'r-', linewidth=2)

def init():

    line.set\_data([], [])

    return line,

def update(frame):

    x = [0, frame]

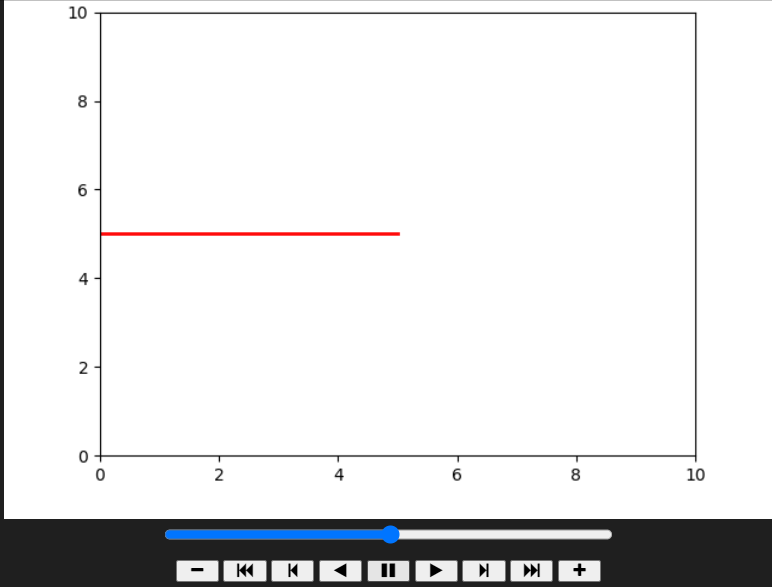
    y = [5, 5]

    line.set\_data(x, y)

    return line,

ani = animation.FuncAnimation(fig, update, frames=np.arange(0, 10, 0.1), init\_func=init, interval=50, blit=True)

HTML(ani.to\_jshtml())

****

**Pendulum**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.animation as ani

from IPython.display import HTML

import math

plt.ioff()

fig=plt.figure(figsize=(10,5))

ax=fig.add\_subplot(111)

t1=np.linspace(math.pi\*(7/6),math.pi\*(11/6),200)

t2=np.linspace(math.pi\*(11/6),math.pi\*(7/6),200)

t=np.concatenate((t1,t2))

t=np.delete(t,200)

r=5

ax.set\_xlim(-5.5,5.5)

ax.set\_ylim(-5.5,0.5)

ax.plot([0],[0],'go',markersize=10)

line,=ax.plot([],[],'g-',lw=2)

point,=ax.plot([],[],'go',markersize=10)

def init():

    line.set\_data([],[])

    return line,

def fun(fr):

    y=r\*math.sin(t[fr])

    x=r\*math.cos(t[fr])

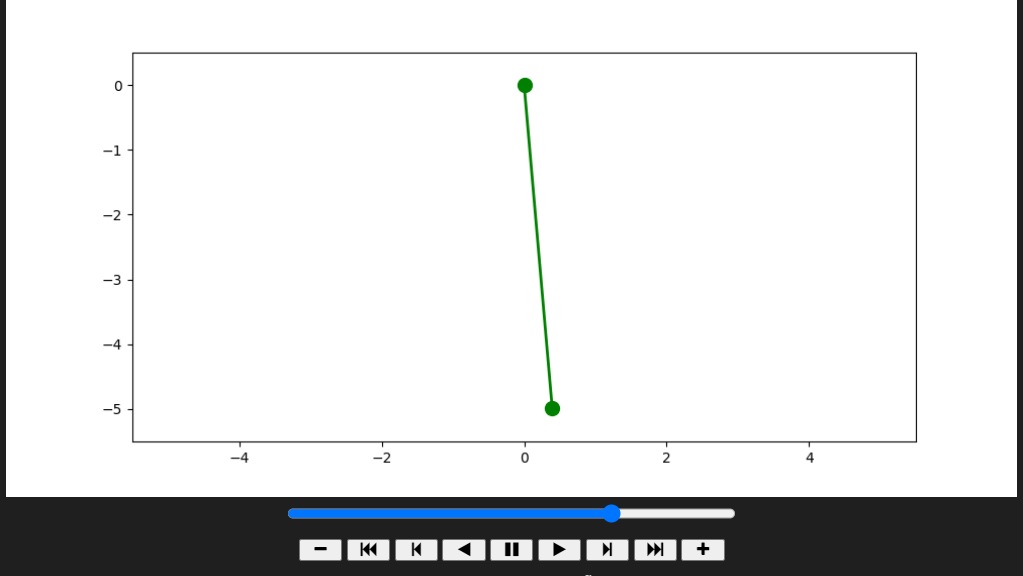
    line.set\_data([0,x],[0,y])

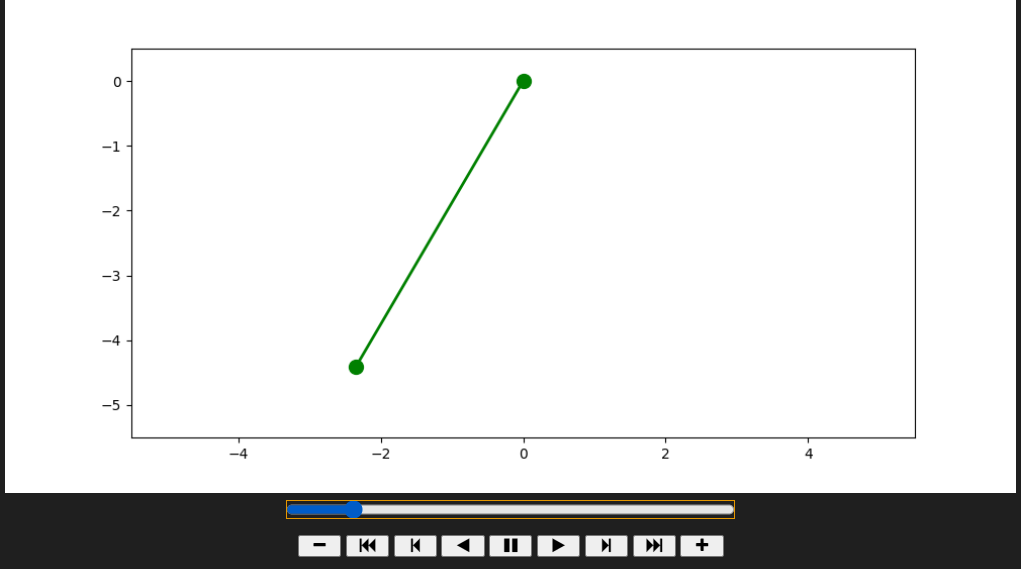
    point.set\_data([x],[y])

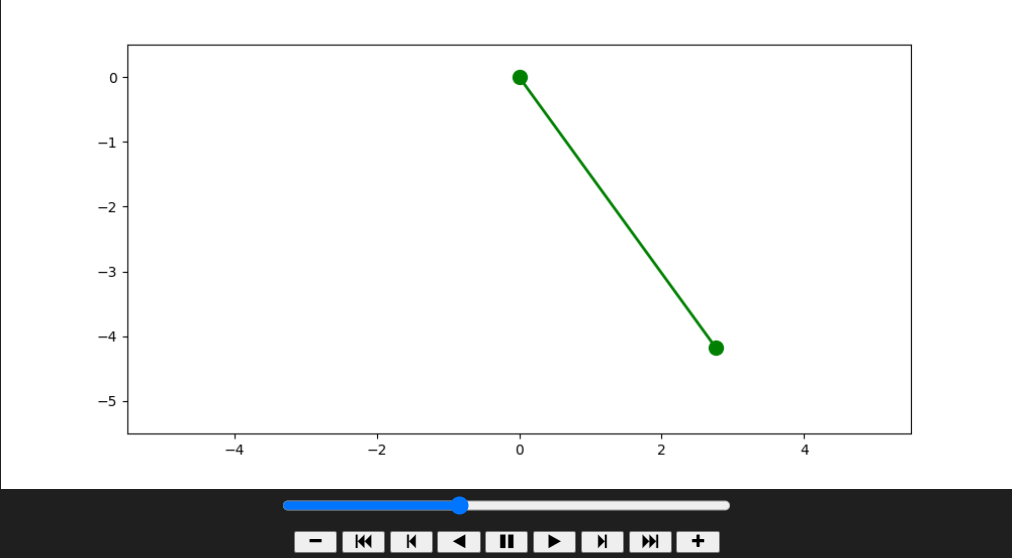
    return line, point,

a=ani.FuncAnimation(fig,fun,frames=np.arange(0,len(t)),init\_func=init,interval=5)

HTML(a.to\_jshtml())

****

****

****

**S Design animation Horizontal**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.animation as animation

from IPython.display import HTML

plt.ioff()

fig, ax = plt.subplots()

ax.set\_xlim(-1, 1)

ax.set\_ylim(0, 10)

line, = ax.plot([], [], 'r-', linewidth=2)

def init():

    line.set\_data([], [])

    return line,

def update(frame):

    t = np.linspace(0, frame, 100)

    x = np.sin(t)

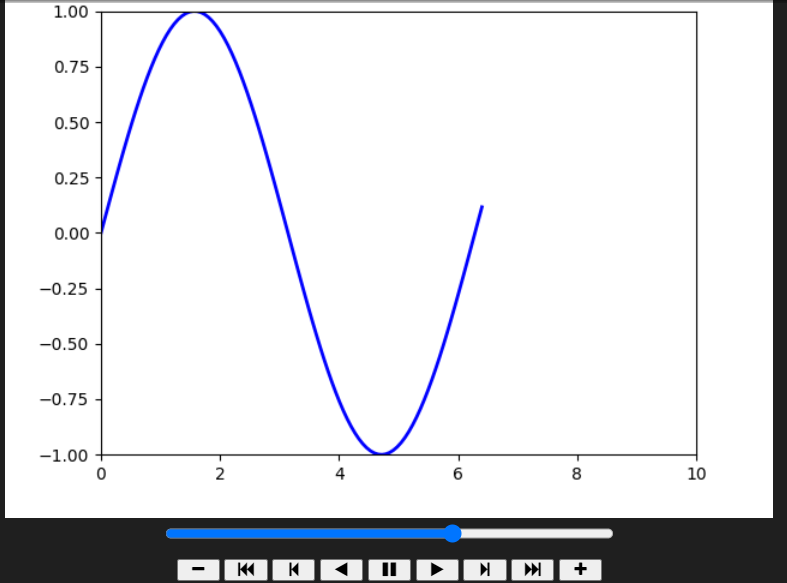
    y = t

    line.set\_data(x, y)

    return line,

ani = animation.FuncAnimation(fig, update, frames=np.arange(0, 10, 0.1), init\_func=init, interval=50, blit=True)

HTML(ani.to\_jshtml())

****

**Bar Graph**

x = ['a', 'b', 'c','d']

y = [20, 50, 30, 60]

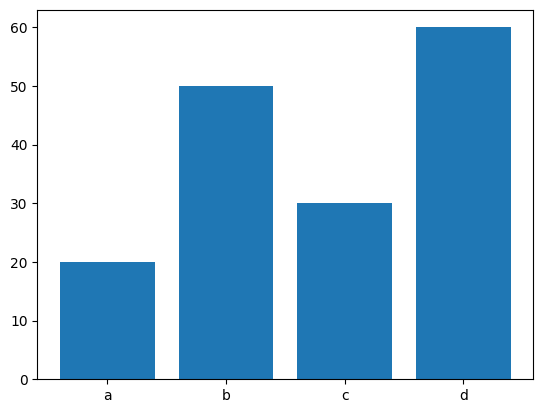
plt.ioff()

fig = plt.figure()

ax = fig.add\_subplot(111)

ax.bar(x, y)

plt.show()

****

**Animation Bar Graph**

x = ['a', 'b', 'c','d']

y = [20, 50, 30, 60]

fig = plt.figure()

ax = fig.add\_subplot(111)

ax.clear()

ax.set\_xlim(0, 65)

bar=ax.barh(x,[0] \* len(y), color='b')

print(bar)

def fun(frame):

