In [5]:
```
import numpy as np
np.linspace(2.0, 3.0, num=5)
```

Out[5]:
```
array([2.  , 2.25, 2.5 , 2.75, 3.  ])
```

In [6]:
```
import numpy as np
np.linspace(2.0, 3.0, num=5, retstep=True)
```

Out[6]:
```
(array([2.  , 2.25, 2.5 , 2.75, 3.  ]), 0.25)
```

In [8]:
```
import numpy as np
np.linspace(2.0, 3.0, num=5, retstep=True, endpoint = False)
```

Out[8]:
```
(array([2. , 2.2, 2.4, 2.6, 2.8]), 0.2)
```

In [11]:
```
import numpy as np
np.linspace(2.0, 3.0, num=2, retstep=True, endpoint = False)
```

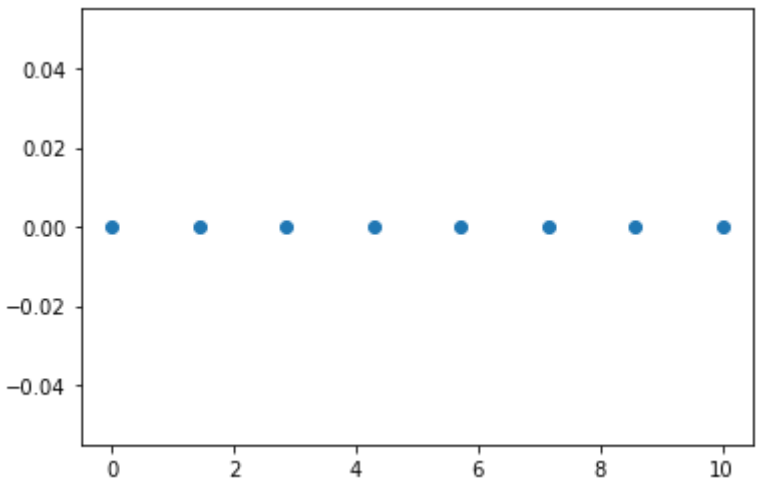
Out[11]:
```
(array([2. , 2.5]), 0.5)
```

In [13]:
```
N = 8
y = np.zeros(N)
print(y)
```

```
[0. 0. 0. 0. 0. 0. 0. 0.]
```

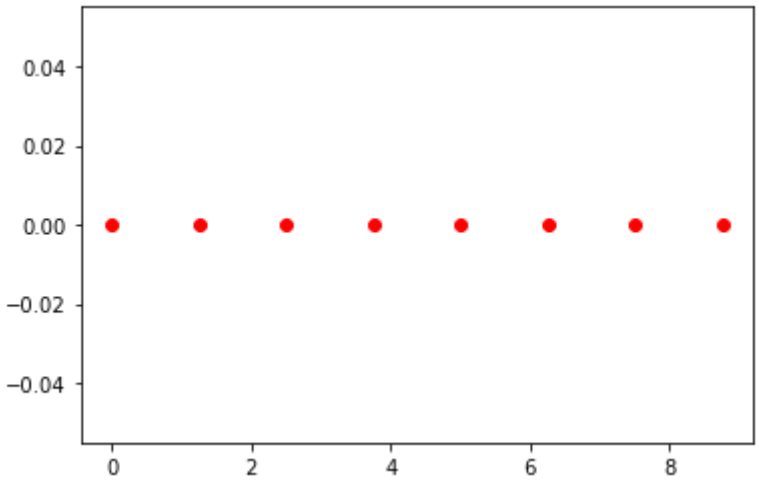
In [14]:
```
import matplotlib.pyplot as plt
N = 8
y = np.zeros(N)
x1 = np.linspace(0, 10, N, endpoint=True)
x2 = np.linspace(0, 10, N, endpoint=False)
plt.plot(x1, y, 'o')
```

Out[14]:
```
[<matplotlib.lines.Line2D at 0x24401419580>]
```



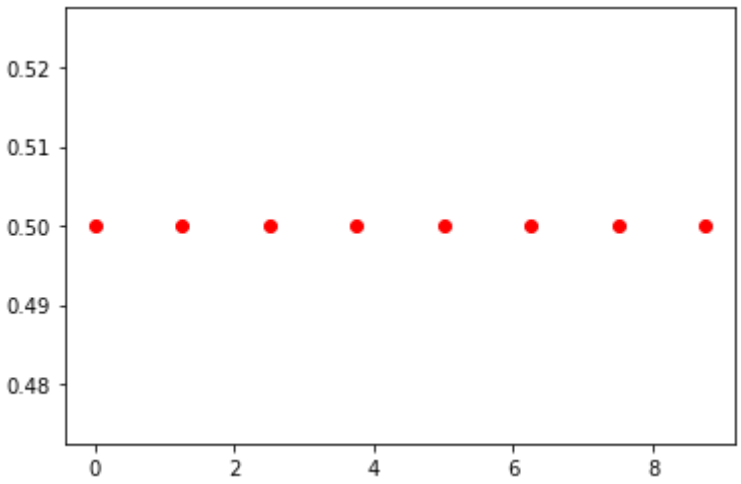

In [17]:
```
import matplotlib.pyplot as plt
N = 8
y = np.zeros(N)
x1 = np.linspace(0, 10, N, endpoint=True)
x2 = np.linspace(0, 10, N, endpoint=False)
plt.plot(x2, y, 'or')  #o = circles; r = redcolor
```

Out[17]:
```
[<matplotlib.lines.Line2D at 0x244017f9160>]
```



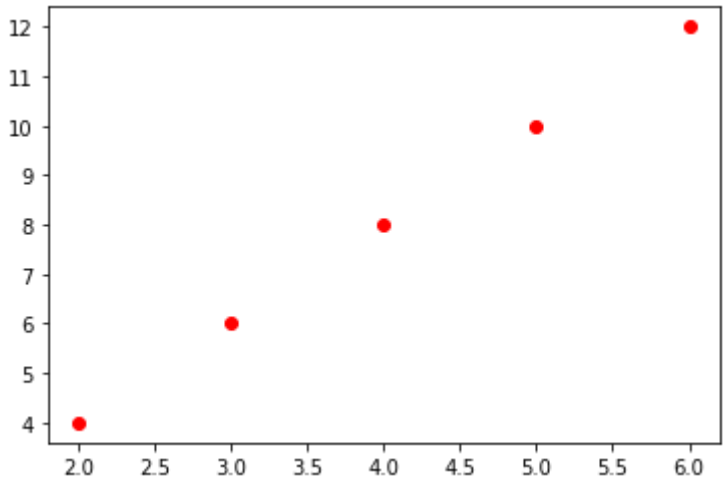

In [18]:
```
import matplotlib.pyplot as plt
N = 8
y = np.zeros(N)
x1 = np.linspace(0, 10, N, endpoint=True)
x2 = np.linspace(0, 10, N, endpoint=False)
plt.plot(x2, y + 0.5, 'or')  #o = circles; r = redcolor
```

Out[18]:
```
[<matplotlib.lines.Line2D at 0x2440184b040>]
```

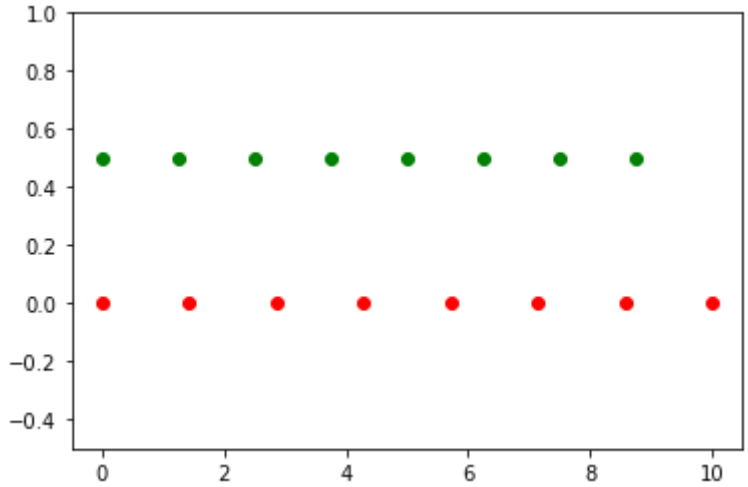



In [19]:
```
import matplotlib.pyplot as plt

l1 = [2, 3, 4, 5, 6]

l2 = [4, 6, 8, 10, 12]

plt.plot(l1, l2, 'or')
```

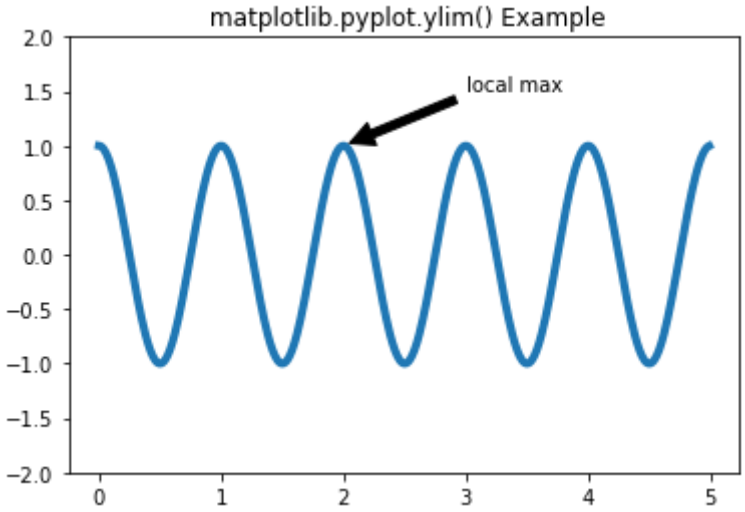
Out[19]:
```
[<matplotlib.lines.Line2D at 0x24401895880>]
```



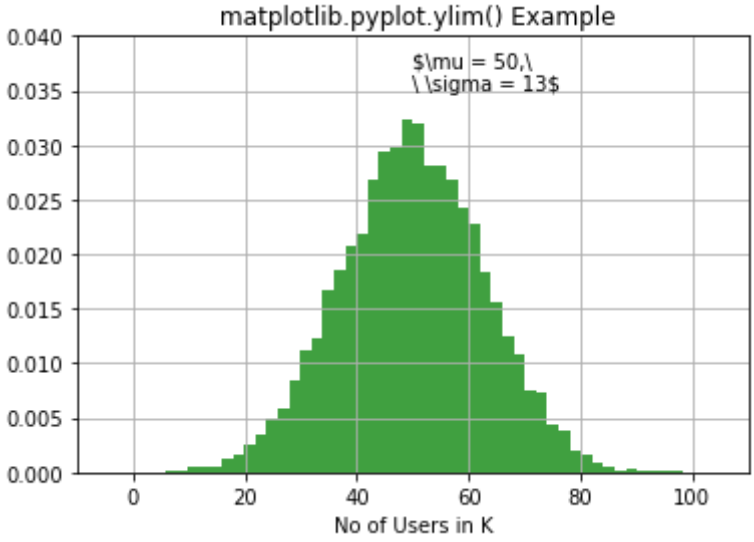

In [22]:
```
import matplotlib.pyplot as plt
N = 8
y = np.zeros(N)
x1 = np.linspace(0, 10, N, endpoint=True)
x2 = np.linspace(0, 10, N, endpoint=False)
plt.plot(x1, y, 'or')
plt.plot(x2, y + 0.5, 'og')
plt.ylim([-0.5, 1])
plt.show()
```




In [24]:
```
# Implementation of matplotlib function
import matplotlib.pyplot as plt
import numpy as np

ax = plt.subplot(111)

t = np.arange(0.0, 5.0, 0.01)
s = np.cos(2 * np.pi * t)
line, = plt.plot(t, s, lw = 4)  #lw: Line Width

plt.annotate('local max', xy =(2, 1),
             xytext =(3, 1.5),
             arrowprops = dict(facecolor ='black',
                               shrink = 0.05), )

plt.ylim(-2, 2)
plt.title(" matplotlib.pyplot.ylim() Example")
plt.show()
```

