Introduction of making graph by Python
By T. Ishikawa rev. 2023/5/11

Importing libraries

In [1]:
```
import numpy as np # Library for matrix calculation
import pandas as pd # Library for dataframe
import matplotlib.pyplot as plt # Library of graphing. Matplot-like plot library
import os # Libraly to use the function of OS universaly. You can use the function (
%matplotlib inline
# The last line is needed to make graph in line (inside the notebook).
# plt.style.use('mystyle') # Please use this line after you understand the styleshee
```

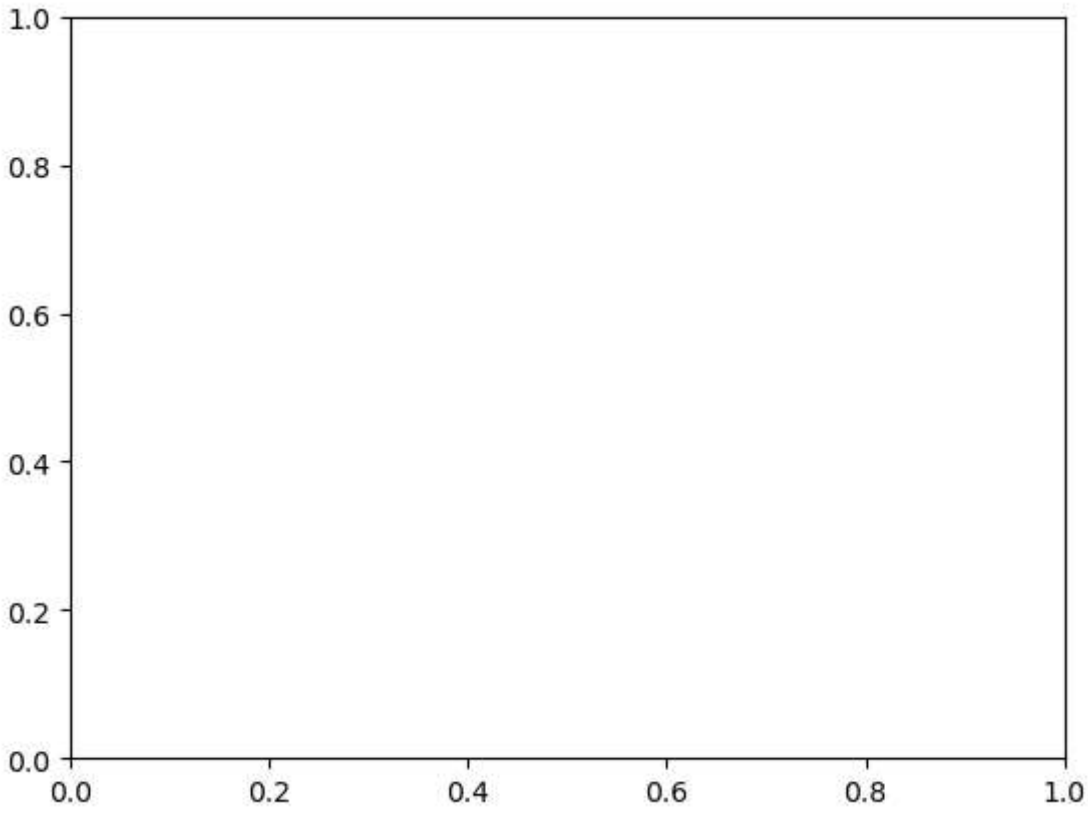
How to make canvas

In [2]:
```
fig=plt.figure() # Preparing the canvas named as 'fig'
```

```
<Figure size 640x480 with 0 Axes>
```

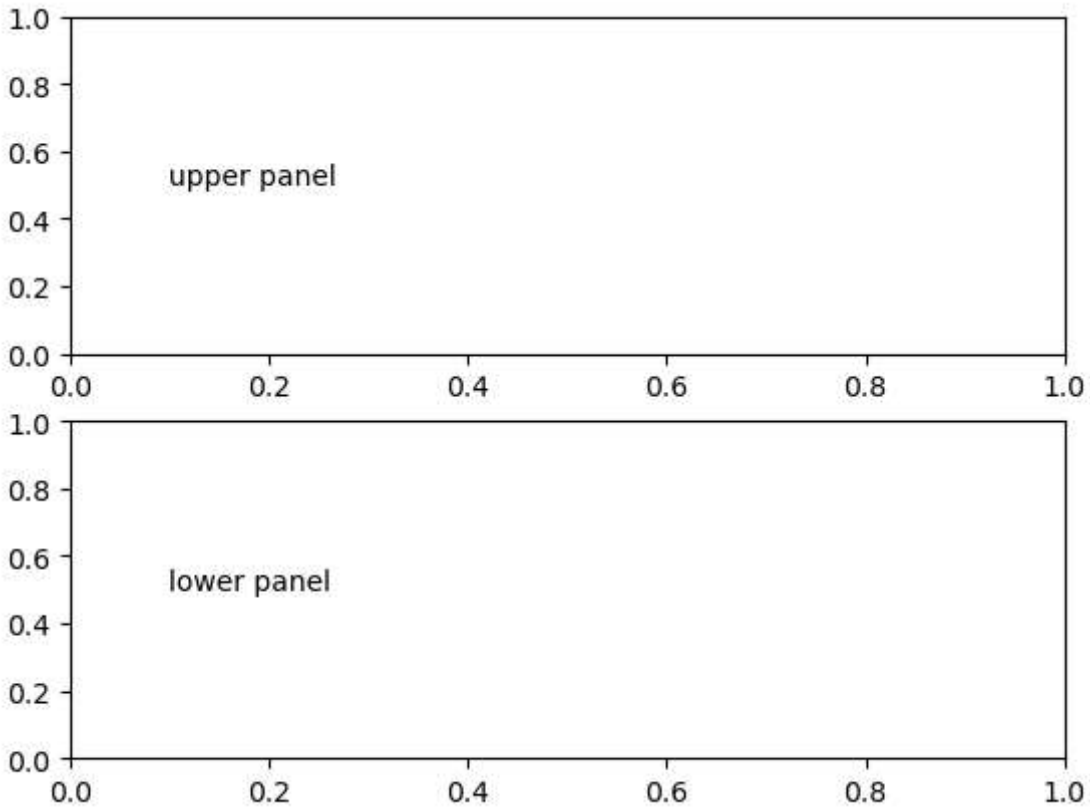
How to make axes

In [3]:
```
fig=plt.figure() # Preparing the canvas named as 'fig'
ax=fig.add_subplot(111) # Preparing the axes named as 'ax' in 'fig'. (111) means (ro
plt.show()
```

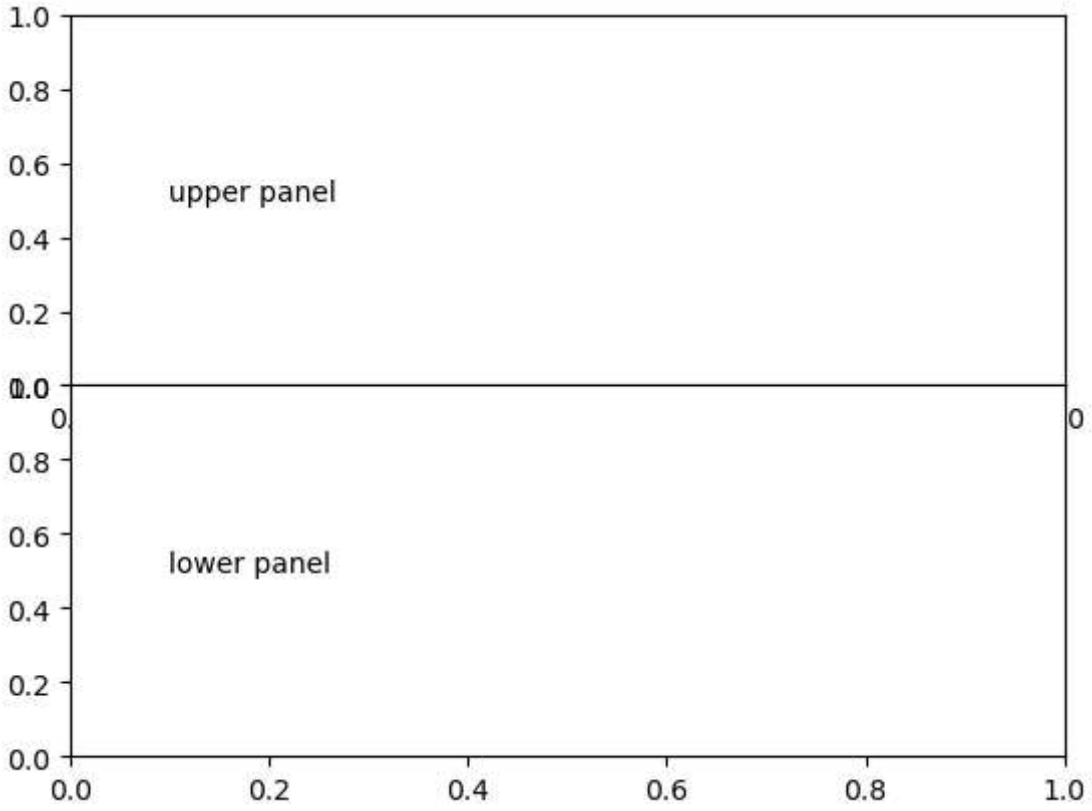


How to make multi panels

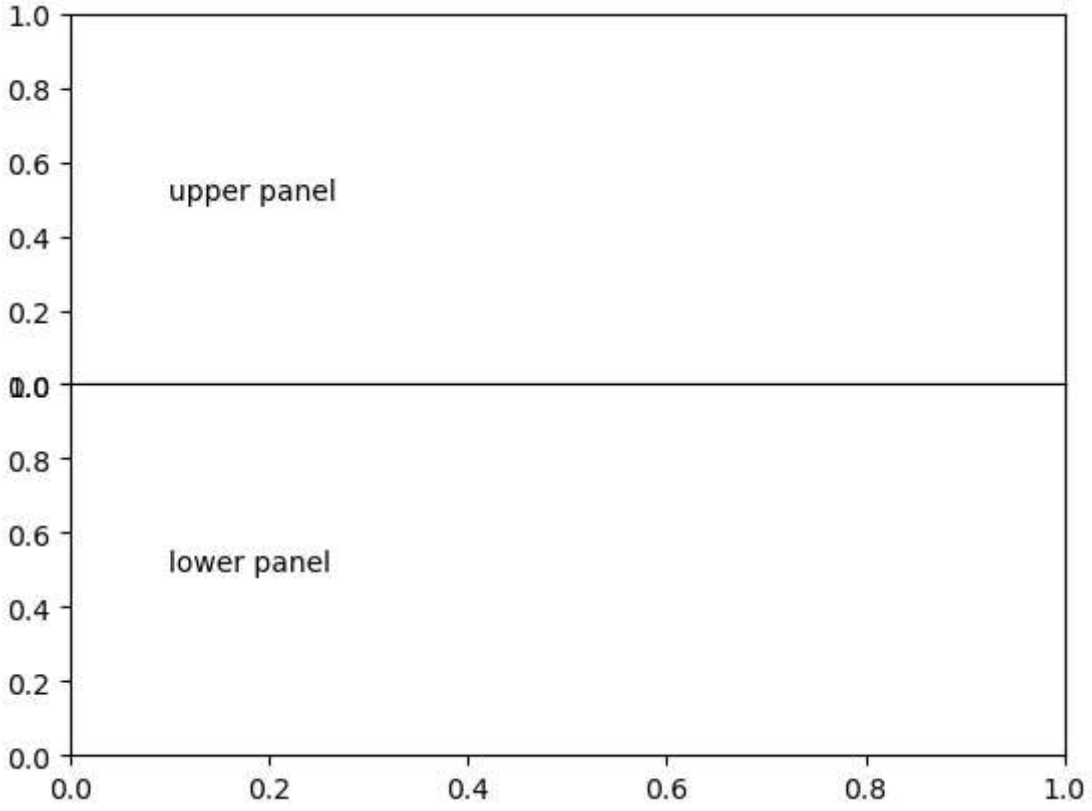
In [4]:
```
fig=plt.figure()
ax=fig.add_subplot(211)
ax.text(0.1,0.5,'upper panel')
ax1=fig.add_subplot(212)
ax1.text(0.1,0.5,'lower panel')
plt.show()
```



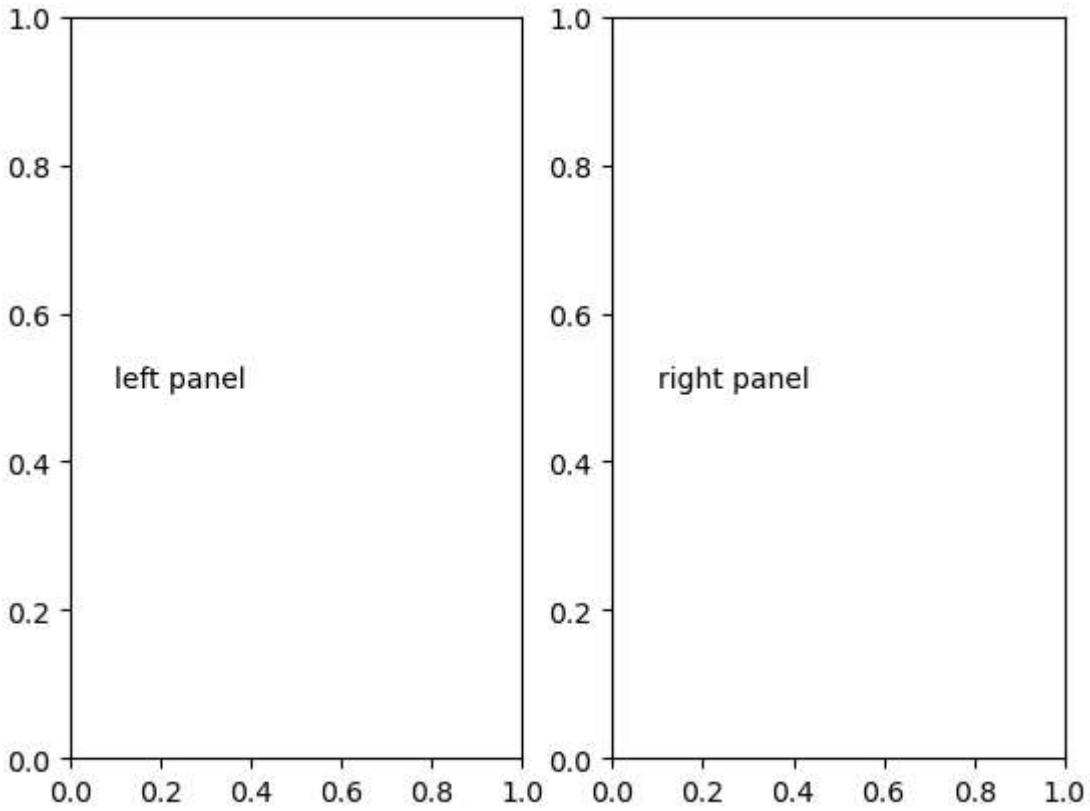
In [5]:
```
fig=plt.figure()
plt.subplots_adjust(hspace=0) # Remove the space between two panels
ax=fig.add_subplot(211)
ax.text(0.1,0.5,'upper panel')
ax1=fig.add_subplot(212)
ax1.text(0.1,0.5,'lower panel')
plt.show()
```



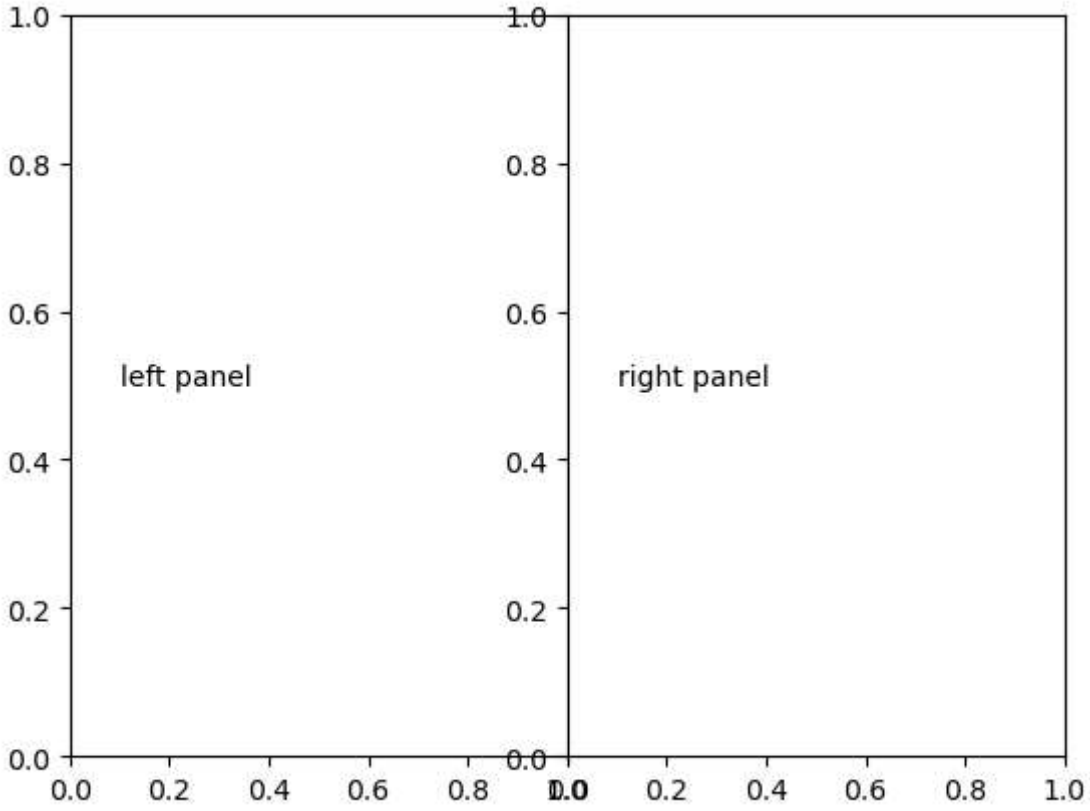
In [6]:
```
fig=plt.figure()
plt.subplots_adjust(hspace=0)
ax=fig.add_subplot(211)
ax.text(0.1,0.5,'upper panel')
ax.tick_params(labelbottom=None) # Adjust the tick parameters
ax1=fig.add_subplot(212)
ax1.text(0.1,0.5,'lower panel')
plt.show()
```



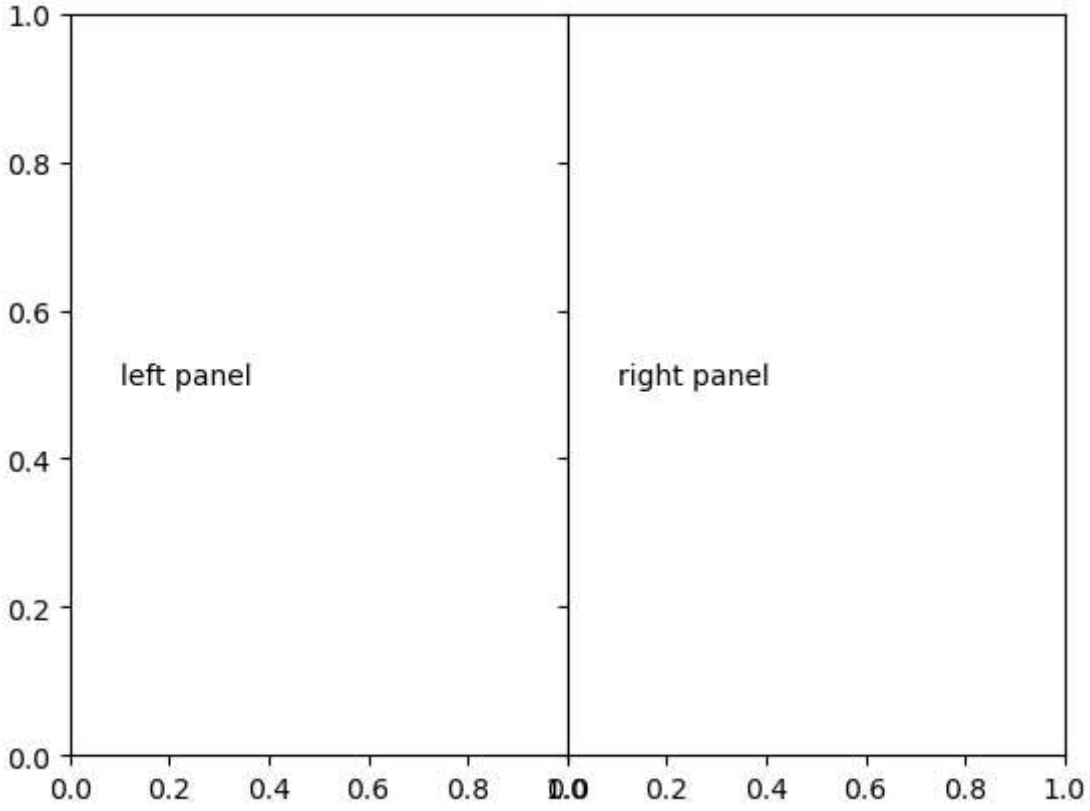
In [7]:
```
fig=plt.figure()
ax=fig.add_subplot(121)
ax.text(0.1,0.5,'left panel')
ax1=fig.add_subplot(122)
ax1.text(0.1,0.5,'right panel')
plt.show()
```



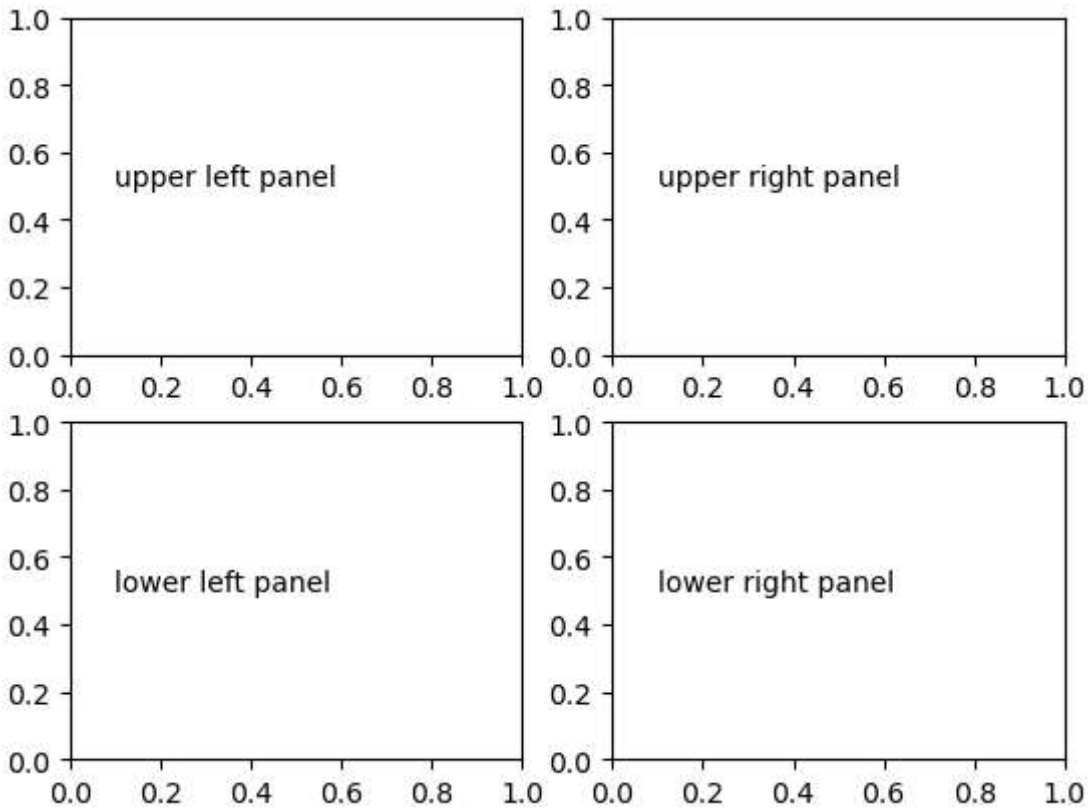
In [8]:
```
fig=plt.figure()
plt.subplots_adjust(wspace=0) # Delete the space between the panels
ax=fig.add_subplot(121)
ax.text(0.1,0.5,'left panel')
ax1=fig.add_subplot(122)
ax1.text(0.1,0.5,'right panel')
plt.show()
```



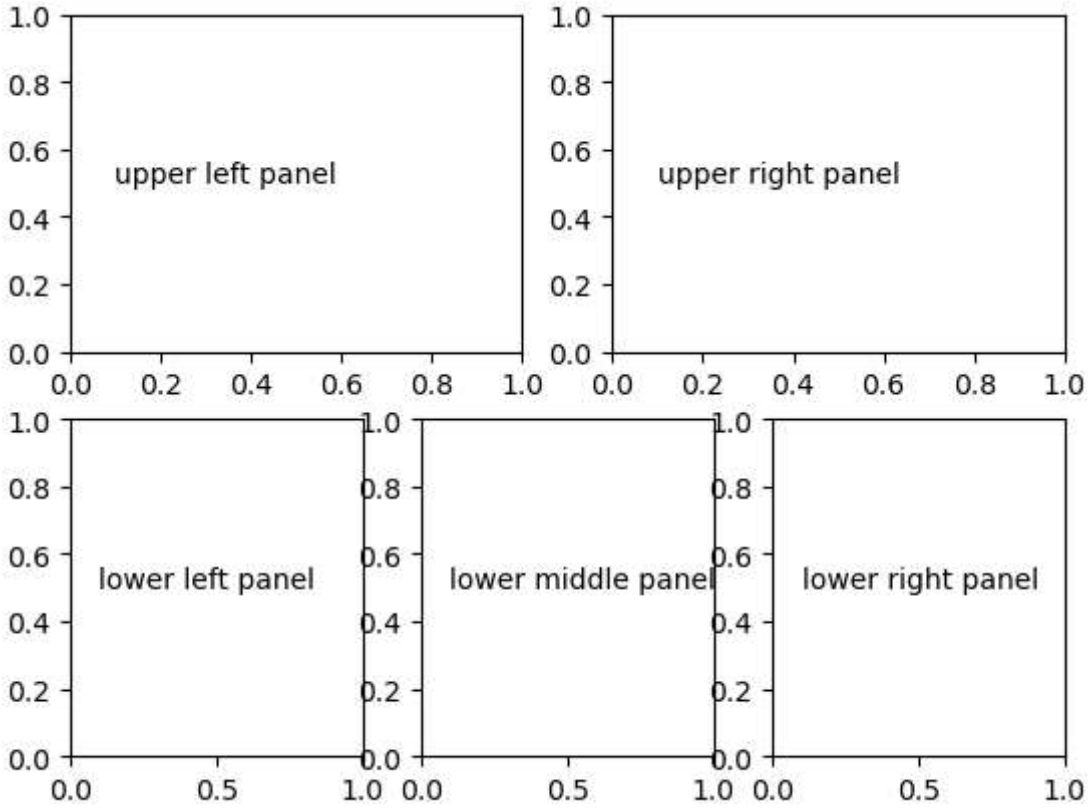
In [9]:
```
fig=plt.figure()
plt.subplots_adjust(wspace=0)
ax=fig.add_subplot(121)
ax.text(0.1,0.5,'left panel')
ax1=fig.add_subplot(122)
ax1.text(0.1,0.5,'right panel')
ax1.tick_params(labelleft=None)
plt.show()
```



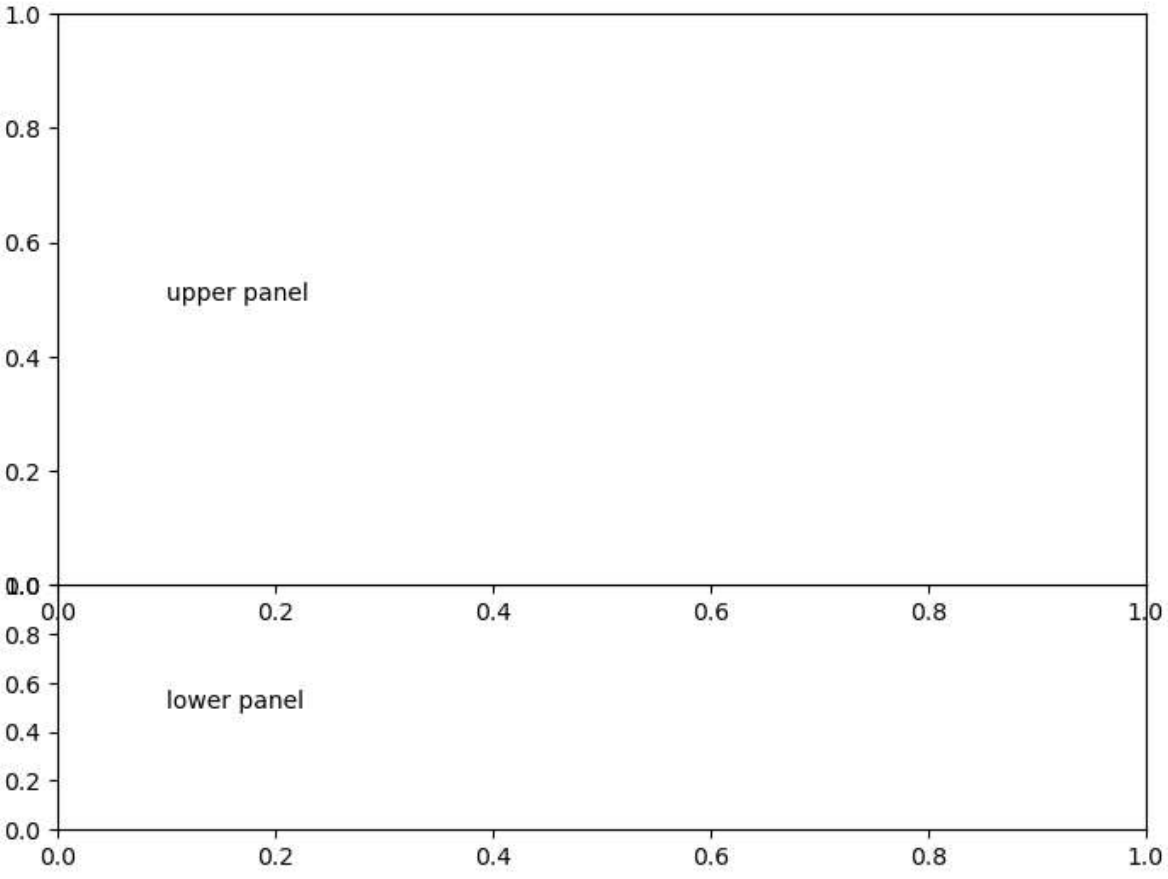
In [10]:
```
fig=plt.figure()
ax=fig.add_subplot(221)
ax.text(0.1,0.5,'upper left panel')
ax1=fig.add_subplot(222)
ax1.text(0.1,0.5,'upper right panel')
ax2=fig.add_subplot(223)
ax2.text(0.1,0.5,'lower left panel')
ax3=fig.add_subplot(224)
ax3.text(0.1,0.5,'lower right panel')
plt.show()
```



