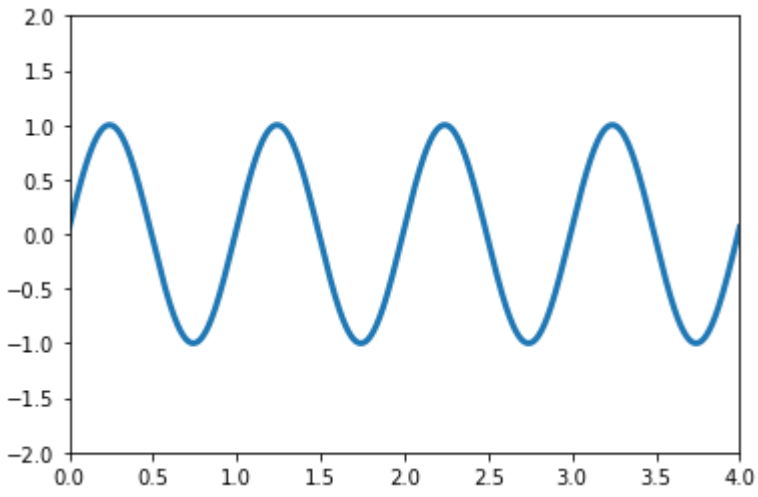
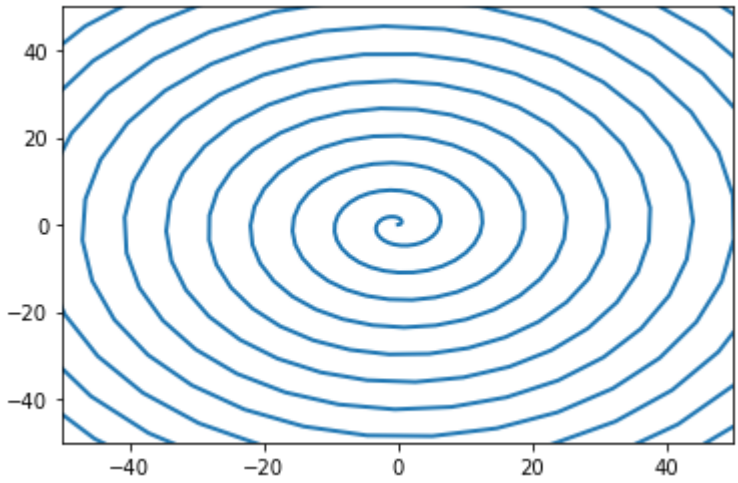
In [1]:

```
# Animations with Matplotlib

%matplotlib qt
%matplotlib inline
import numpy as np
from matplotlib import pyplot as plt
from matplotlib.animation import FuncAnimation
# Let's define the objects, in other words, the figure, axes,
# and plot, as follows:
fig = plt.figure()
ax = plt.axes(xlim=(0, 4), ylim=(-2, 2))
line, = ax.plot([], [], lw=3)
# Let's define the function init(), which will initialize the animation
# and set the data for the animation, as shown here:
def init():
    line.set_data([], [])
    return line,
# Let's define an animation function, as shown here:
def animate(i):
    x = np.linspace(0, 4, 1000)
    y = np.sin(2 * np.pi * (x - 0.01 * i))
    line.set_data(x, y)
    return line,
# This function accepts the frame number as an argument
# (in this case the variable named i) and renders the frame for animation.
# Now that we have defined the components, let's create an animation
# object using the function call FuncAnimation(). It accepts the created
# functions as arguments. It also accepts the number of frames and the
# interval as arguments. The argument for the parameter blit is True.
# This means that only the parts of the plot that have changed are
#redrawn.
anim = FuncAnimation(fig, animate,init_func=init,
                     frames=1000,
                     interval=10,
                     blit=True)
# You can also save the animation as a GIF as follows:
anim.save('sine_wave.gif', writer='pillow')
```



In [3]:

```
# You can interact with animation and change the orientation with your
# mouse. Explore all the interactive possibilities before proceeding
# further. You can create a progressive spiral, as shown here:

%matplotlib qt
%matplotlib inline
import numpy as np
from matplotlib import pyplot as plt
from matplotlib.animation import FuncAnimation
fig = plt.figure()
ax = plt.axes(xlim=(-50, 50), ylim=(-50, 50))
line, = ax.plot([], [], lw=2)
def init():
    line.set_data([], [])
    return line,
xdata, ydata = [], []
def animate(i):
    t = 0.2*i
    x = t*np.cos(t)
    y = t*np.sin(t)
    xdata.append(x)
    ydata.append(y)
    line.set_data(xdata, ydata)
    return line,
anim = FuncAnimation(fig, animate,init_func=init,
                     frames=3000,
                     interval=5,
                     blit=True)
anim.save('spiral.gif', writer='pillow')
```