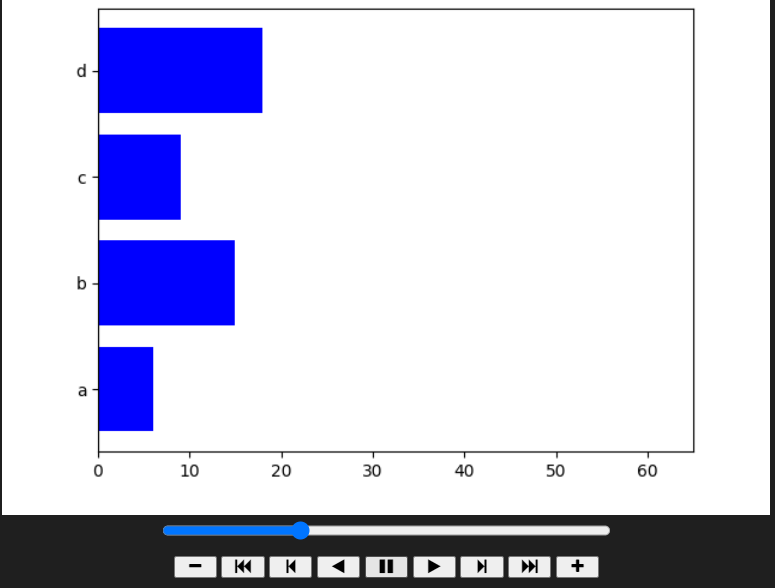
    for b, wd in zip(bar, y):

        b.set\_width(frame\*wd/100)

    return bar

a = animation.FuncAnimation(fig,fun,frames=np.arange(0,101,1),interval=50)

HTML(a.to\_jshtml())

****

**U Shaped Animation**

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.animation as animation

from IPython.display import HTML

fig=plt.figure(figsize=(10,5))

ax=fig.add\_subplot(111)

x=np.linspace(-10,10,200)

y=np.square(x)

ax.set\_xlim(-10,10)

ax.set\_ylim(min(y)-1,max(y)+1)

line,=ax.plot([],[],'g',linewidth=4)

def init():

    line.set\_data([],[])

    return line,

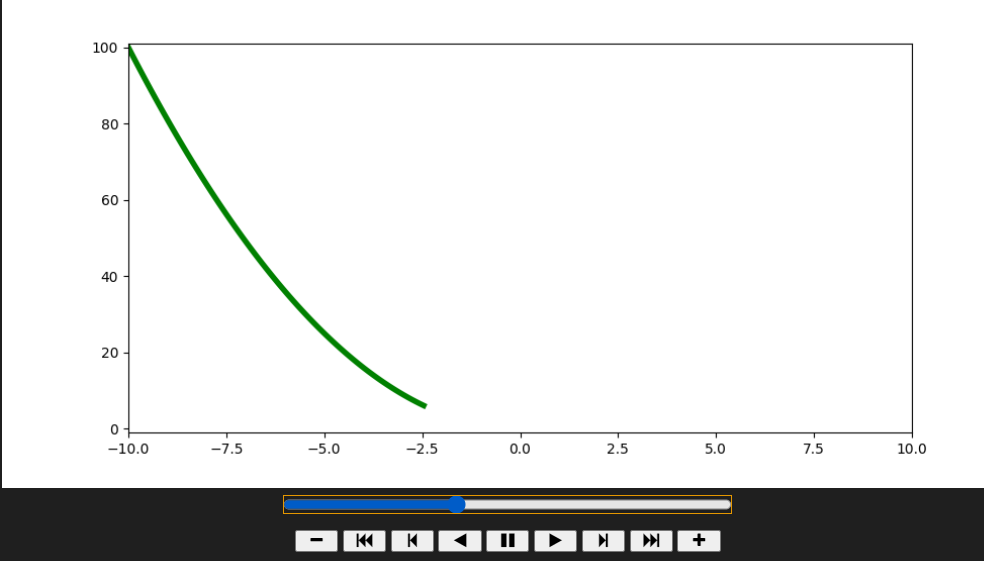
def fun(fr):