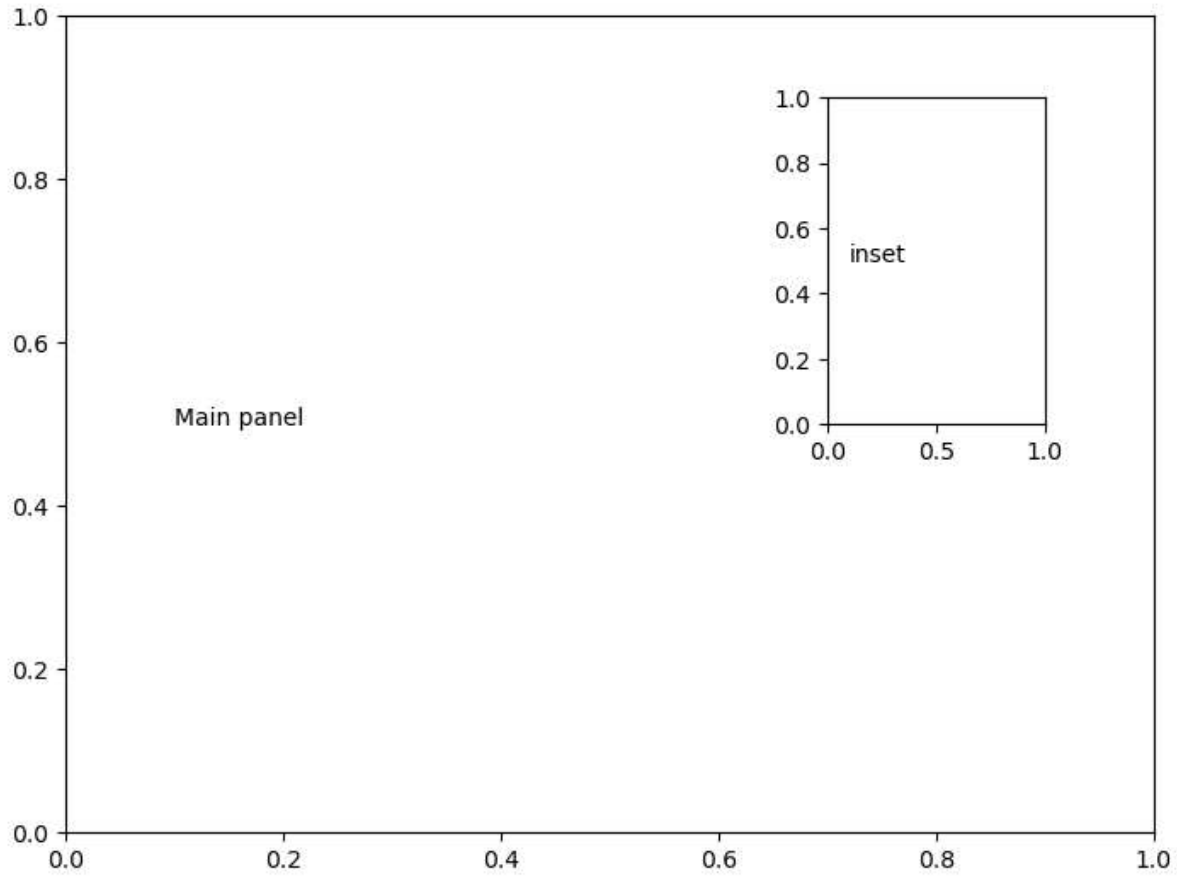
In [11]:
```
fig=plt.figure()
ax=fig.add_subplot(221)
ax.text(0.1,0.5,'upper left panel')
ax1=fig.add_subplot(222) # the 2nd panel of 2x2 panels
ax1.text(0.1,0.5,'upper right panel')
ax2=fig.add_subplot(234) # the 4th panel of 2x3 panels
ax2.text(0.1,0.5,'lower left panel')
ax3=fig.add_subplot(235)
ax3.text(0.1,0.5,'lower middle panel')
ax4=fig.add_subplot(236)
ax4.text(0.1,0.5,'lower right panel')
plt.show()
```



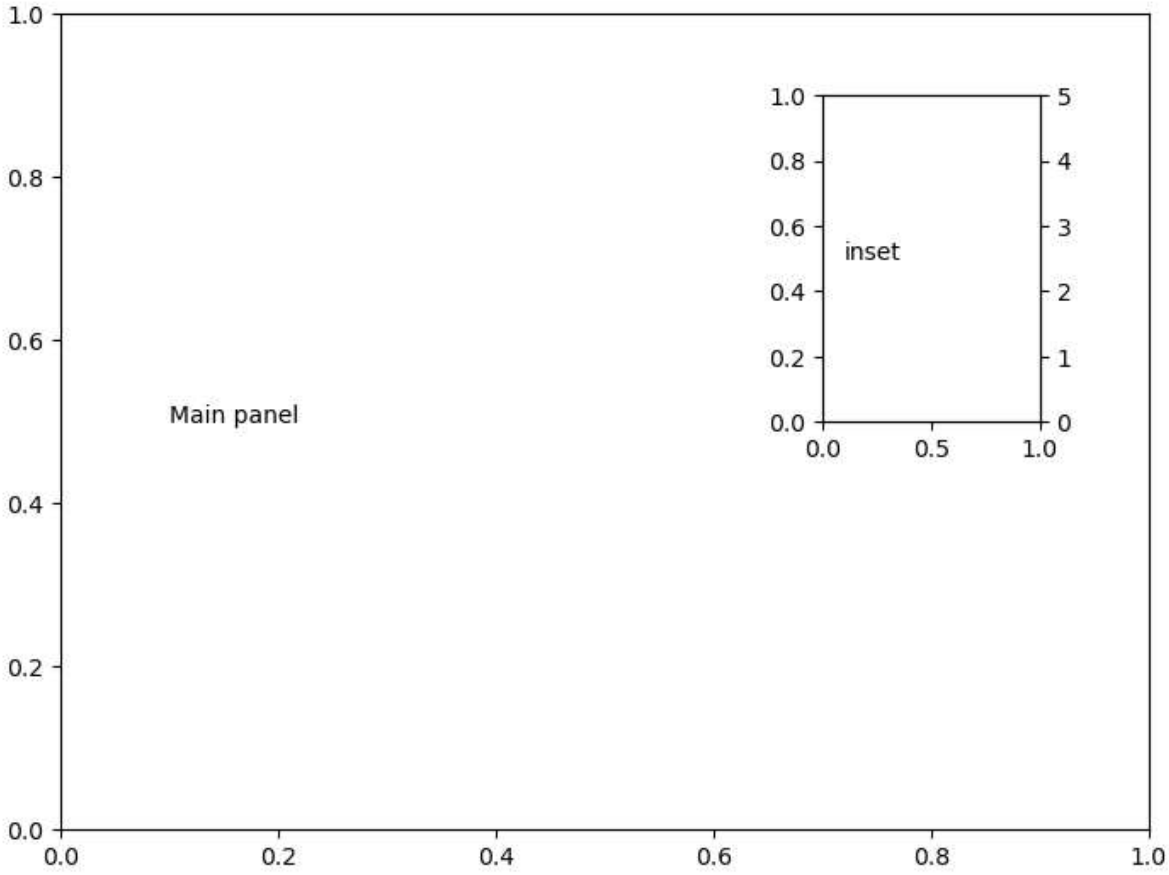
In [12]:
```
fig=plt.figure()
ax=fig.add_axes([0,0,1,0.3]) # [x value of lower and left position, y value of lower
ax.text(0.1,0.5,'lower panel')
ax1=fig.add_axes([0,0.3,1,0.7])
ax1.text(0.1,0.5,'upper panel')
plt.show()
```



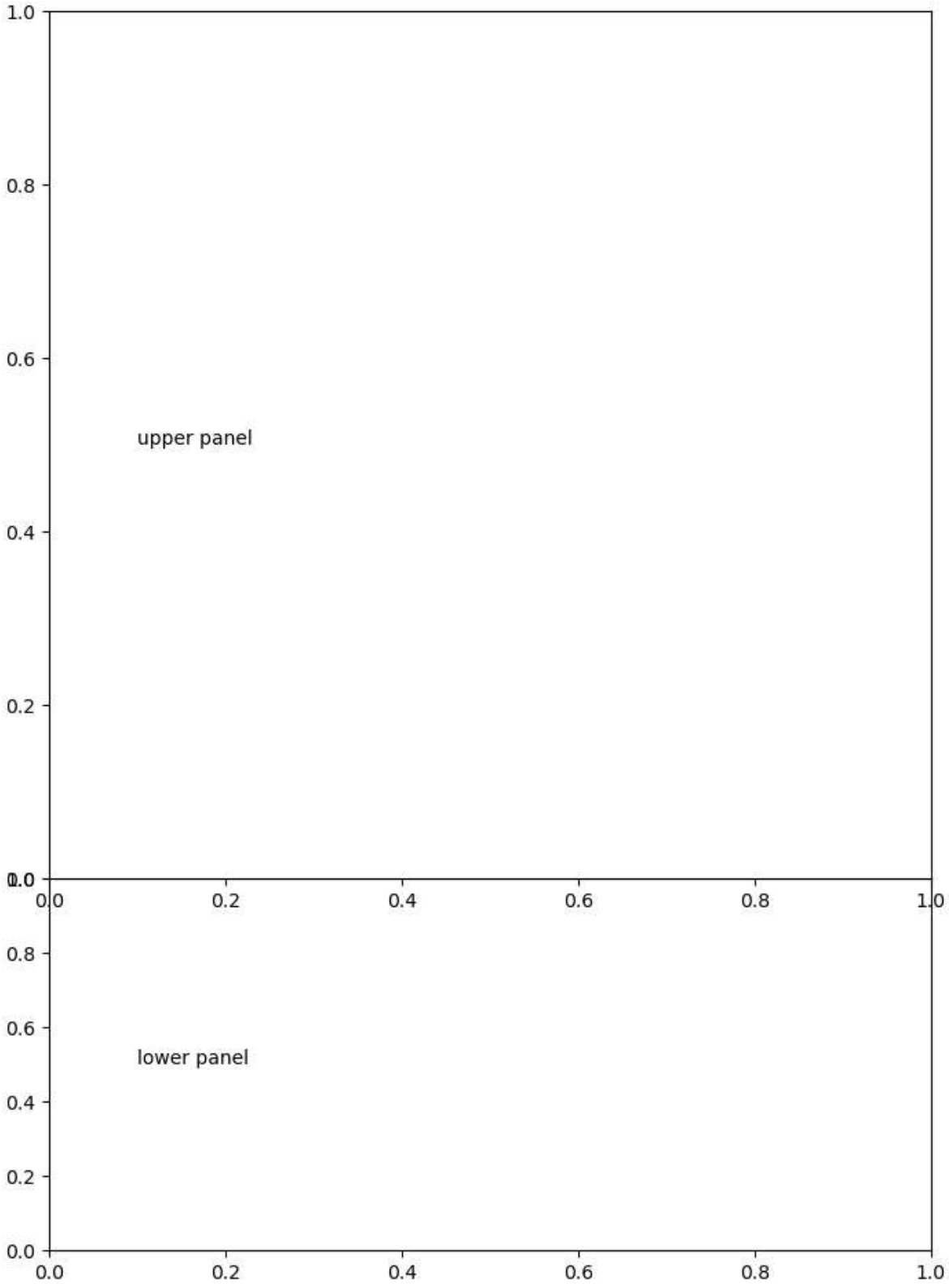
In [13]:
```
fig=plt.figure()
ax=fig.add_axes([0,0,1,1]) # [x value of lower and left position, y value of lower a
ax.text(0.1,0.5,'Main panel')
ax1=fig.add_axes([0.7,0.5,0.2,0.4]) # Making the inset figure
ax1.text(0.1,0.5,'inset')
plt.show()
```



In [14]:
```
fig=plt.figure()
ax=fig.add_axes([0,0,1,1]) # [x value of lower and left position, y value of lower a
ax.text(0.1,0.5,'Main panel')
ax1=fig.add_axes([0.7,0.5,0.2,0.4]) # Making the inset figure
ax1.text(0.1,0.5,'inset')
ax1_2=ax1.twinx()      # add new panel with the same x-axis of original panel at the
ax1_2.set_ylim(0,5)    # Legend of the y-axis are shown in the right axis
plt.show()
```



In [15]:
```
fig=plt.figure(figsize=(6.4,9))
ax=fig.add_axes([0,0,1,0.3]) # [x value of lower and left position, y value of lower
ax.text(0.1,0.5,'lower panel')
ax1=fig.add_axes([0,0.3,1,0.7])
ax1.text(0.1,0.5,'upper panel')
plt.show()
```



Plotting data

In [16]:
```
xx=np.arange(0,100,2) # Making array of arithmetic progression (start, end, step)
print(xx)
```

```
[ 0  2  4  6  8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46
 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94
 96 98]
```