In [4]:

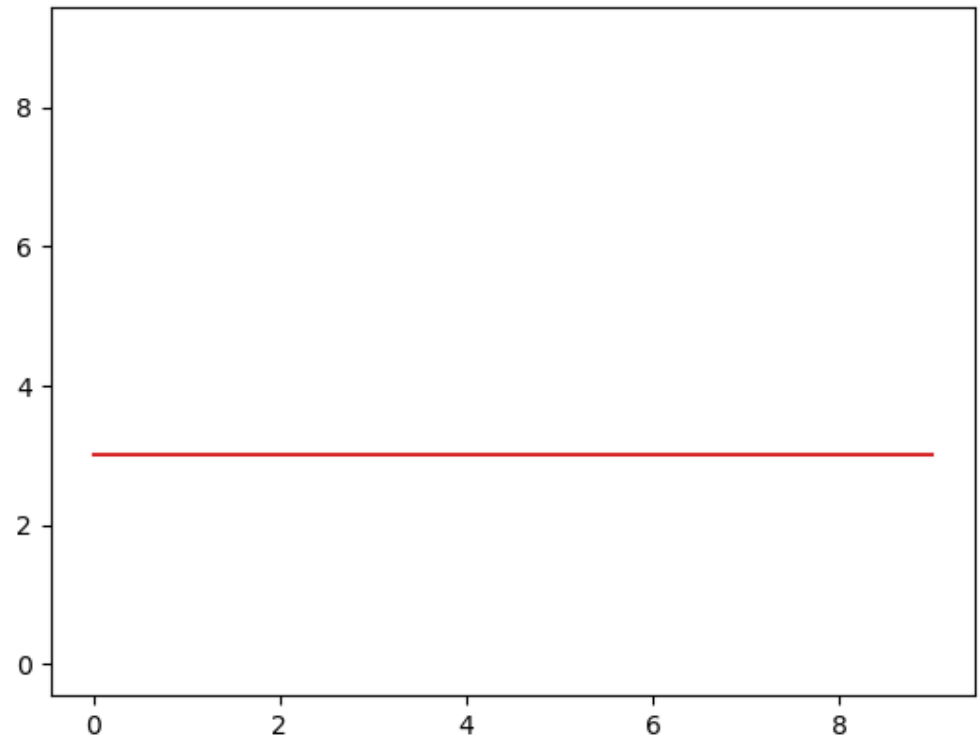
```
# Celluloid Library
# You can use another simple library called Celluloid for animation.
# Let's install it as follows:

!pip3 install celluloid
```

```
Collecting celluloid
  Downloading celluloid-0.2.0-py3-none-any.whl (5.4 kB)
Requirement already satisfied: matplotlib in c:\python\lib\site-packages
(from celluloid) (3.2.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
c:\python\lib\site-packages (from matplotlib->celluloid) (2.4.7)
Requirement already satisfied: cycler>=0.10 in c:\python\lib\site-packages
(from matplotlib->celluloid) (0.10.0)
Requirement already satisfied: python-dateutil>=2.1 in c:\python\lib\site-
packages (from matplotlib->celluloid) (2.8.1)
Requirement already satisfied: numpy>=1.11 in c:\python\lib\site-packages
(from matplotlib->celluloid) (1.19.4)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\python\lib\site-pac
kages (from matplotlib->celluloid) (1.2.0)
Requirement already satisfied: six in c:\python\lib\site-packages (from cy
cler>=0.10->matplotlib->celluloid) (1.12.0)
Installing collected packages: celluloid
Successfully installed celluloid-0.2.0
```

In [2]:

```
# You can import it as follows:

%matplotlib qt
import numpy as np
from matplotlib import pyplot as plt
from matplotlib.animation import FuncAnimation
import celluloid
from celluloid import Camera
from IPython.display import HTML
# Let's create a figure and camera object as follows:
fig = plt.figure()
camera = Camera(fig)
# Let's create the frames of an animation and save them in memory
# with the function called camera.snap(), as follows:
for i in range(10):
    plt.plot([i] * 10)
    camera.snap()
# Let's create the animation as follows:
animation = camera.animate()
HTML(animation.to_jshtml())
# animation.save('line.gif', writer='pillow')
```