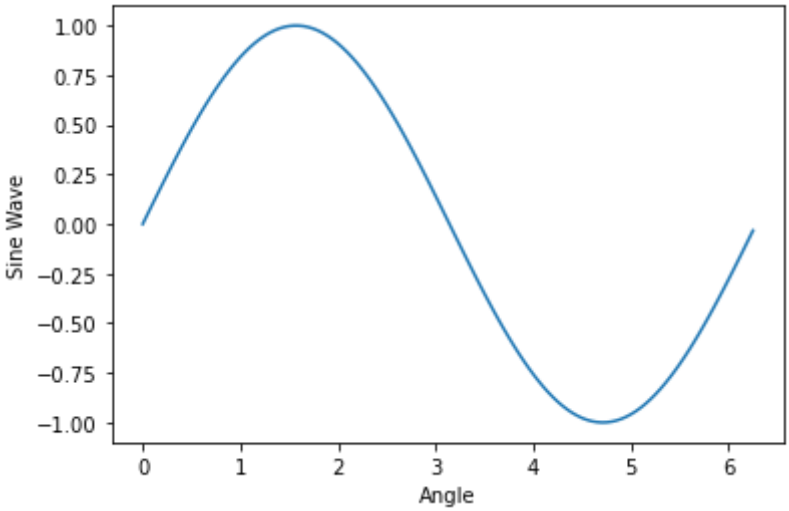



In [25]:
```
# Implementation of matplotlib function
import matplotlib.pyplot as plt
import numpy as np

np.random.seed(9680801)

mu, sigma = 50, 13
x = mu + sigma * np.random.randn(10000)

# the histogram of the data
n, bins, patches = plt.hist(x, 50, density = True, facecolor ='g', alpha = 0.75)


plt.xlabel('No of Users in K')
plt.title('Histogram of IQ')
plt.text(50, .035, r'$\mu = 50,\
\ \sigma = 13$')

plt.xlim(-10, 110)
plt.ylim(0, 0.04)

plt.grid(True)
plt.title(" matplotlib.pyplot.ylim() Example")

plt.show()
```



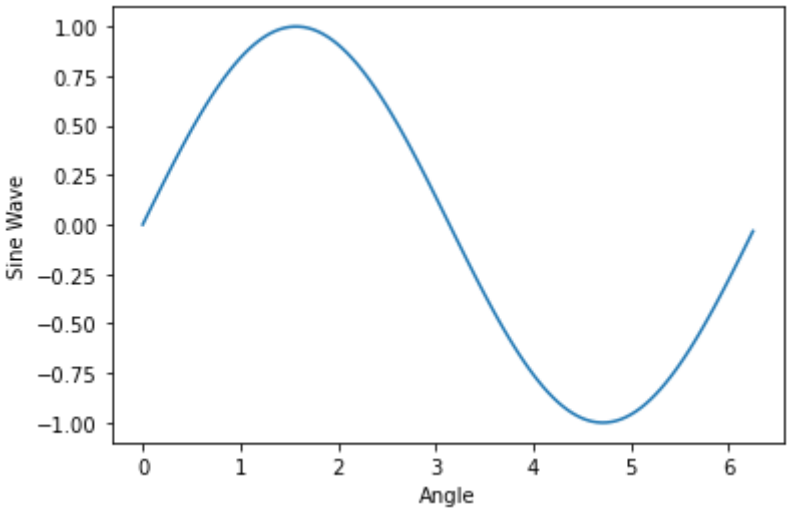

In [29]:
```
import matplotlib.pyplot as plt

import numpy as np
```

```
import math

x = np.arange(0, math.pi*2, 0.05)

y = np.sin(x)

plt.plot(x, y)

plt.xlabel("Angle")

plt.ylabel("Sine Wave")

plt.show()
```




In [30]:

```
import matplotlib.pyplot as plt

import numpy as np

import math

x = np.arange(0, math.pi*2, 0.05)

y = np.sin(x)

plt.plot(x, y)

plt.xlabel("Angle")

plt.ylabel("Sine Wave")

plt.show()

%matplotlib inline  #Display plot outputs inside the notebook itself (and not in the
```


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Sine Wave
0
1
2
3
4
5
6
Angle