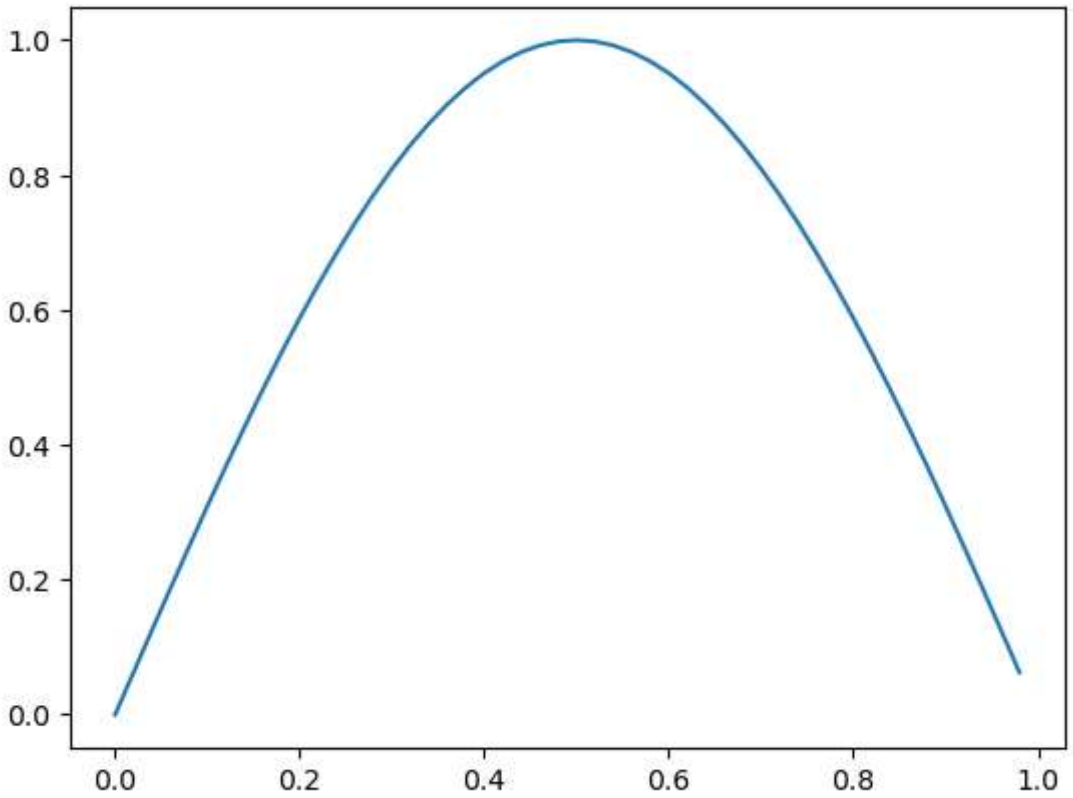
In [17]:
```
xx=xx/100
print(xx)
```

```
[0.   0.02 0.04 0.06 0.08 0.1  0.12 0.14 0.16 0.18 0.2  0.22 0.24 0.26
 0.28 0.3  0.32 0.34 0.36 0.38 0.4  0.42 0.44 0.46 0.48 0.5  0.52 0.54
 0.56 0.58 0.6  0.62 0.64 0.66 0.68 0.7  0.72 0.74 0.76 0.78 0.8  0.82
 0.84 0.86 0.88 0.9  0.92 0.94 0.96 0.98]
```

In [18]:
```
yy=np.sin(xx*np.pi)
print(yy)
```

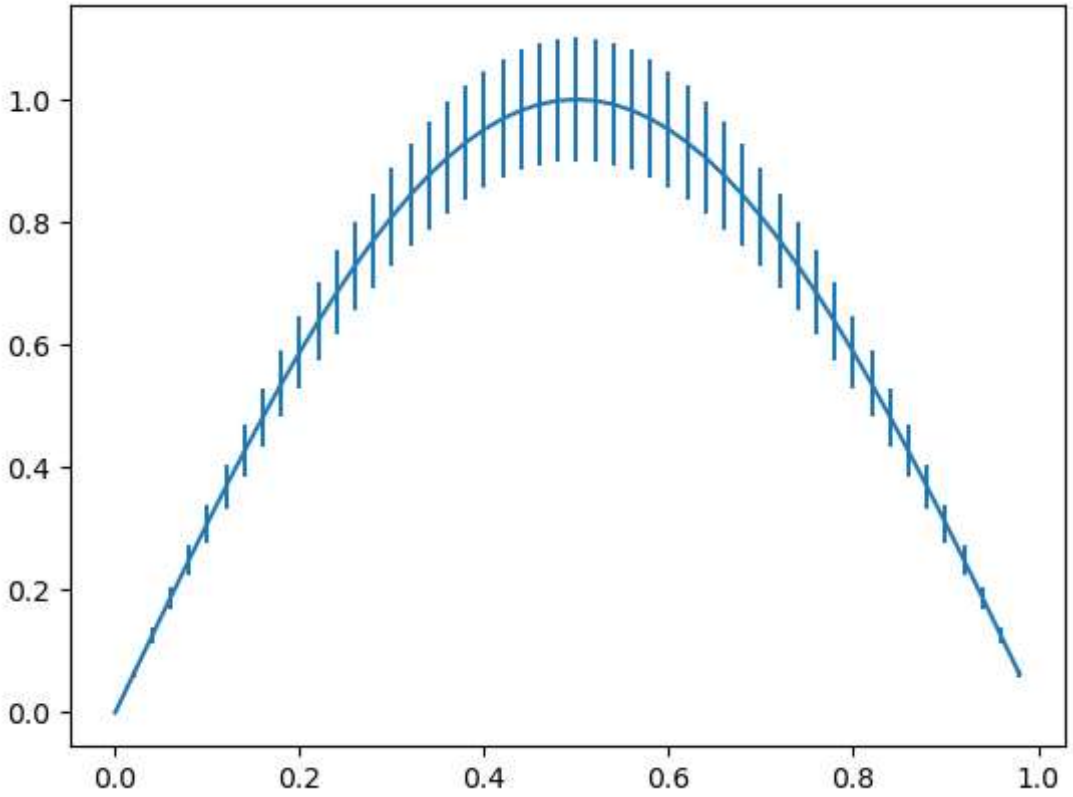
```
[0.         0.06279052 0.12533323 0.18738131 0.24868989 0.30901699
 0.36812455 0.42577929 0.48175367 0.53582679 0.58778525 0.63742399
 0.68454711 0.72896863 0.77051324 0.80901699 0.84432793 0.87630668
 0.90482705 0.92977649 0.95105652 0.96858316 0.98228725 0.9921147
 0.99802673 1.         0.99802673 0.9921147  0.98228725 0.96858316
 0.95105652 0.92977649 0.90482705 0.87630668 0.84432793 0.80901699
 0.77051324 0.72896863 0.68454711 0.63742399 0.58778525 0.53582679
 0.48175367 0.42577929 0.36812455 0.30901699 0.24868989 0.18738131
 0.12533323 0.06279052]
```

In [19]:
```
fig=plt.figure() # Preparing the canvas named as 'fig'
ax=fig.add_subplot(111) # Preparing the axes named as 'ax' in 'fig'
ax.plot(xx,yy) # Plotting (x,y) data
plt.show()
```

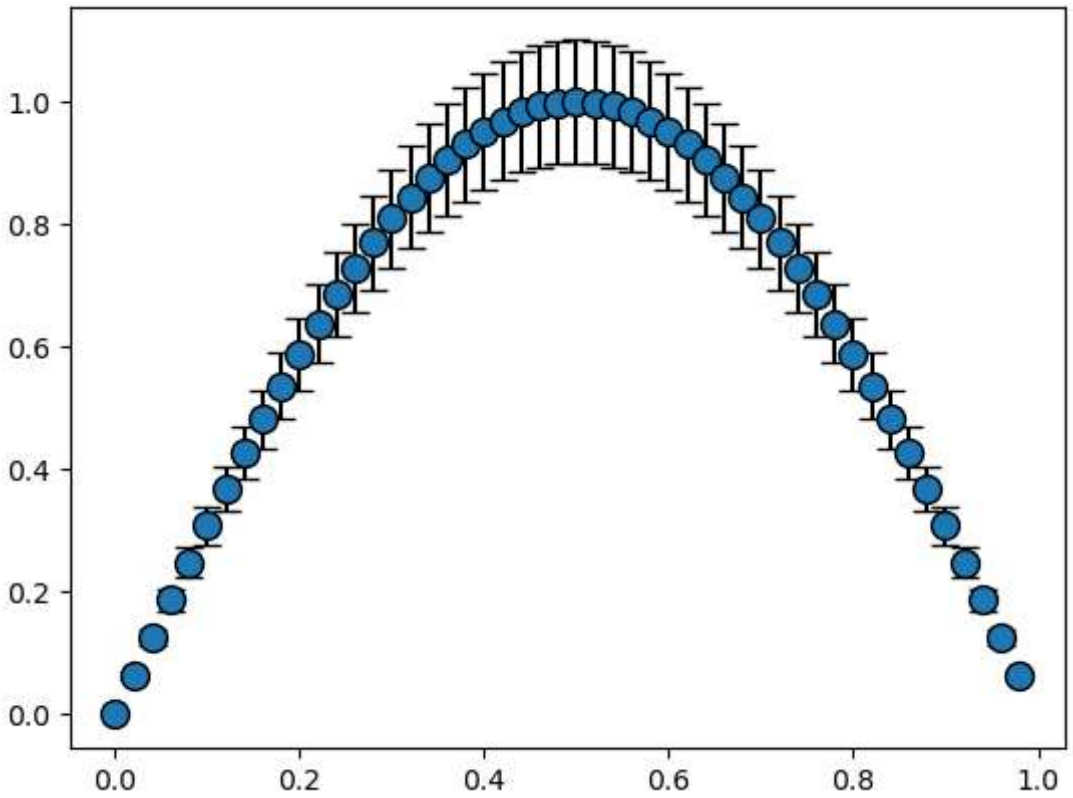


In [20]:
```
error_y=yy*0.1
```

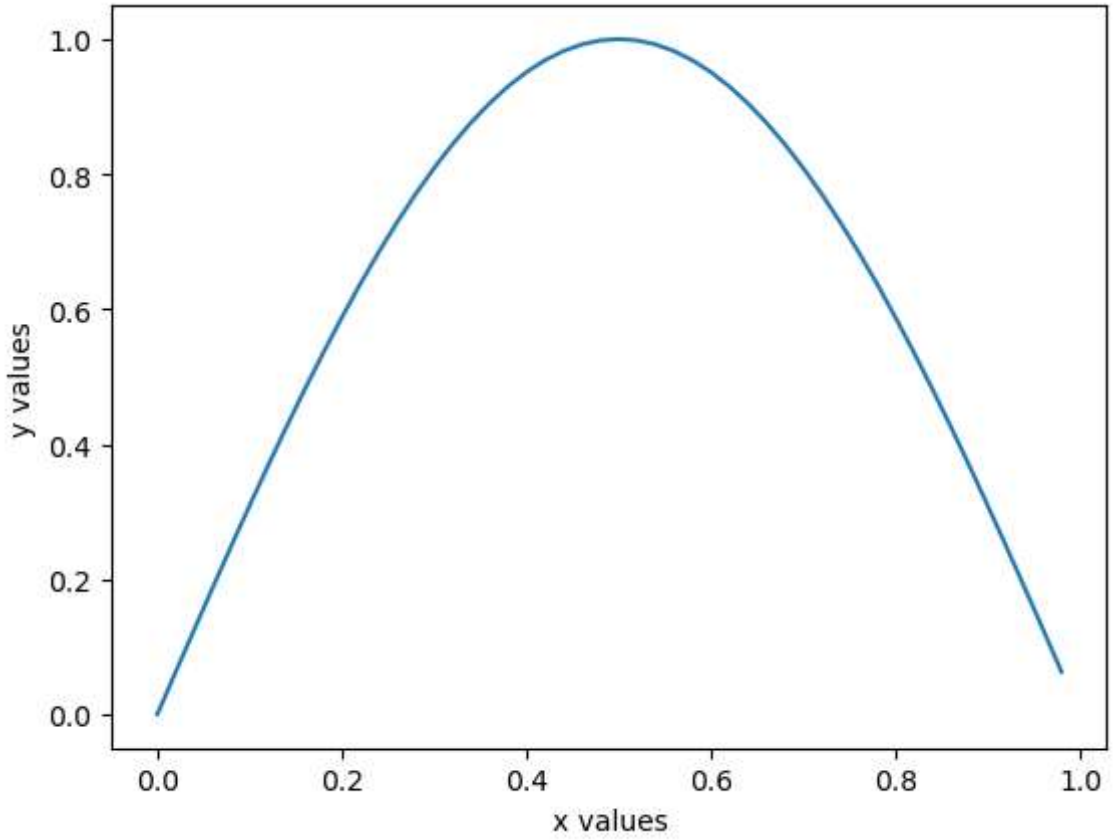
In [21]:
```
fig_error=plt.figure() # Preparing the canvas named as 'fig'
ax=fig_error.add_subplot(111) # Preparing the axes named as 'ax' in 'fig'
ax.errorbar(xx,yy,error_y)
plt.show()
```



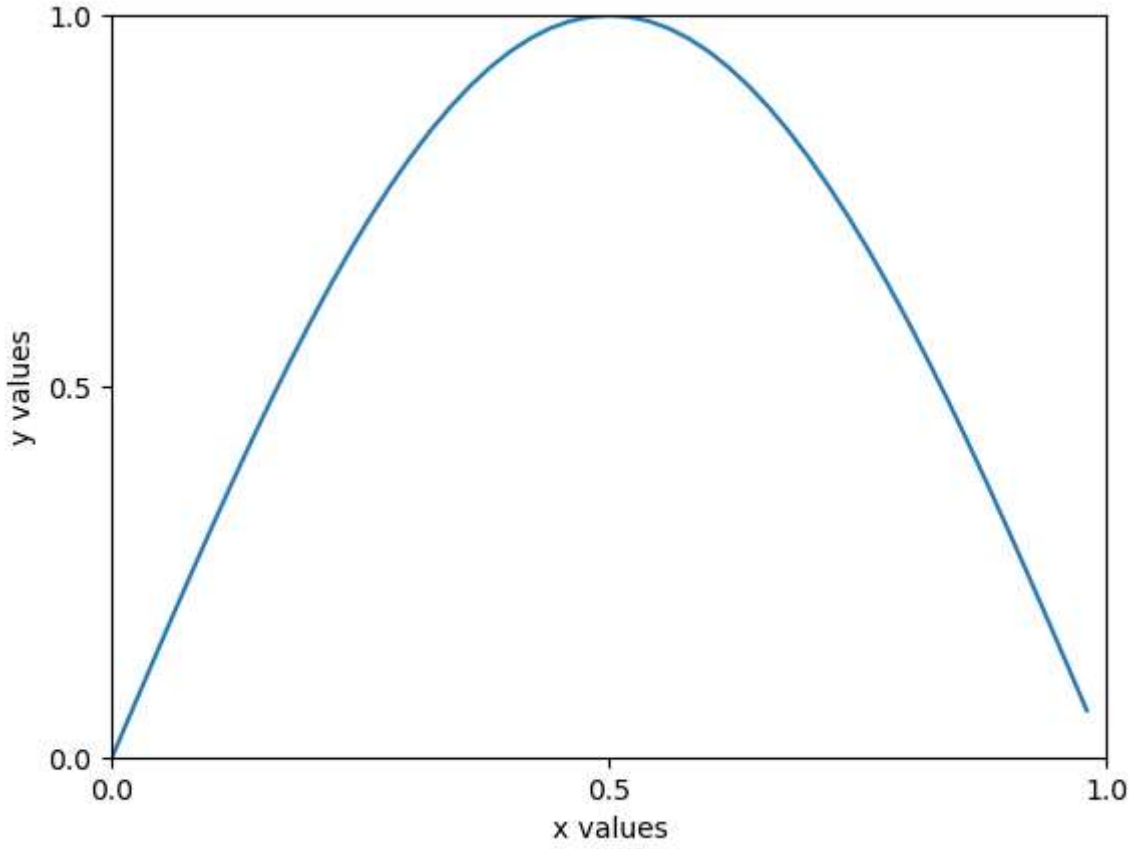
In [22]:
```
fig_error_2=plt.figure() # Preparing the canvas named as 'fig'
ax=fig_error_2.add_subplot(111) # Preparing the axes named as 'ax' in 'fig'
ax.errorbar(xx,yy,error_y,
            capsize=5, # Draw the end of errorbar
            fmt='o', # Draw circles (markers) at the data points
            markersize=10,
            ecolor='black', # Color of errorbar
            markeredgecolor = "black",
            # color='w' # Color inside the markers
           )
plt.show()
```



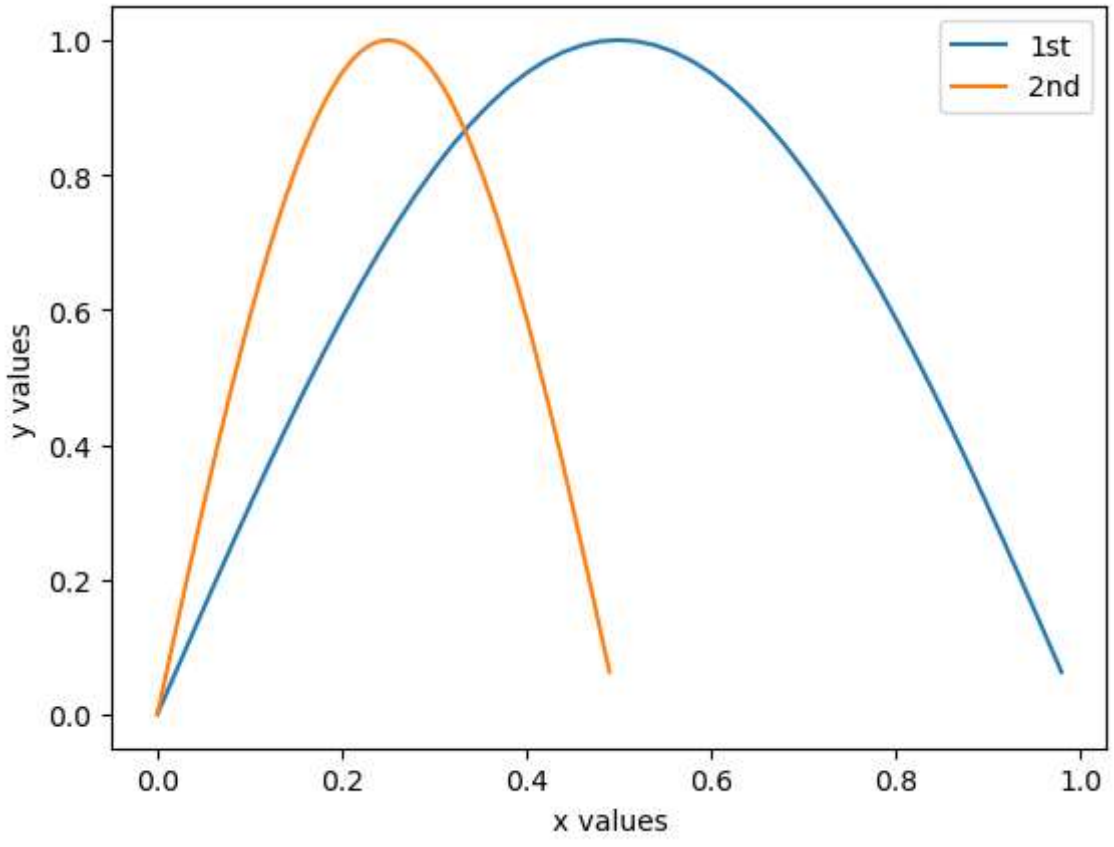
In [23]:
```
fig=plt.figure() # Preparing the canvas named as 'fig'
ax=fig.add_subplot(111) # Preparing the axes named as 'ax' in 'fig'
ax.plot(xx,yy) # Plotting (x,y) data
ax.set_xlabel('x values') # Title of the axis can be shown
ax.set_ylabel('y values')
plt.show()
```



In [24]:
```
fig=plt.figure() # Preparing the canvas named as 'fig'
ax=fig.add_subplot(111) # Preparing the axes named as 'ax' in 'fig'
ax.plot(xx,yy) # Plotting (x,y) data
ax.set_xlabel('x values') # Title of the axis can be shown
ax.set_ylabel('y values')
ax.set_xlim(0,1) # Set the x range
ax.set_xticks(np.arange(0,1.001,0.5))
ax.set_ylim(0,1) # Set the y range
ax.set_yticks(np.arange(0,1.001,0.5))
plt.show()
```

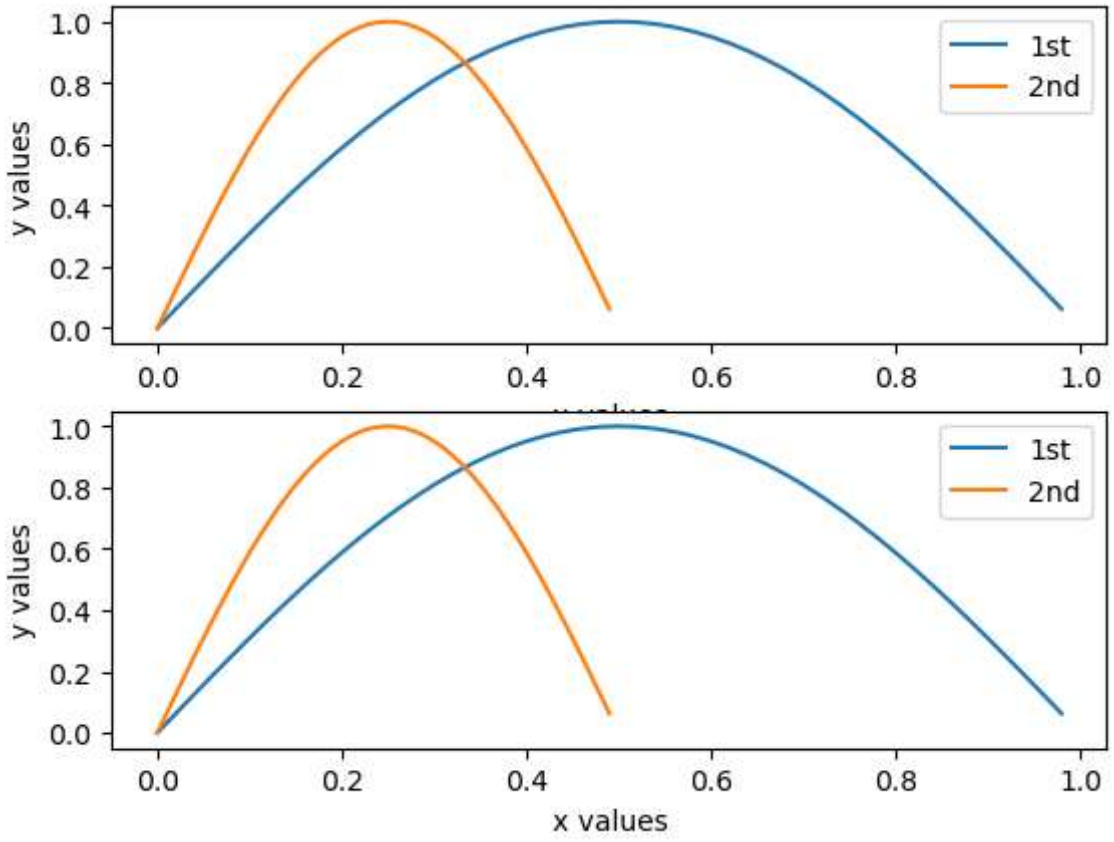


In [25]:
```
fig=plt.figure() # Preparing the canvas named as 'fig'
ax=fig.add_subplot(111) # Preparing the axes named as 'ax' in 'fig'
ax.plot(xx,yy, label='1st') # Plotting (x,y) data
ax.plot(xx/2,yy, label='2nd')
ax.set_xlabel('x values')
ax.set_ylabel('y values')
ax.legend() # legend can be shown if each plot has its label
plt.show()
```

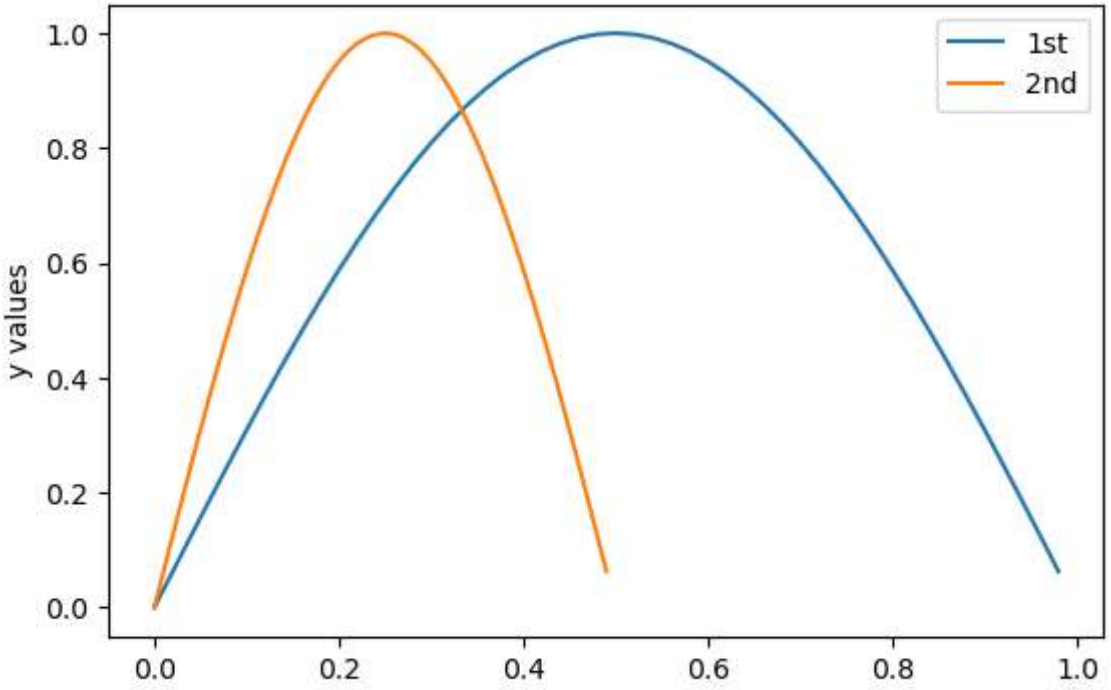


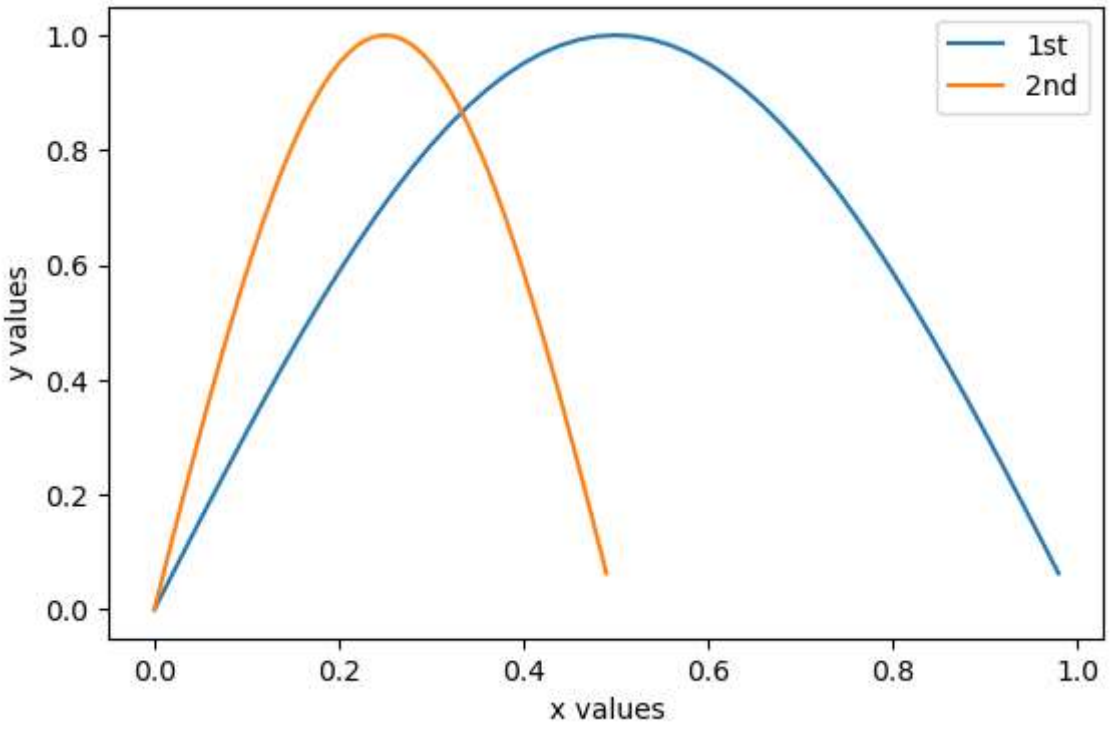
In [26]:
```
fig.savefig('temp_sin.svg') # Graph can be saved as a graphic file. "svg" formatted
```

In [27]:
```
fig1=plt.figure() # Preparing the canvas named as 'fig1'

ax=fig1.add_subplot(211) # Preparing the axes named as 'ax' in 'fig1'
ax.plot(xx,yy, label='1st') # Plotting (x,y) data
ax.plot(xx/2,yy, label='2nd')
ax.set_xlabel('x values')
ax.set_ylabel('y values')
ax.legend()

ax1=fig1.add_subplot(212) # Preparing the axes named as 'ax1' in 'fig1'
ax1.plot(xx,yy, label='1st') # Plotting (x,y) data
ax1.plot(xx/2,yy, label='2nd')
ax1.set_xlabel('x values')
ax1.set_ylabel('y values')
ax1.legend()
plt.show()
```