Out[2]:

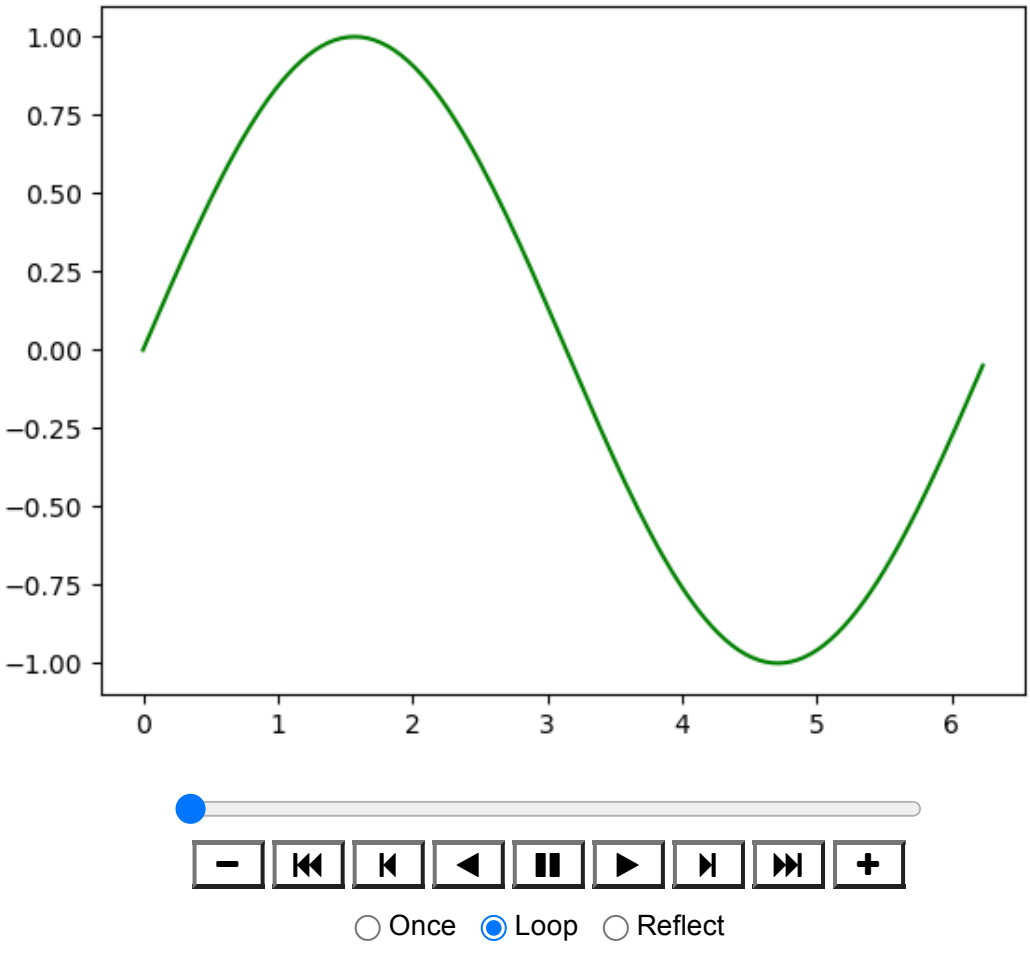


Once Loop Reflect

In [3]:

```
# You can also create a sine wave as follows:

%matplotlib qt
import numpy as np
from matplotlib import pyplot as plt
from matplotlib.animation import FuncAnimation
import celluloid
from celluloid import Camera
from IPython.display import HTML
fig, axes = plt.subplots()
camera = Camera(fig)
t = np.linspace(0, 2 * np.pi, 128, endpoint=False)
for i in t:
    plt.plot(t, np.sin(t + i), color='green')
    camera.snap()
animation = camera.animate()
HTML(animation.to_jshtml())
# animation.save('sine wave.gif', writer='pillow')
```

Out[3]:



In [16]:

```
# Another example with a bar graph is as follows:

%matplotlib qt
import numpy as np
from matplotlib import pyplot as plt
from matplotlib.animation import FuncAnimation
import celluloid
from celluloid import Camera
from IPython.display import HTML
fig, axes = plt.subplots()
camera = Camera(fig)
y = np.arange(5)
for i in y:
    plt.bar( np.random.rand(5)*10 , y)
    camera.snap()
animation = camera.animate()
HTML(animation.to_jshtml())
# animation.save('Bar Graph.gif', writer='pillow')
```

Out[16]: