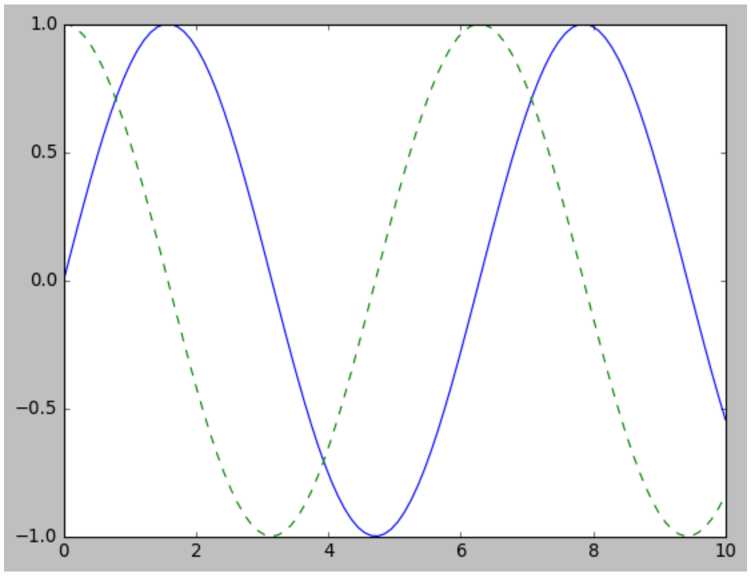
In [74]:
```
import matplotlib as mpd
import matplotlib.pyplot as plt
#importing required libraries
```

In [75]:
```
plt.style.use("classic")#using classic matplotlib
#use plt.show() oly oce as multiple plt.show() can cause unexpected errors
```

In [76]:
```
%matplotlib inline
#static images of plot will be embedded in notebook
#other mode is noteook which interactive plots will embedded in notebook
```

In [77]:
```
#sample program
import numpy as np
x = np.linspace(0, 10, 100)
fig=plt.figure()
plt.plot(x,np.sin(x),'-')
plt.plot(x,np.cos(x),'--');
```



In [78]:
```
#plots can also be saved using savefig method
fig.savefig("graph.png")
```

In [79]:
```
#file types supported in system for saving plots
fig.canvas.get_supported_filetypes()
```

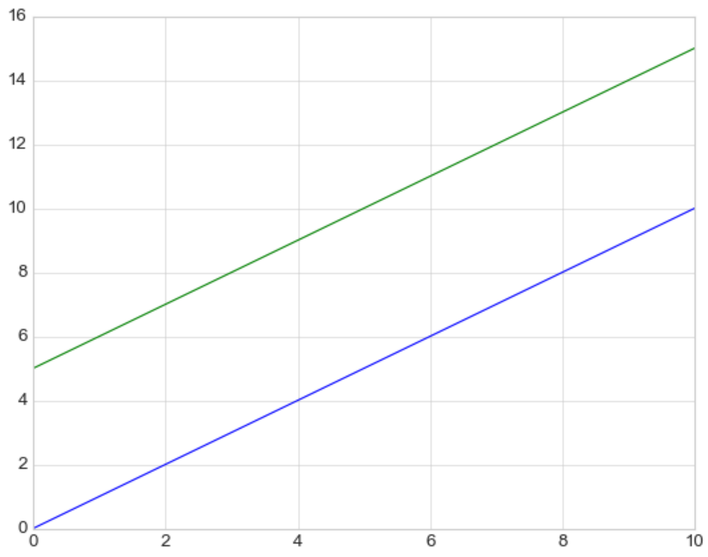
Out[79]:
```
{'eps': 'Encapsulated Postscript',
 'jpg': 'Joint Photographic Experts Group',
 'jpeg': 'Joint Photographic Experts Group',
 'pdf': 'Portable Document Format',
 'pgf': 'PGF code for LaTeX',
 'png': 'Portable Network Graphics',
 'ps': 'Postscript',
 'raw': 'Raw RGBA bitmap',
 'rgba': 'Raw RGBA bitmap',
 'svg': 'Scalable Vector Graphics',
 'svgz': 'Scalable Vector Graphics',
 'tif': 'Tagged Image File Format',
 'tiff': 'Tagged Image File Format'}
```

In [80]:
```
#simple line plots in matplotlib
```

In [81]:
```
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use("seaborn-whitegrid")
import numpy as np
```

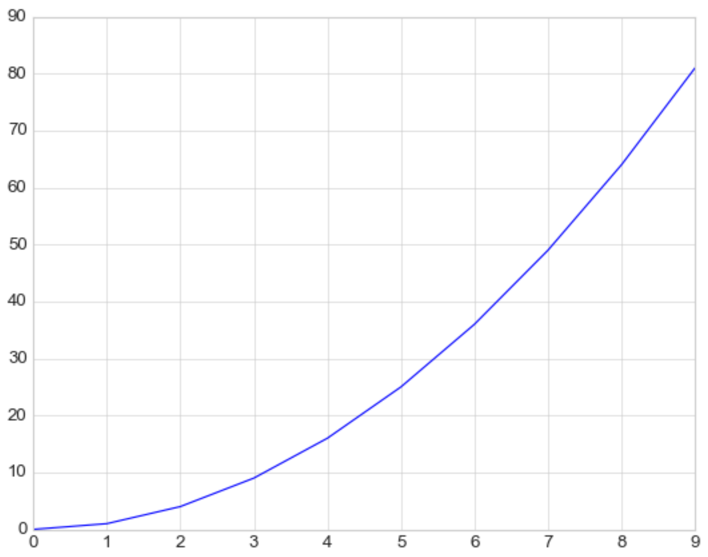
In [82]:
```
fig=plt.figure()
ax=plt.axes()
x = np.linspace(0, 10, 100)
ax.plot(x,x)#y(x)=x
ax.plot(x,x+5)#plotting y(x)=x+5
```

Out[82]:
```
[<matplotlib.lines.Line2D at 0x28811a13a90>]
```



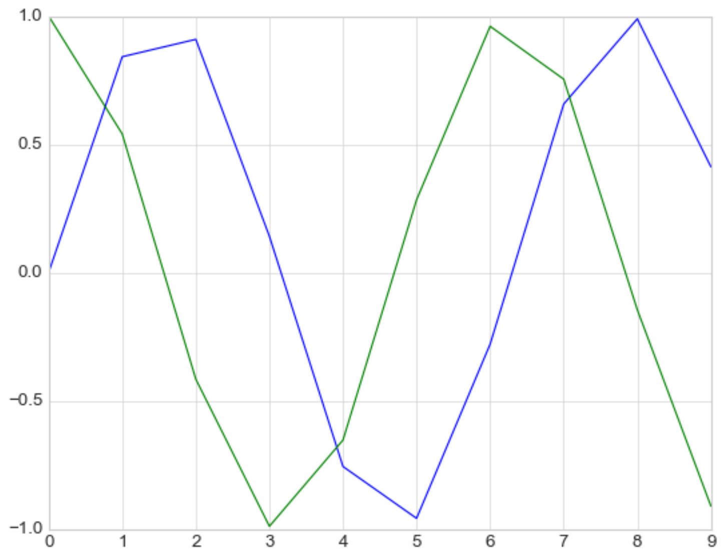
In [83]:
```
#directly creating figure and plot by calling plot from pyplot
x=np.arange(0,10)
plt.plot(x,np.power(x,2))#plottig y(x)=x^2 directly without creatig figure and axes
```

Out[83]:
```
[<matplotlib.lines.Line2D at 0x288141d2580>]
```



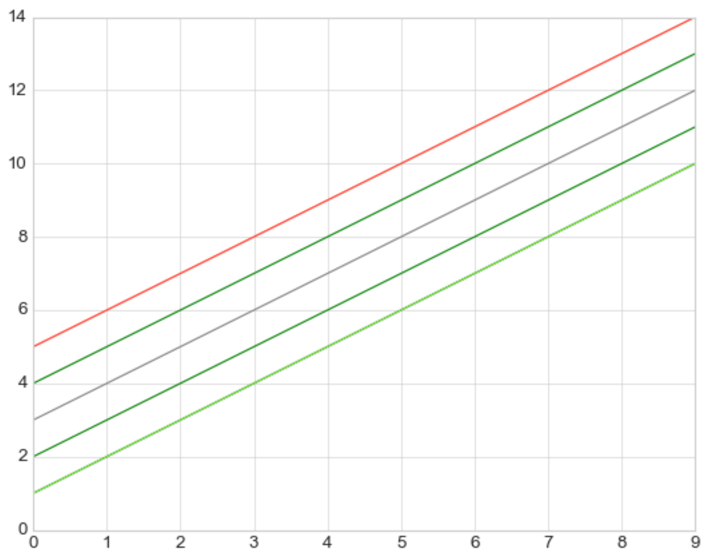
In [84]:
```
plt.plot(x,np.sin(x))
plt.plot(x,np.cos(x))
```