In [28]:
```
fig2=plt.figure(figsize=(6.4,9)) # Preparing the canvas named as 'fig2'. Size of fig

ax=fig2.add_subplot(211) # Preparing the axes named as 'ax' in 'fig2'
ax.plot(xx,yy, label='1st') # Plotting (x,y) data
ax.plot(xx/2,yy, label='2nd')
# ax.set_xlabel('x values')
ax.set_ylabel('y values')
ax.legend()

ax1=fig2.add_subplot(212) # Preparing the axes named as 'ax1' in 'fig2'
ax1.plot(xx,yy, label='1st') # Plotting (x,y) data
ax1.plot(xx/2,yy, label='2nd')
ax1.set_xlabel('x values')
ax1.set_ylabel('y values')
ax1.legend()
plt.show()
```

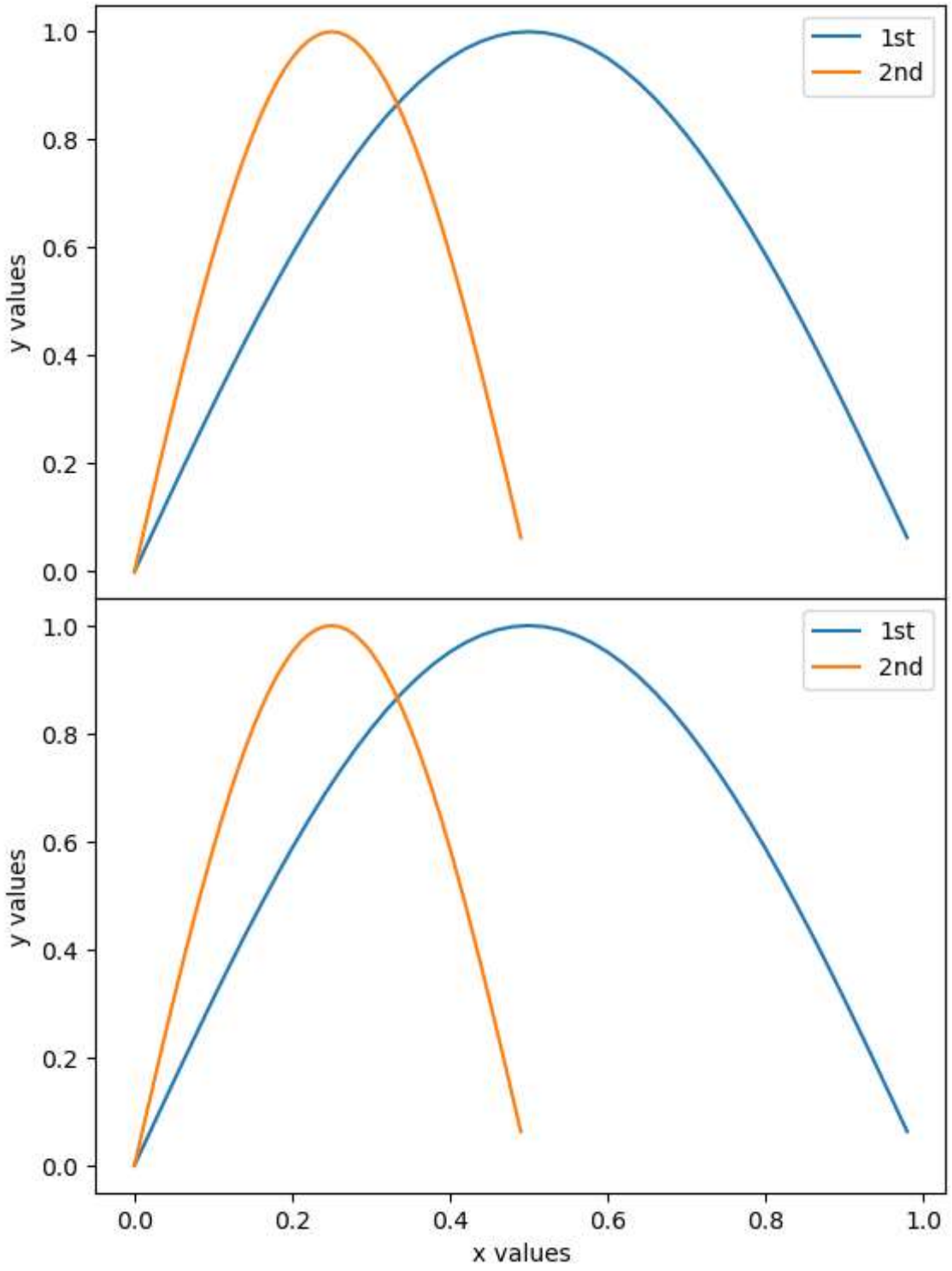




In [29]:

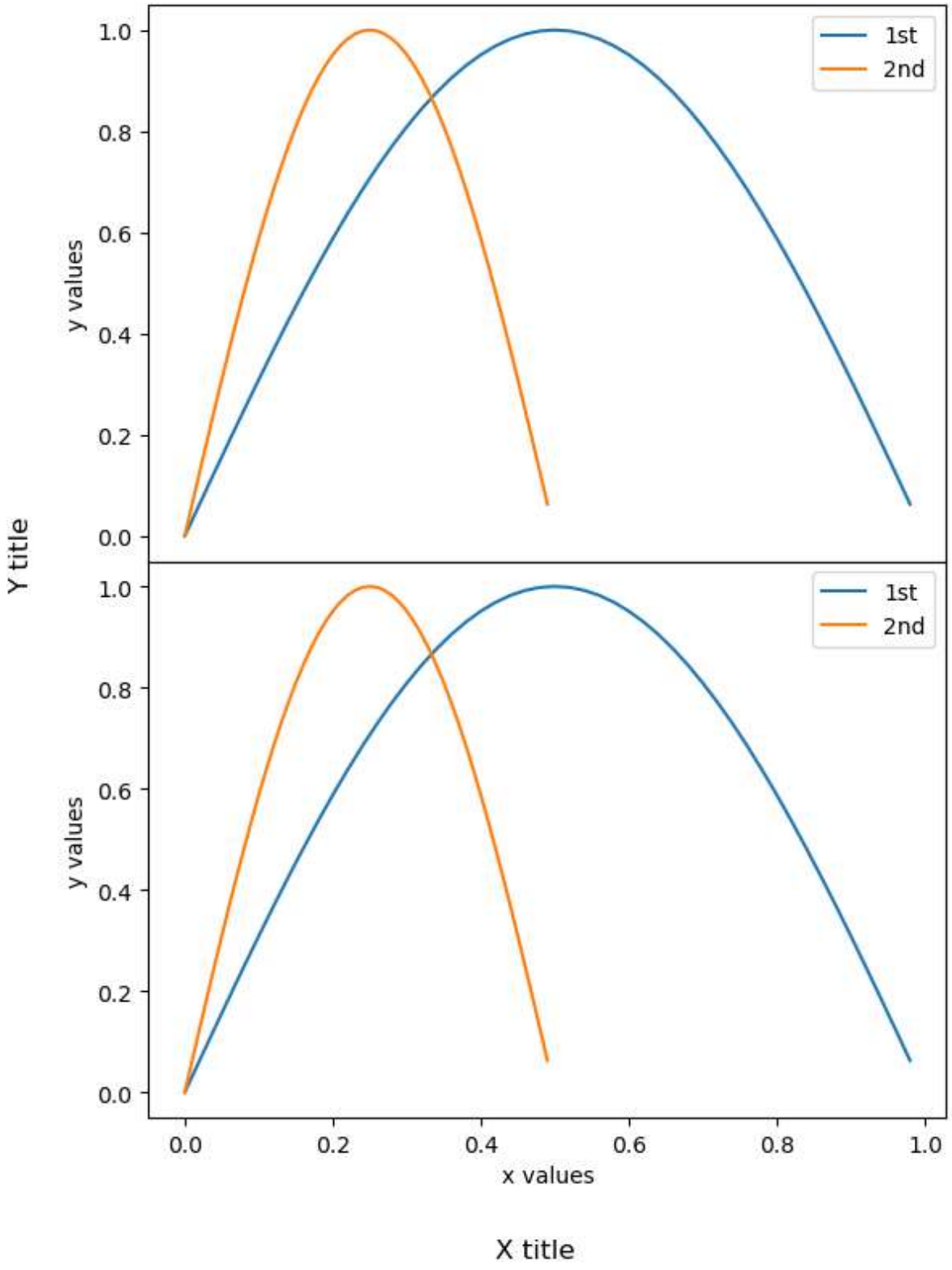
```
fig3=plt.figure(figsize=(6.4,9)) # Preparing the canvas named as 'fig3'

plt.subplots_adjust(hspace=0) # Adjust the space between the panels

ax=fig3.add_subplot(211) # Preparing the axes named as 'ax' in 'fig3'
ax.plot(xx,yy, label='1st') # Plotting (x,y) data in 'ax'
ax.plot(xx/2,yy, label='2nd')
# ax.set_xlabel('x values')
ax.tick_params(labelbottom=None) # Erasing the xticklabels in the upper panel
ax.set_ylabel('y values')
ax.legend()

ax1=fig3.add_subplot(212) # Preparing the axes named as 'ax1' in 'fig3'
ax1.plot(xx,yy, label='1st') # Plotting (x,y) data in 'ax1'
ax1.plot(xx/2,yy, label='2nd')
ax1.set_xlabel('x values')
```

```
ax1.set_ylabel('y values')
ax1.legend()
plt.show()
```



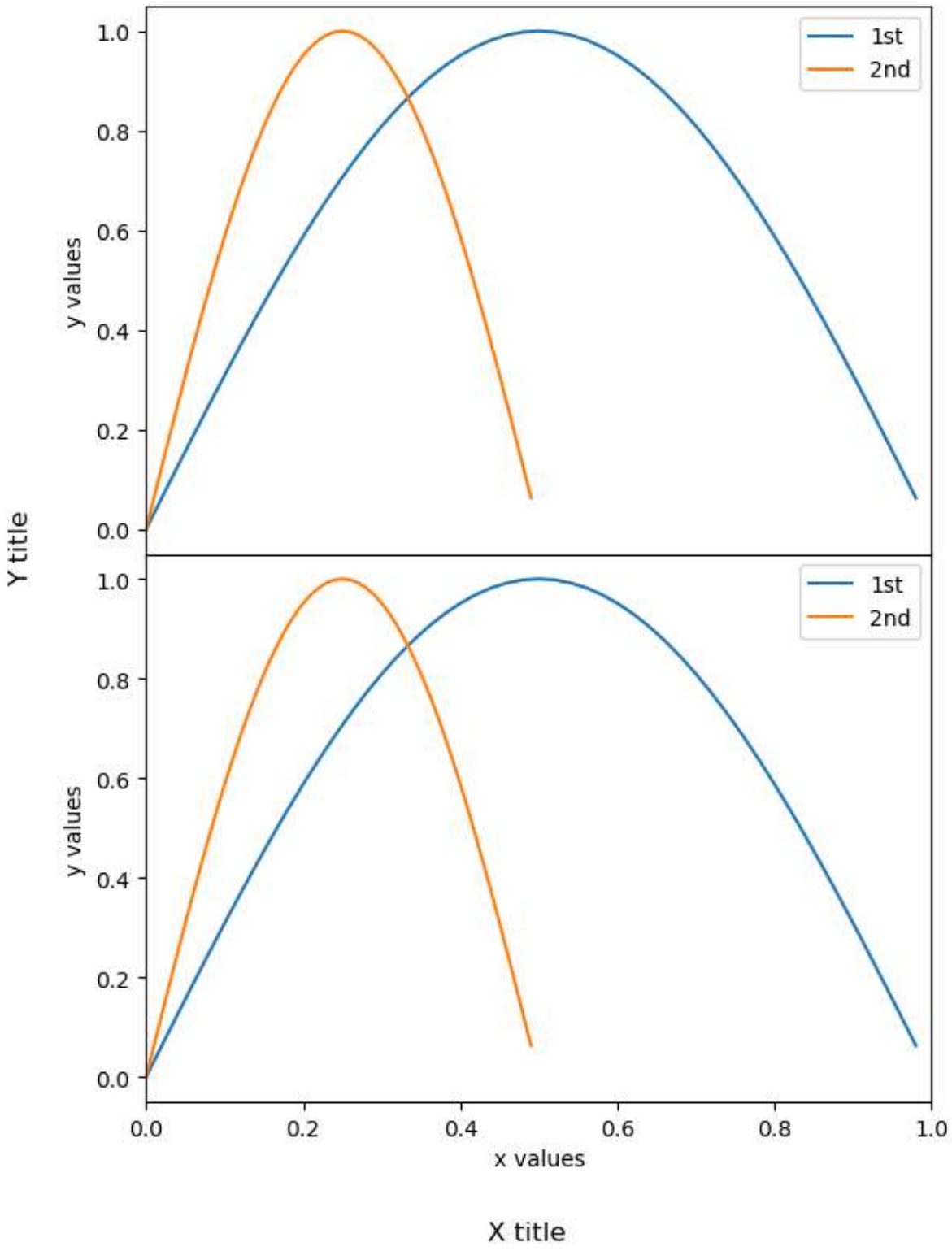
In [30]:
```
fig4=plt.figure(figsize=(6.4,9)) # Preparing the canvas named as 'fig4'

plt.subplots_adjust(hspace=0) # Adjust the space between the panels

ax=fig4.add_subplot(211) # Preparing the axes named as 'ax' in 'fig4'
ax.plot(xx,yy, label='1st') # Plotting (x,y) data in 'ax'
ax.plot(xx/2,yy, label='2nd')
# ax.set_xlabel('x values')
ax.tick_params(labelbottom=None) # Erasing the xticklabels in the upper panel
ax.set_ylabel('y values')
ax.legend()
```

```
ax1=fig4.add_subplot(212) # Preparing the axes named as 'ax1' in 'fig4'
ax1.plot(xx,yy, label='1st') # Plotting (x,y) data in 'ax1'
ax1.plot(xx/2,yy, label='2nd')
ax1.set_xlabel('x values')
ax1.set_ylabel('y values')

fig4.supxlabel('X title')
fig4.supylabel('Y title', x=-0.01)
ax1.legend()
plt.show()
```



In [31]:
```
fig4_1=plt.figure(figsize=(6.4,9)) # Preparing the canvas named as 'fig4_1'

plt.subplots_adjust(hspace=0) # Adjust the space between the panels
```

```
ax=fig4_1.add_subplot(211) # Preparing the axes named as 'ax' in 'fig4_1'
ax.plot(xx,yy, label='1st') # Plotting (x,y) data in 'ax'
ax.plot(xx/2,yy, label='2nd')
# ax.set_xlabel('x values')
ax.set_xlim(0,1)
ax.tick_params(labelbottom=None) # Erasing the xticklabels in the upper panel
ax.set_ylabel('y values')
ax.legend()

ax1=fig4_1.add_subplot(212, sharex=ax) # Preparing the axes named as 'ax1' in 'fig4_
ax1.plot(xx,yy, label='1st') # Plotting (x,y) data in 'ax1'
ax1.plot(xx/2,yy, label='2nd')
ax1.set_xlim(0,1)
ax1.set_xlabel('x values')
ax1.set_ylabel('y values')

fig4_1.supxlabel('X title') # X-axis title for the figure, not for axis
fig4_1.supylabel('Y title', x=-0.01) # Y-axis title for the figure, not for axis
ax1.legend()

plt.show()
```