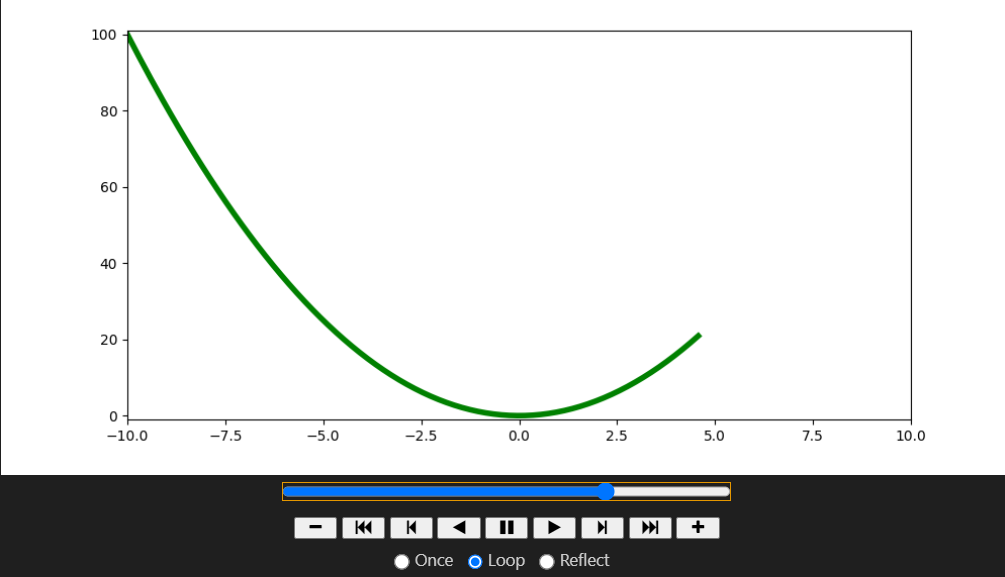
    line.set\_data(x[:fr],y[:fr])

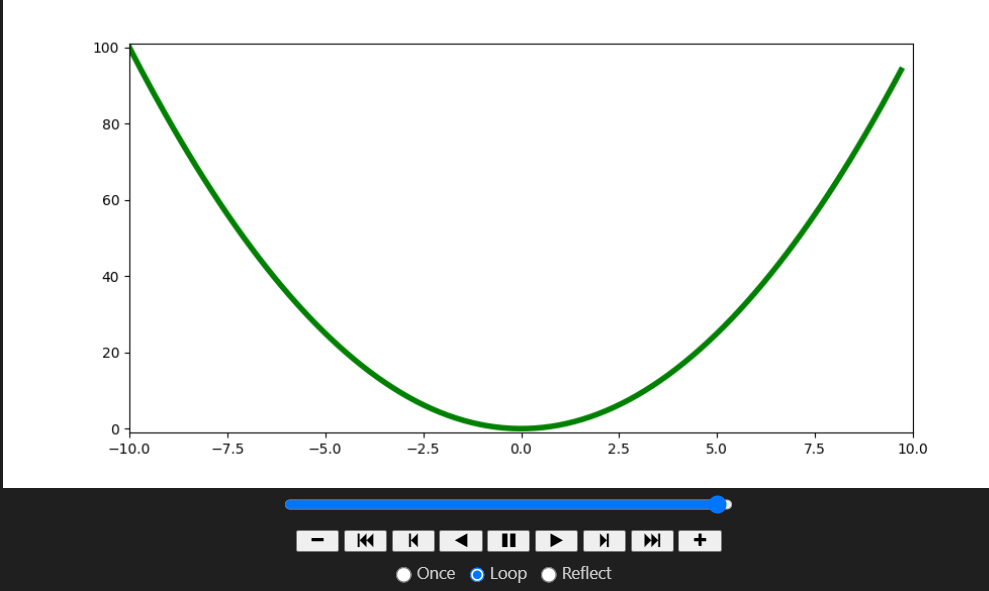
    return line,

a= animation.FuncAnimation(fig,fun,frames=np.arange(0,201),init\_func=init,interval=10)

HTML(a.to\_jshtml())

****

****

****