Out[84]:
```
[<matplotlib.lines.Line2D at 0x28814477910>]
```



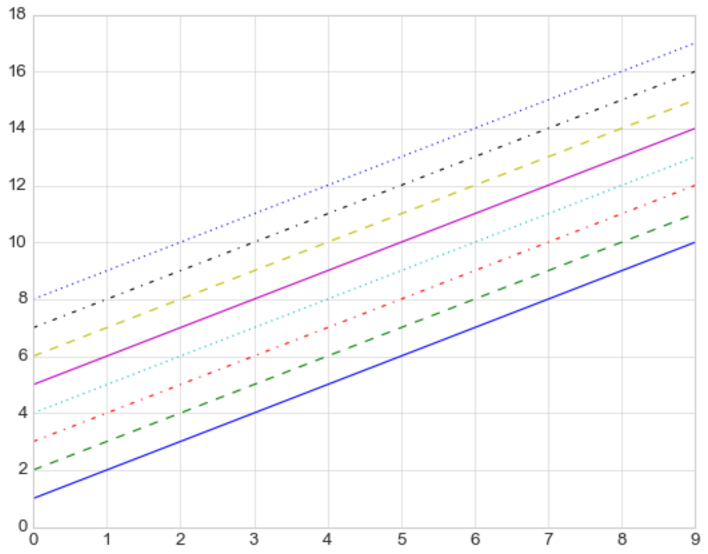
In [85]:
```
#adjusting the plot-line colors and styles
#setting colors
plt.plot(x,x+1,color="blue")#specifying by color name
plt.plot(x,x+2,color="g")#short color code (rgbcmyk)
plt.plot(x,x+3,color="0.5")#grayscale between 0 and 1
plt.plot(x,x+4,color="g")#short color code (rgbcmyk)
plt.plot(x,x+5,color="#FFDD44")#hexadecimal code
plt.plot(x,x+5,color=(1,0.2,0.3))#RGB tuple
plt.plot(x,x+1,color="chartreuse")#html supported format
```

Out[85]:
```
[<matplotlib.lines.Line2D at 0x288144b5520>]
```



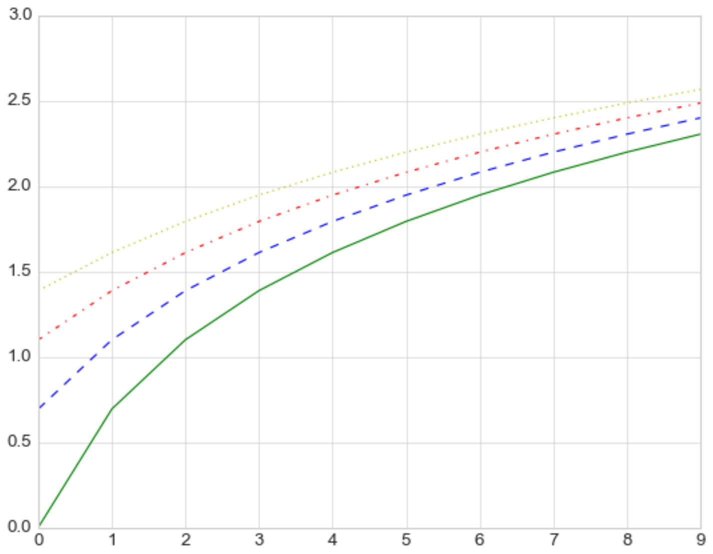
In [86]:
```
#setting linestyle
plt.plot(x,x+1,linestyle="solid")
plt.plot(x,x+2,linestyle="dashed")
plt.plot(x,x+3,linestyle="dashdot")
plt.plot(x,x+4,linestyle="dotted")
plt.plot(x,x+5,linestyle="-")
plt.plot(x,x+6,linestyle="--")
plt.plot(x,x+7,linestyle="-.")
plt.plot(x,x+8,linestyle=":")
```

Out[86]:
```
[<matplotlib.lines.Line2D at 0x28814524940>]
```



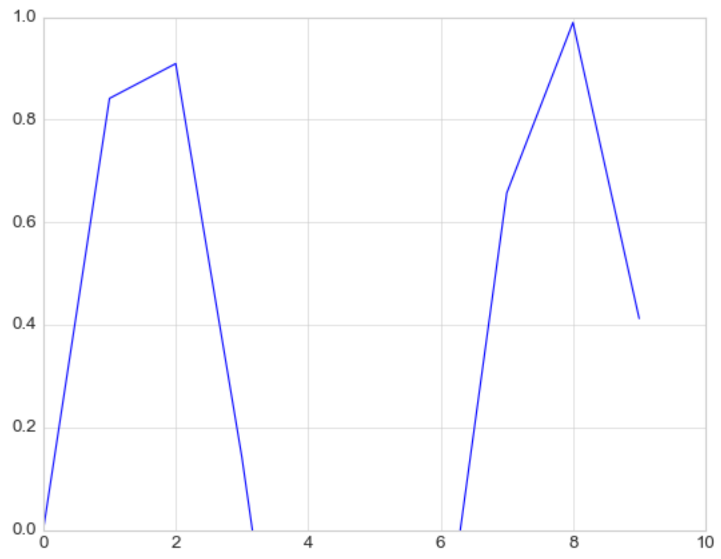
In [87]:
```
#combining color and line style as one argument
plt.plot(x,np.log(x+1),"-g")#solid green
plt.plot(x,np.log(x+2),"--b")#dashed blue
plt.plot(x,np.log(x+3),"-.r")#dot dashed
plt.plot(x,np.log(x+4),":y")#dotted red
```

Out[87]:
```
[<matplotlib.lines.Line2D at 0x288145bb190>]
```



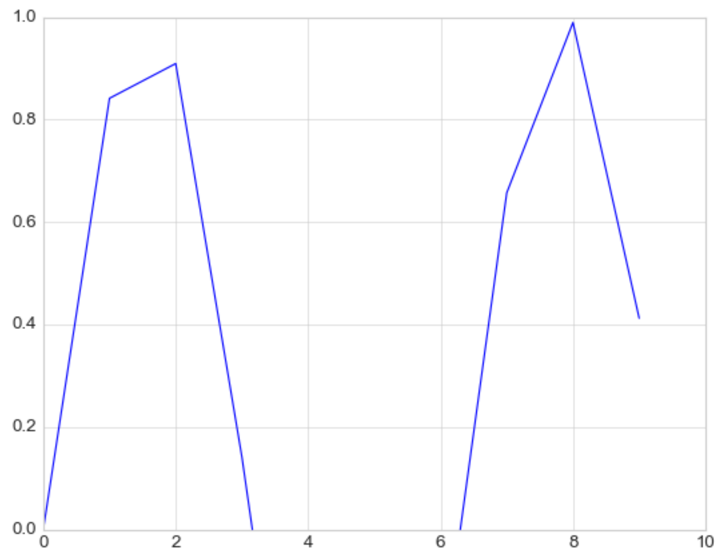
In [88]:
```
#axis limits
plt.plot(x,np.sin(x))
plt.xlim(0,10)
plt.ylim(0,1)
```

Out[88]:
```
(0.0, 1.0)
```



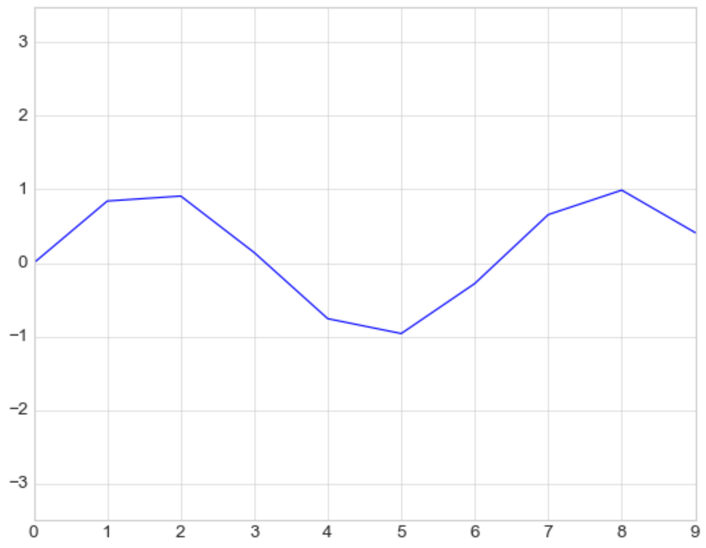
In [89]:
```
#using axis function which takes list of form [xmin,xmax,ymin,ymax]
plt.plot(x,np.sin(x))
plt.axis([0,10,0,1])
```

Out[89]:
```
(0.0, 10.0, 0.0, 1.0)
```



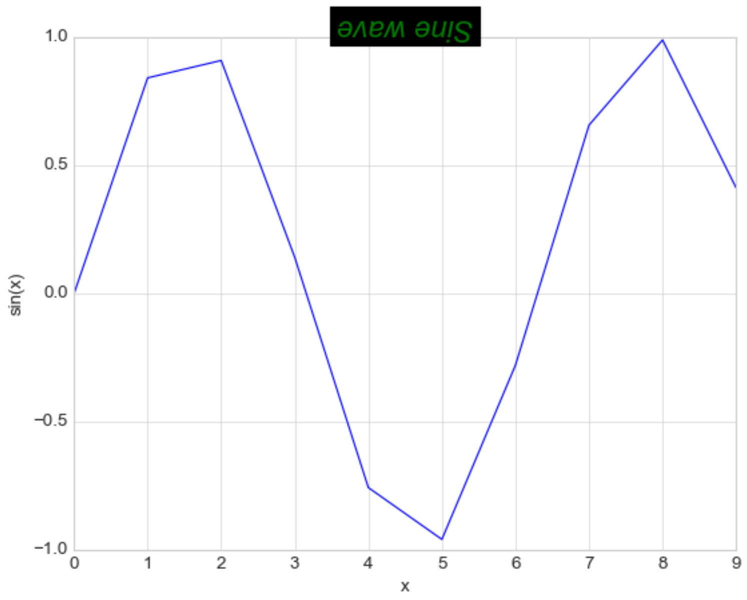
In [90]:
```
#tightening bounds for plot
plt.plot(x,np.sin(x))
plt.axis("tight")#tightening plot
plt.axis("equal")#equal x to y aspect ratio on screen one uit of x is equal to one unit of y
```

Out[90]:
```
(0.0, 9.0, -1.0, 1.0)
```



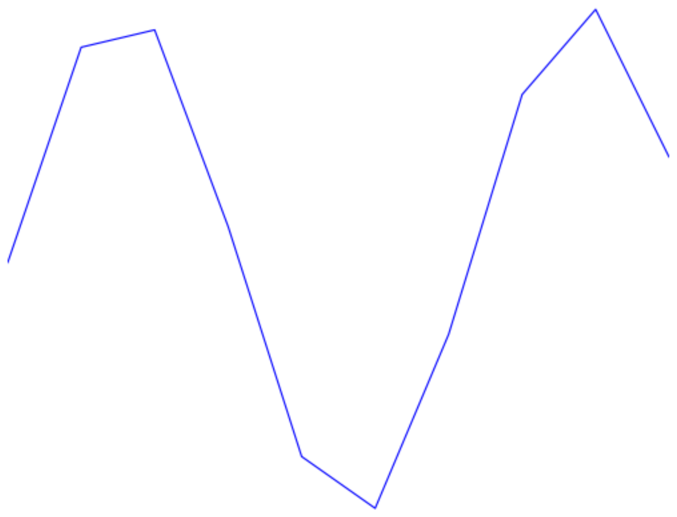
In [91]:
```
#labelling plot
plt.plot(x,np.sin(x))
plt.title(label="Sine wave",fontsize=20,color="green",pad=2,fontstyle="italic",rotation=180,backgroundcolor="black")
plt.xlabel("x")#giving x-axis label as "x"
plt.ylabel("sin(x)")#giving y-axis label as "sin(x)"
```

Out[91]:
```
Text(0, 0.5, 'sin(x)')
```



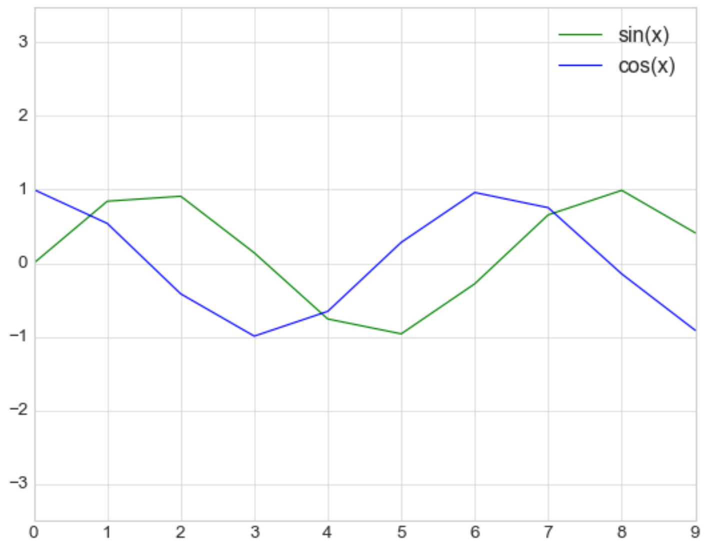
In [92]:
```
plt.plot(x,np.sin(x))
plt.axis("off")#turning off axes and labels "on" will rig ack visibility
```

Out[92]:
```
(0.0, 9.0, -1.0, 1.0)
```



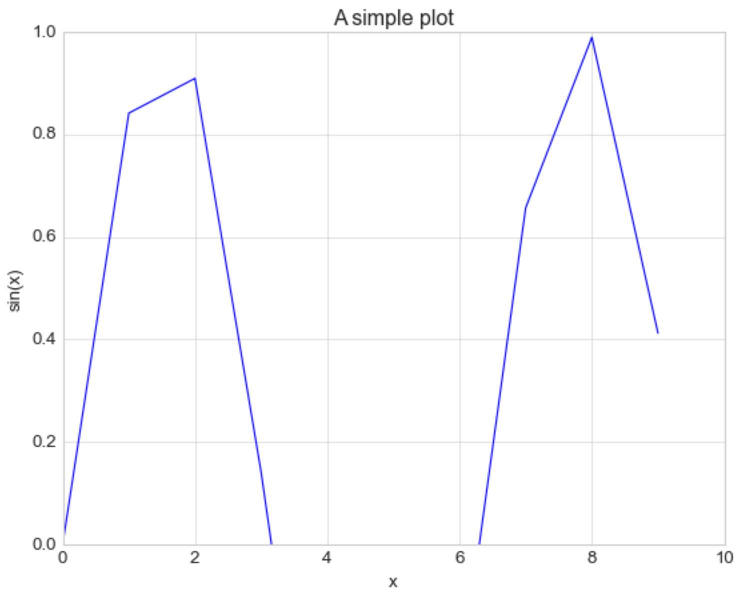
In [93]:
```
#plt.legend() method in matplotlib
plt.plot(x,np.sin(x),"-g",label="sin(x)")#sinx with green
plt.plot(x,np.cos(x),"-b",label="cos(x)")#cosx with blue
plt.axis("equal")
plt.legend()#legend function puts a table that specifies which line corresponds to which function
```

Out[93]:
```
<matplotlib.legend.Legend at 0x28814749940>
```



In [94]:
```
#transitioning from maltlab to object oriented
ax=plt.axes()
ax.plot(x,np.sin(x))
ax.set(xlim=(0,10),ylim=(0,1),xlabel="x",ylabel="sin(x)",title="A simple plot")
#set function in object oriented approach is used to set multiple properties at once
```

Out[94]:
```
[(0.0, 10.0),
 (0.0, 1.0),
 Text(0.5, 0, 'x'),
 Text(0, 0.5, 'sin(x)'),
 Text(0.5, 1.0, 'A simple plot')]
```


A simple plot
sin(x)
x