Checking the default setting of matplotlib

In [32]:
```
import matplotlib as mpl
mpl.matplotlib_fname() # Showing the default setting file
```

Out[32]:
```
'C:¥¥Users¥¥tadai¥¥anaconda3¥¥lib¥¥site-packages¥¥matplotlib¥¥mpl-data¥¥matplotlibrc'
```

In [33]:
```
print(plt.style.available) # Showing the stylesheets available
```

```
['Solarize_Light2', 'Yifeng', '_classic_test_patch', '_mpl-gallery', '_mpl-gallery-nogrid', 'bmh', 'classic', 'dark_background', 'fast', 'fivethirtyeight', 'ggplot', 'grayscale', 'mystyle', 'seaborn-v0_8', 'seaborn-v0_8-bright', 'seaborn-v0_8-colorblind', 'seaborn-v0_8-dark', 'seaborn-v0_8-dark-palette', 'seaborn-v0_8-darkgrid', 'seaborn-v0_8-deep', 'seaborn-v0_8-muted', 'seaborn-v0_8-notebook', 'seaborn-v0_8-paper', 'seaborn-v0_8-pastel', 'seaborn-v0_8-poster', 'seaborn-v0_8-talk', 'seaborn-v0_8-ticks', 'seaborn-v0_8-white', 'seaborn-v0_8-whitegrid', 'tableau-colorblind10']
```

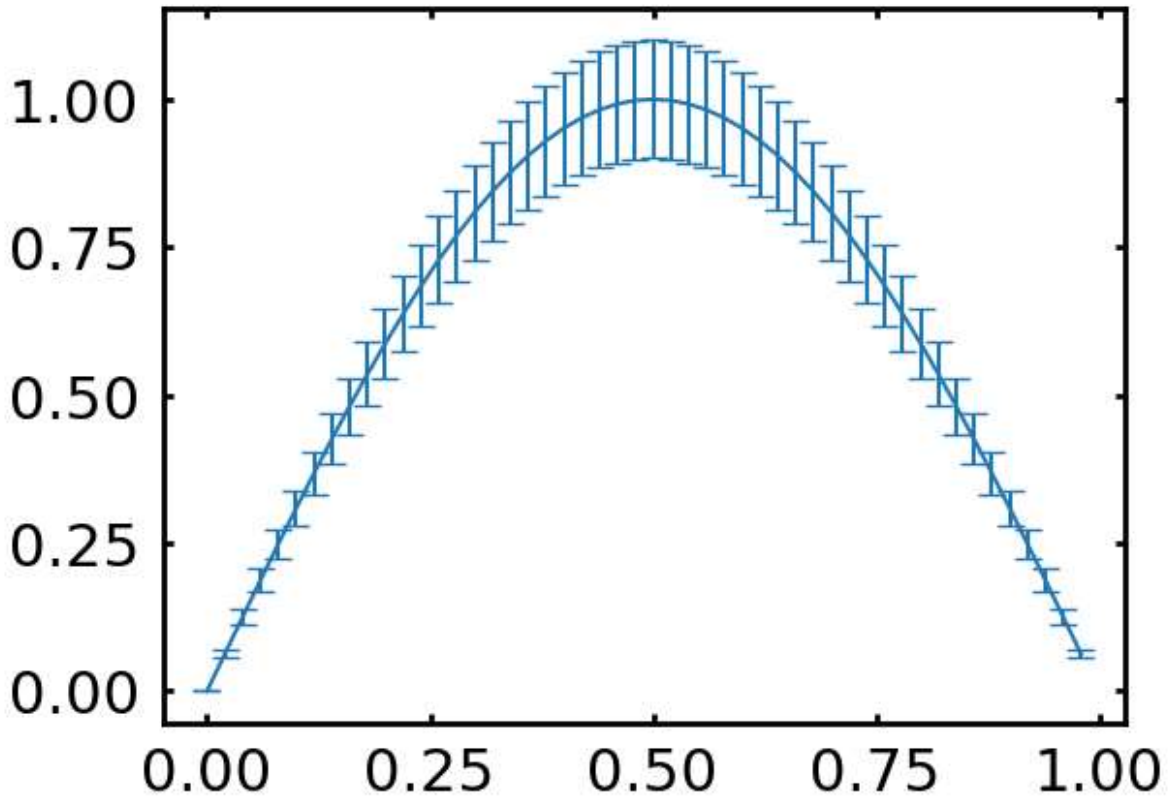
In [34]:
```
mpl.get_configdir() # Showing the folder name of which contains the configuration fi
                    # You can make your own stylesheet in the folder, '"configdir_na
```

Out[34]:
```
'C:¥¥Users¥¥tadai¥¥.matplotlib'
```

How to use my stylesheet

In [35]:
```
plt.style.use('mystyle') # You can use your own stylesheet
                # (Important) When you start the jupyterlab, magic command, % matplo
                # At least, the setting of figure size has been changed to [6.0, 4.0
                # Or define the setting to your stylesheet and use it.
```

In [36]:
```
# the same script in cell [21]
fig_error=plt.figure() # Preparing the canvas named as 'fig_error'
ax=fig_error.add_subplot(111) # Preparing the axes named as 'ax' in 'fig_error'
ax.errorbar(xx,yy,error_y)
plt.show()
```



Using dataframes

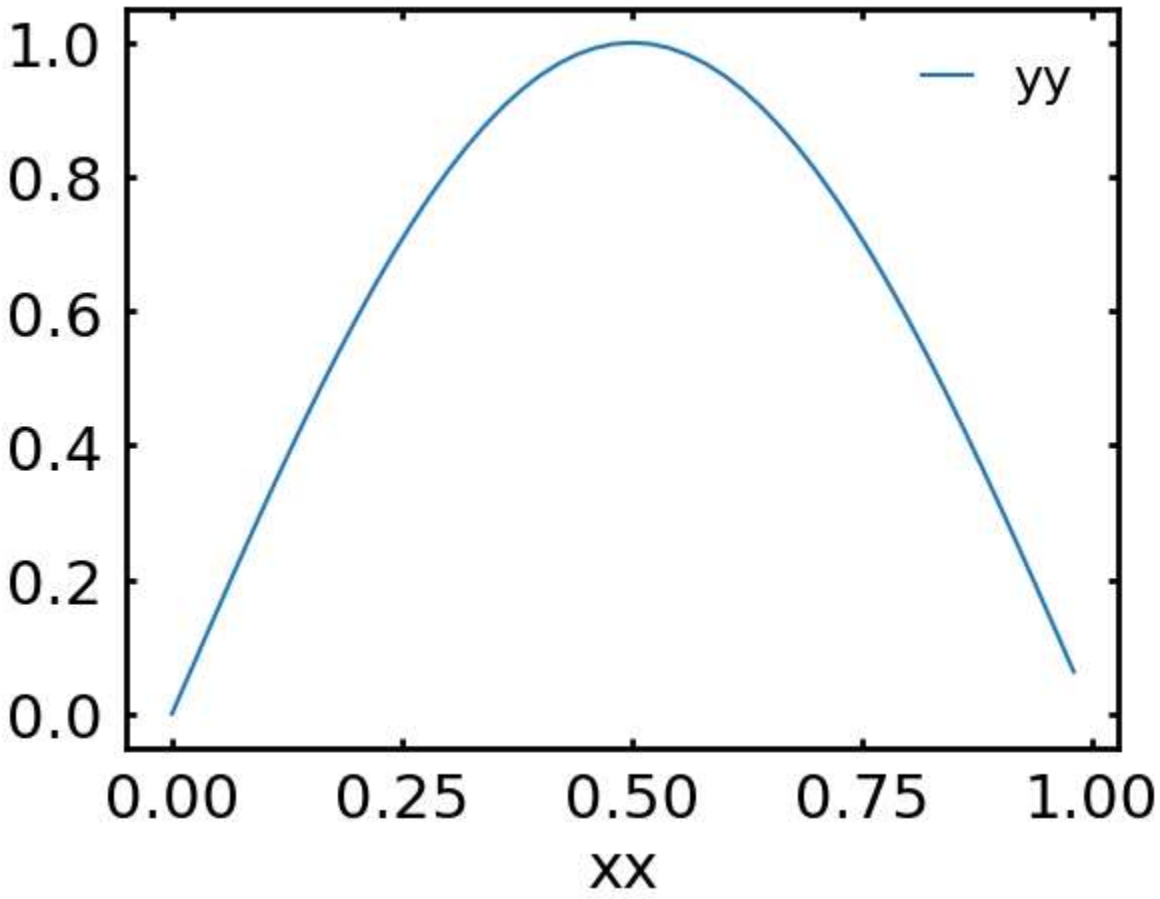
In [37]:
```
df=pd.DataFrame(index=xx, data=yy, columns=['yy']) # Definition of the dataframe
df.index.name='xx'
df.head()
```

Out[37]:

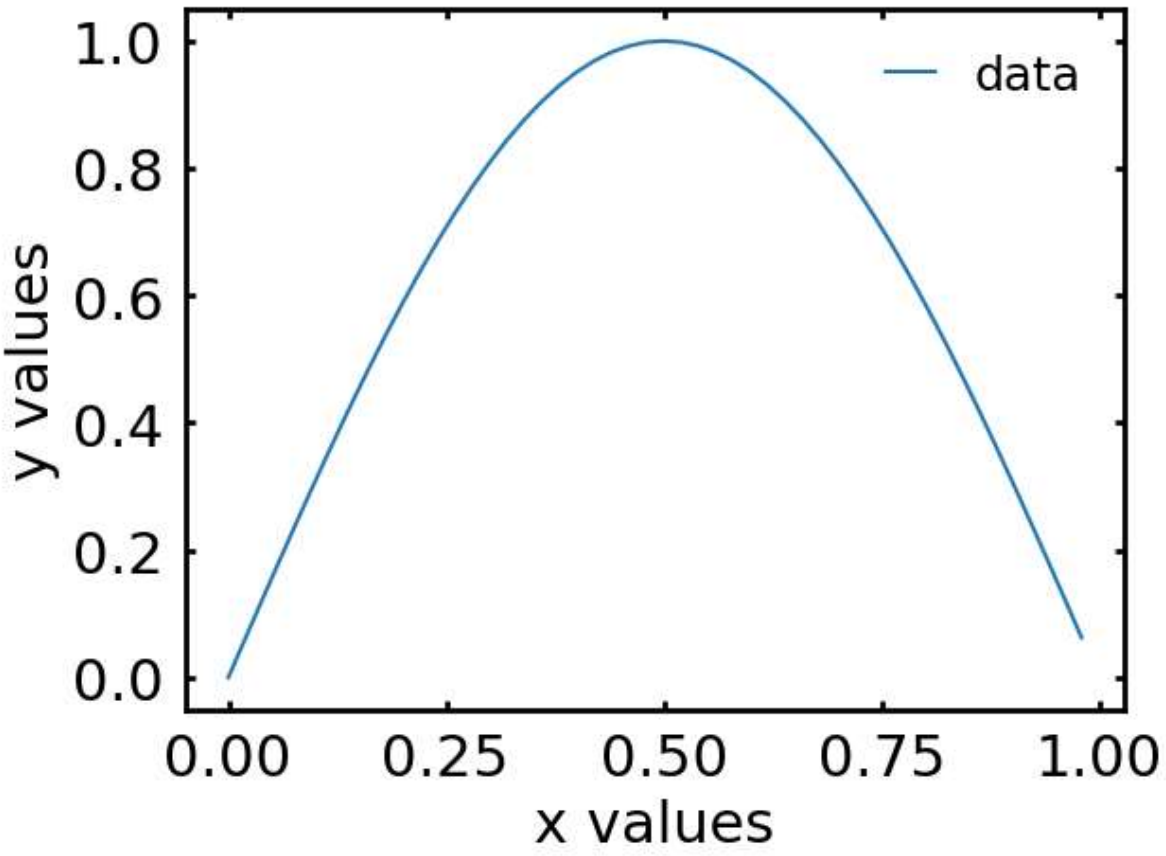
xx	yy
0.00	0.000000
0.02	0.062791
0.04	0.125333
0.06	0.187381
0.08	0.248690

In [38]:
```
df.plot() # This is the easiest way to check the data in the dataframe by your eyes
```

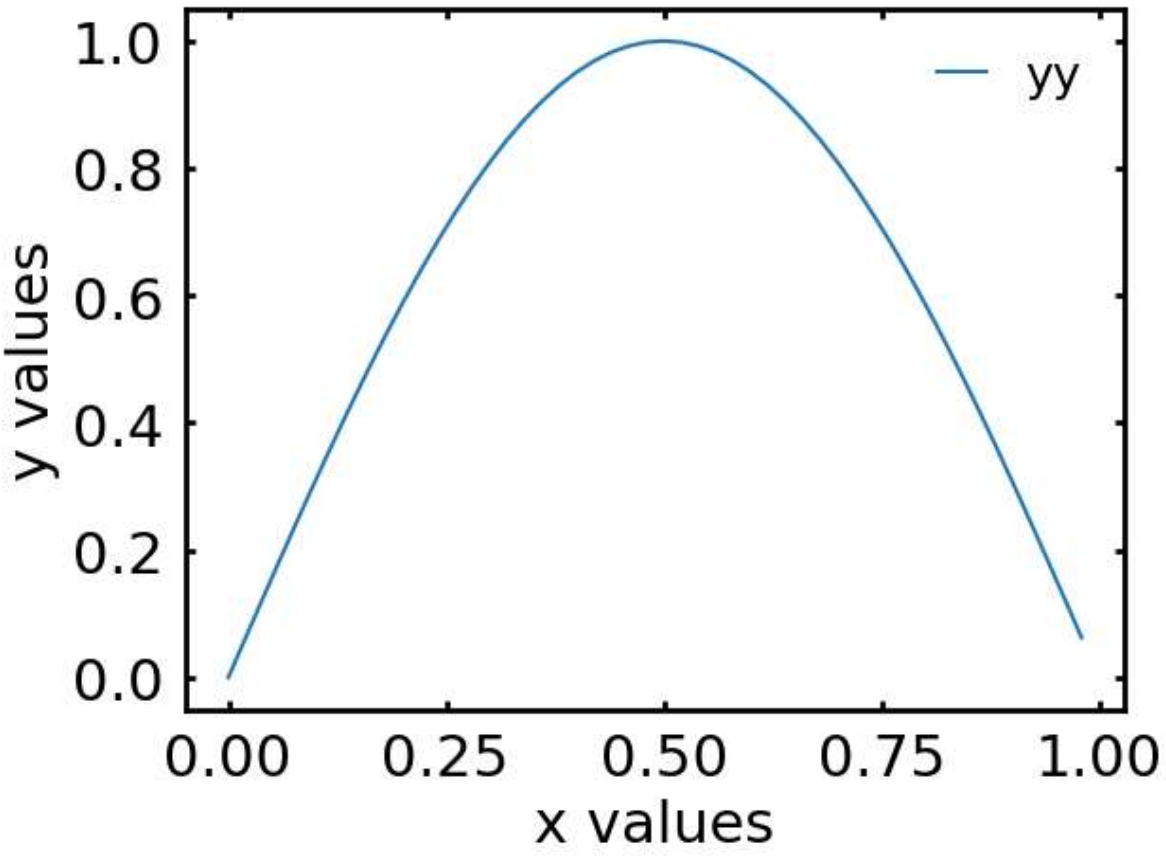
Out[38]:
```
<Axes: xlabel='xx'>
```



In [39]:
```
# Dataframe can be used in the graph drawing
fig4=plt.figure() # Preparing the canvas named as 'fig4'
ax=fig4.add_subplot(111) # Preparing the axes named as 'ax' in 'fig4'
ax.plot(df, label='data') # Plotting all the data in dataframe
ax.set_xlabel('x values') # Title of the axis can be shown
ax.set_ylabel('y values')
ax.legend()
plt.show()
```



In [40]:
```
# If you want to use legend, this is the sample script
fig4=plt.figure() # Preparing the canvas named as 'fig4'
ax=fig4.add_subplot(111) # Preparing the axes named as 'ax' in 'fig4'
for f in df.columns:
    ax.plot(df[f], label=f) # Plotting all the data in dataframe
ax.set_xlabel('x values') # Title of the axis can be shown
ax.set_ylabel('y values')
ax.legend()
plt.show()
```



Using the OS functions for selecting data files

In [41]:
```
pwd # Show the present working folder
```

Out[41]:
```
'C:¥¥Users¥¥tadai¥¥Dropbox¥¥Work_data¥¥Python¥¥Tutrials_by_TI¥¥New_tutrial_for_data_handling'
```

In [42]:
```
df.to_csv('temp.csv') # Saving the dataframe 'df' as a csv-formatted file, 'temp.csv
```

In [43]:
```
flst=os.listdir() # filelist of the present working folder
flst
```

Out[43]:
```
['.ipynb_checkpoints',
 'mystyle.mplstyle',
 'Search_folder_name_matplotlib.ipynb',
 'temp.csv',
 'temp_sin.svg',
 'Tutrial_for_data_handling_20211220.ipynb',
 'Tutrial_for_data_handling_20211220.pdf',
 'Tutrial_for_data_handling_20230420.ipynb',
 'Tutrial_for_KKT_20220324.pdf',
 'Tutrial_for_making_graph_20230511.ipynb']
```

In [44]:
```
path='./' # present folder
flst2=os.listdir(path) # filelist of the folder specfied by 'path'
flst2
```

Out[44]:
```
['.ipynb_checkpoints',
 'mystyle.mplstyle',
 'Search_folder_name_matplotlib.ipynb',
 'temp.csv',
 'temp_sin.svg',
 'Tutrial_for_data_handling_20211220.ipynb',
 'Tutrial_for_data_handling_20211220.pdf',
 'Tutrial_for_data_handling_20230420.ipynb',
 'Tutrial_for_KKT_20220324.pdf',
 'Tutrial_for_making_graph_20230511.ipynb']
```

In [45]:
```
flst1=[f for f in flst if '.csv' in f] # Selection of the file names which includir
flst1
```

Out[45]:
```
['temp.csv']
```

Loading the data file

In [46]:
```
df2=pd.read_csv(flst1[0])
df2.head()
```

Out[46]:

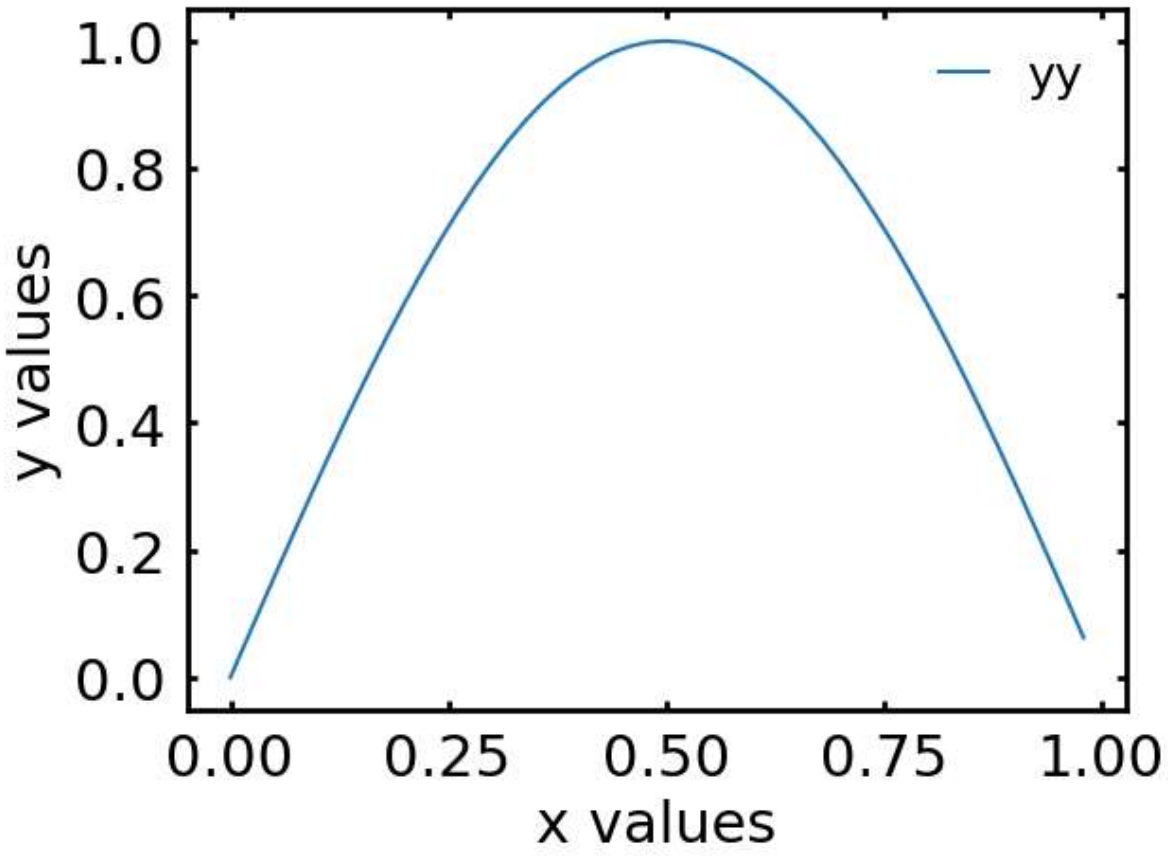
	xx	yy
0	0.00	0.000000
1	0.02	0.062791
2	0.04	0.125333
3	0.06	0.187381
4	0.08	0.248690

In [47]:
```
df2=pd.read_csv(flst1[0], index_col=0)
df2.head()
```

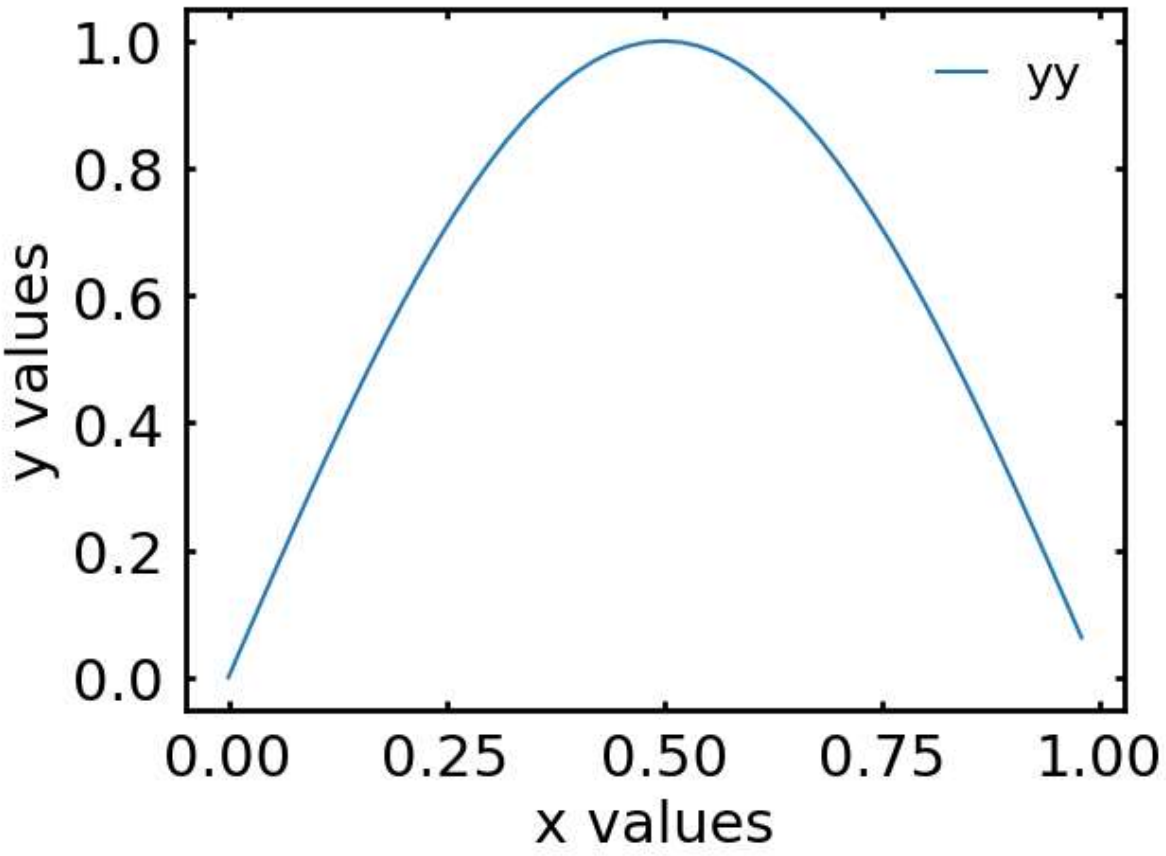
Out[47]:

	yy
xx	
0.00	0.000000
0.02	0.062791
0.04	0.125333
0.06	0.187381
0.08	0.248690

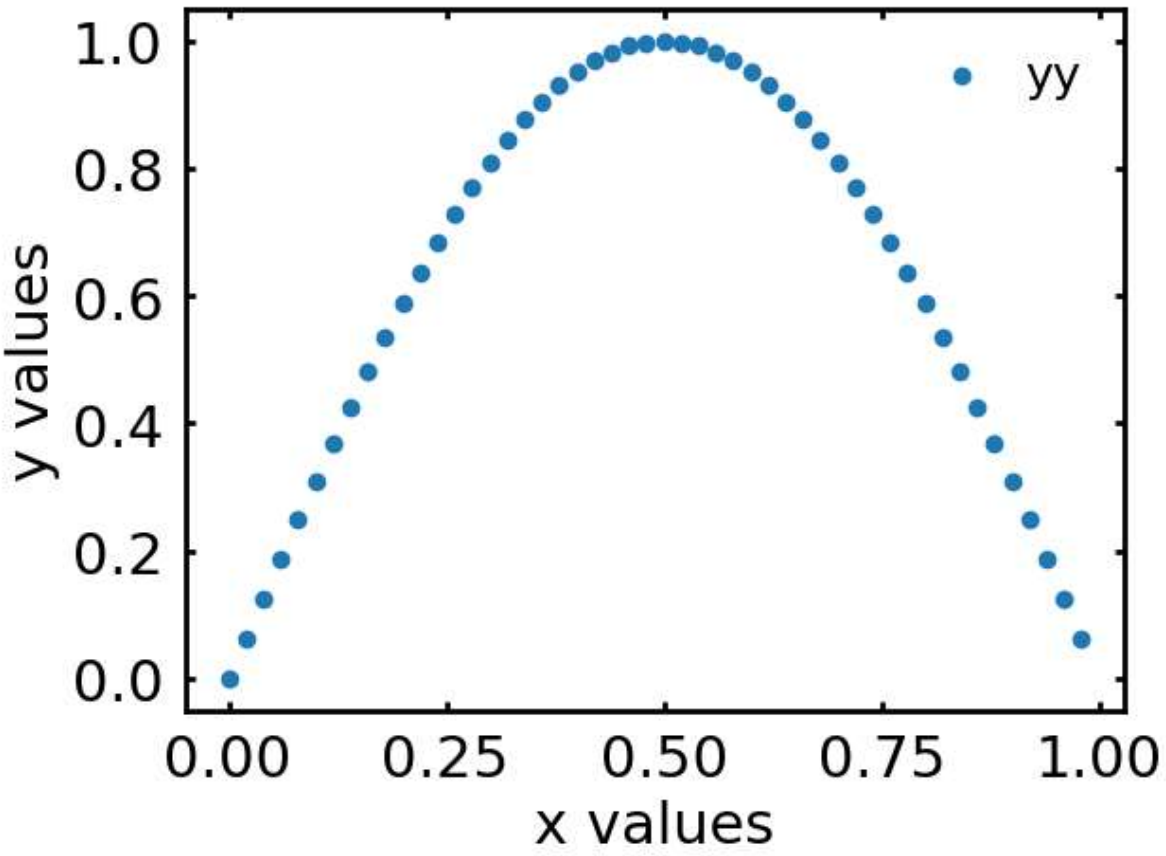
In [48]:
```
# Using dataframe, you can make a graph
fig5=plt.figure() # Preparing the canvas named as 'fig5'
ax=fig5.add_subplot(111) # Preparing the axes named as 'ax' in 'fig5'
for f in df2.columns:
    ax.plot(df2[f], label=f) # Plotting all the data in dataframe
ax.set_xlabel('x values') # Title of the axis can be shown
ax.set_ylabel('y values')
ax.legend()
plt.show()
```



In [49]:
```
# Another style
fig5=plt.figure() # Preparing the canvas named as 'fig5'
ax=fig5.add_subplot(111) # Preparing the axes named as 'ax' in 'fig5'
for f in df2.columns:
    ax.plot(df2.index, df2[f], label=f) # Plotting all the data in dataframe
ax.set_xlabel('x values') # Title of the axis can be shown
ax.set_ylabel('y values')
ax.legend()
plt.show()
```



In [50]:
```
# Scatter plot
fig5=plt.figure() # Preparing the canvas named as 'fig5'
ax=fig5.add_subplot(111) # Preparing the axes named as 'ax' in 'fig5'
for f in df2.columns:
    ax.scatter(df2.index, df2[f], label=f) # Plotting all the data in dataframe
ax.set_xlabel('x values') # Title of the axis can be shown
ax.set_ylabel('y values')
ax.legend()
plt.show()
```



In [ ]: