Solved Examples

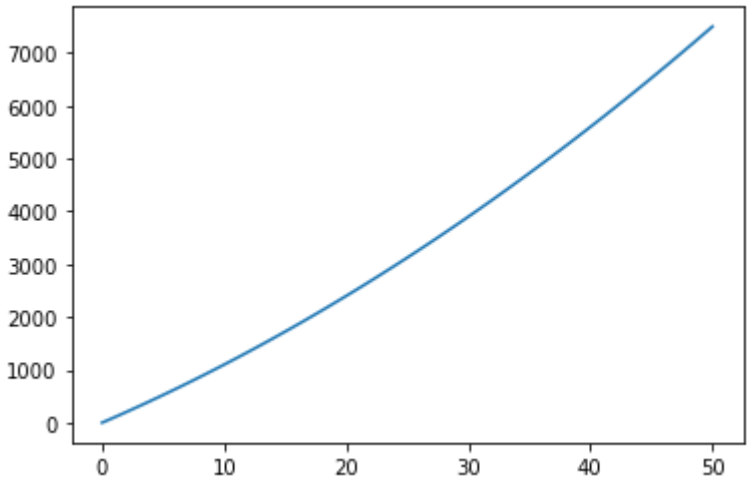
In [1]:
```
import matplotlib
```

In [2]:
```
import matplotlib as mpl
```

In [3]:
```
import matplotlib.pyplot as plt
```

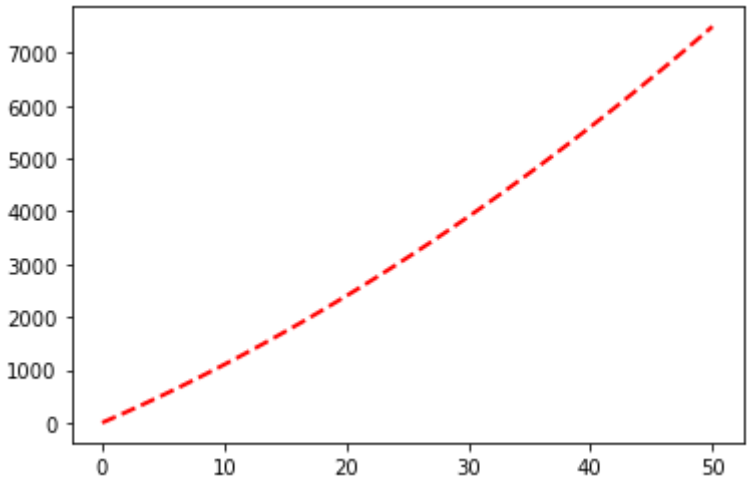
In [4]:
```
# import the NumPy package
import numpy as np
# generate random number using NumPy, generate two sets of random numbers a
x = np.linspace(0,50,100)
y = x * np.linspace(100,150,100)
# Create a basic plot
plt.plot(x,y)
```

Out[4]:
```
[<matplotlib.lines.Line2D at 0x29cdb9e82b0>]
```



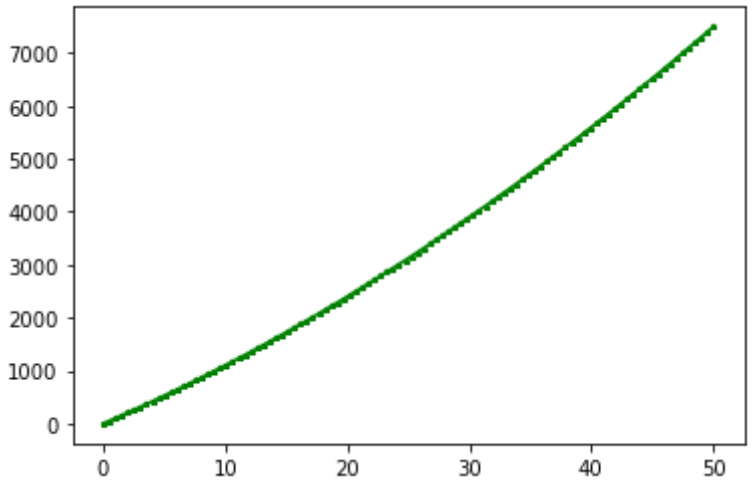
In [5]:
```
plt.plot(x, y, c = 'r', linestyle = '--', linewidth=2)
```

Out[5]:
```
[<matplotlib.lines.Line2D at 0x29cdbae85e0>]
```



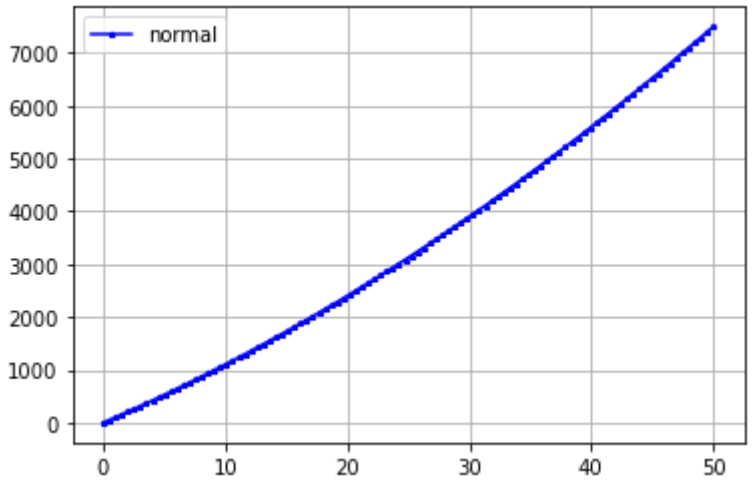
In [6]:
```
plt.plot (x, y, marker='*', c='g', markersize=3)
```

Out[6]:
```
[<matplotlib.lines.Line2D at 0x29cdbb5d580>]
```



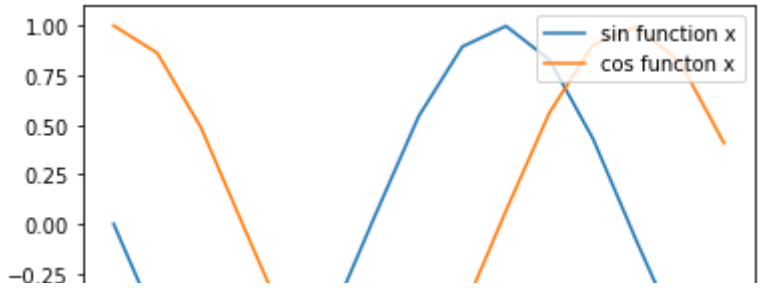
In [7]:
```
plt.plot (x, y, marker='*', markersize=3, c='b', label='normal')
plt.grid(True)
# add legend and label
plt.legend()
```

Out[7]:
```
<matplotlib.legend.Legend at 0x29cdbbbf910>
```



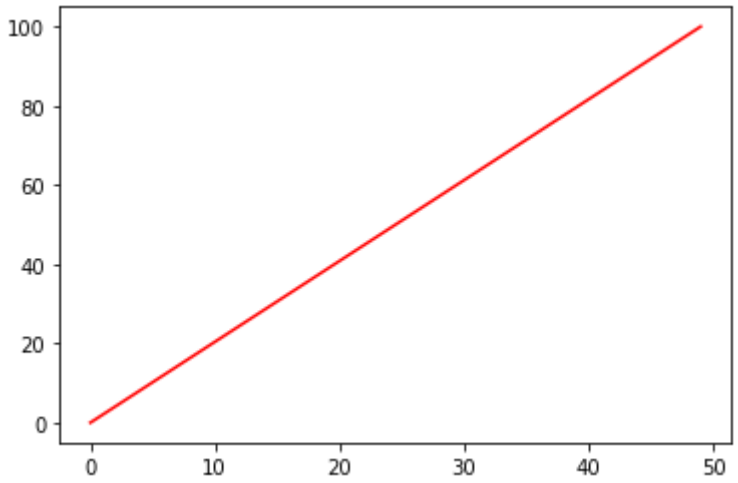
In [8]:
```
# lets plot two lines Sin(x) and Cos(x)
# loc is used to set the location of the legend on the plot
# label is used to represent the label for the line in the legend
# generate the random number
x= np.arange(0,1500,100)
plt.plot(np.sin(x),label='sin function x')
plt.plot(np.cos(x),label='cos functon x')
plt.legend(loc='upper right')
```

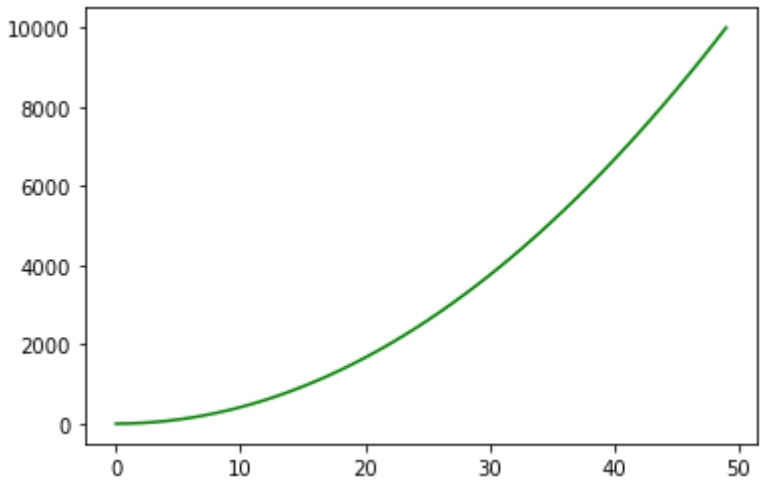
Out[8]:
```
<matplotlib.legend.Legend at 0x29cdbc3cc70>
```



In [9]:

```
x= np.linspace(0,100,50)
plt.plot(x,'r',label='simple x')
plt.show()
plt.plot(x*x,'g',label='two times x')
plt.show()
plt.legend(loc='upper right')
```






```
No artists with labels found to put in legend.  Note that artists whose lab
el start with an underscore are ignored when legend() is called with no arg
ument.
```

Out[9]:

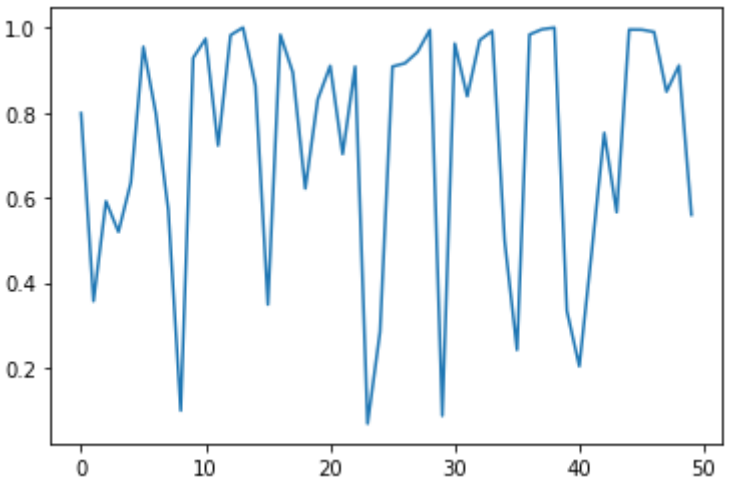
```
<matplotlib.legend.Legend at 0x29cdbd7d4c0>
```

In [10]:

```
# subplots are used to create multiple plots in a single figure
# let's create a single subplot first following by adding more subplots
x = np.random.rand(50)
y = np.sin(x*2)
#need to create an empty figure with an axis as below, figure and axis are
fig, ax = plt.subplots()
#add the charts to the plot
ax.plot(y)
```

Out[10]:

```
[<matplotlib.lines.Line2D at 0x29cdbdef3a0>]
```



In [11]:

```
# Let's add multiple plots using subplots() function
# Give the required number of plots as an argument in subplots(), below fur
fig, axs = plt.subplots(2)
#create data
x=np.linspace(0,100,10)
# assign the data to the plot using axs
axs[0].plot(x, np.sin(x**2))
axs[1].plot(x, np.cos(x**2))
# add a title to the subplot figure
fig.suptitle('Vertically stacked subplots')
```

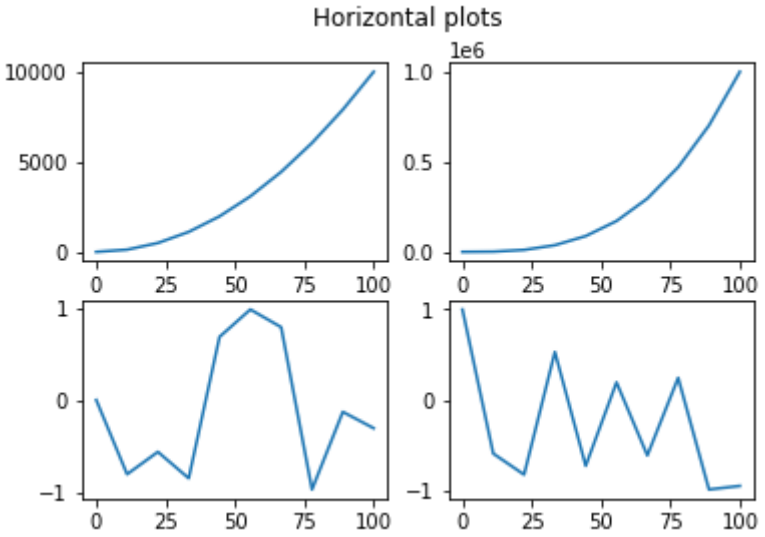
Out[11]:

```
Text(0.5, 0.98, 'Vertically stacked subplots')
```

Vertically stacked subplots

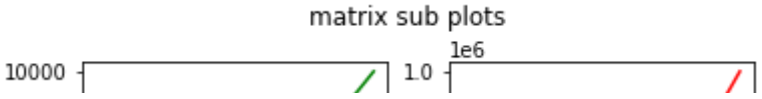
In [12]:
```
# Create horizontal subplots
# Give two arguments rows and columns in the subplot() function
# subplot() gives two dimensional array with 2*2 matrix
# need to provide ax also similar 2*2 matrix as below
fig, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2, 2)
# add the data to the plots
ax1.plot(x, x**2)
ax2.plot(x, x**3)
ax3.plot(x, np.sin(x**2))
ax4.plot(x, np.cos(x**2))
# add title
fig.suptitle('Horizontal plots')
```

Out[12]:
```
Text(0.5, 0.98, 'Horizontal plots')
```



In [13]:
```
# another simple way of creating multiple subplots as below, using axs
fig, axs = plt.subplots(2, 2)
# add the data referring to row and column
axs[0,0].plot(x, x**2,'g')
axs[0,1].plot(x, x**3,'r')
axs[1,0].plot(x, np.sin(x**2),'b')
axs[1,1].plot(x, np.cos(x**2),'k')
# add title
fig.suptitle('matrix sub plots')
```

Out[13]:
```
Text(0.5, 0.98, 'matrix sub plots')
```

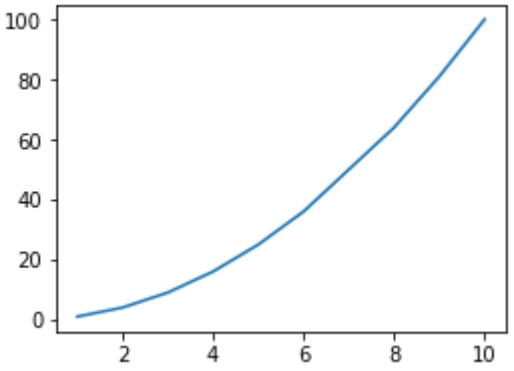


In [14]:

```
# let's create a figure object
# change the size of the figure is 'figsize = (a,b)' a is width and 'b' is
# create a figure object and name it as fig
fig = plt.figure(figsize=(4,3))
# create a sample data
X = np.array([1,2,3,4,5,6,8,9,10])
Y = X**2
# plot the figure
plt.plot(X,Y)
```

Out[14]:

```
[<matplotlib.lines.Line2D at 0x29cdbd18cd0>]
```

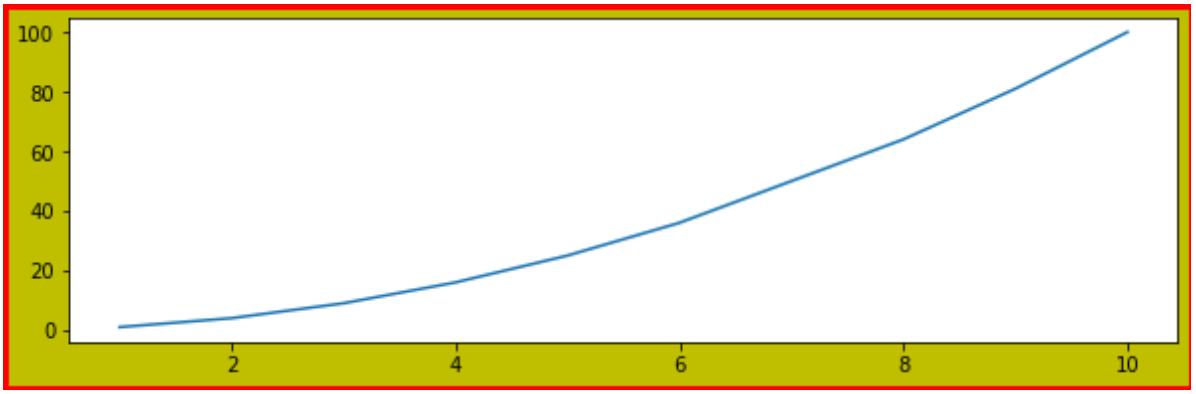


In [15]:

```
# let's change the figure size and also add additional parameters like face
fig = plt.figure(figsize=(10,3),facecolor='y',edgecolor='r',linewidth=5)
# create a sample data
X = np.array([1,2,3,4,5,6,8,9,10])
Y = X**2
# plot the figure
plt.plot(X,Y)
```

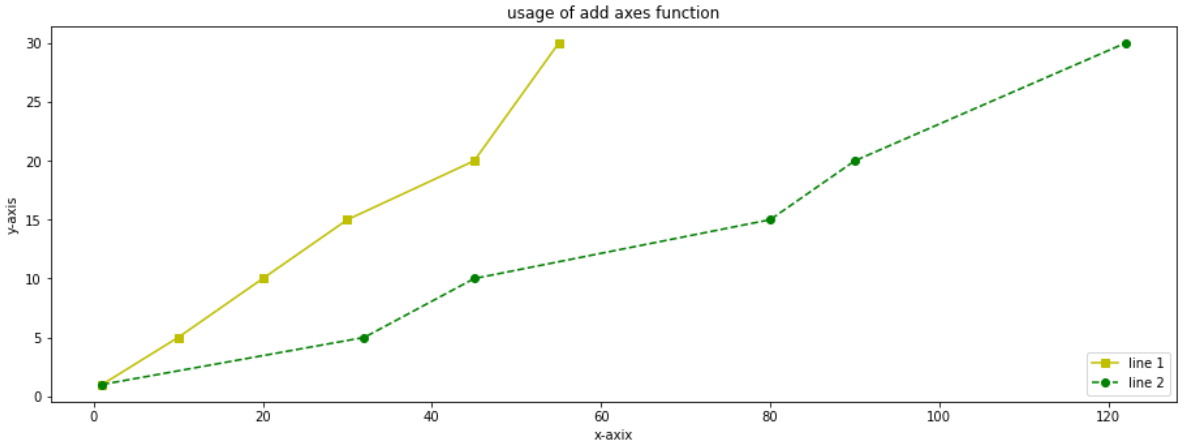
Out[15]:

```
[<matplotlib.lines.Line2D at 0x29cdbcdc730>]
```



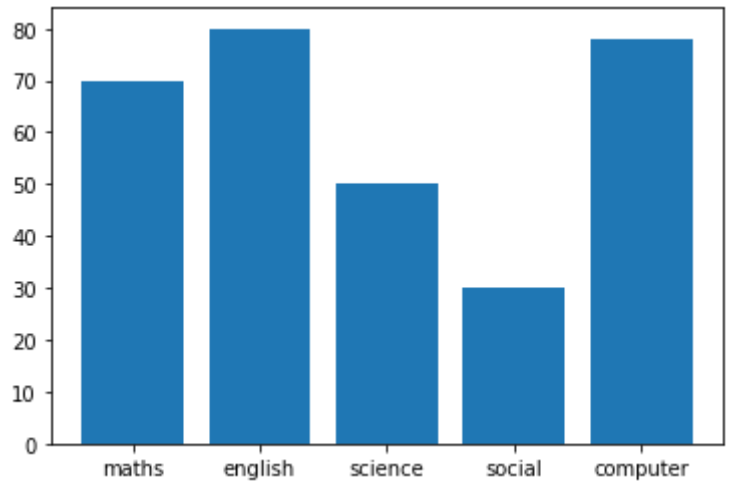
In [16]:

```
# lets add axes using add_axes() method
# create a sample data
y = [1, 5, 10, 15, 20,30]
x1 = [1, 10, 20, 30, 45, 55]
x2 = [1, 32, 45, 80, 90, 122]
# create the figure
fig = plt.figure()
# add the axes
ax = fig.add_axes([0,0,2,1])
l1 = ax.plot(x1,y,'ys-')
l2 = ax.plot(x2,y,'go--')
# add additional parameters
ax.legend(labels = ('line 1', 'line 2'), loc = 'lower right')
ax.set_title("usage of add axes function")
ax.set_xlabel('x-axix')
ax.set_ylabel('y-axis')
plt.show()
```



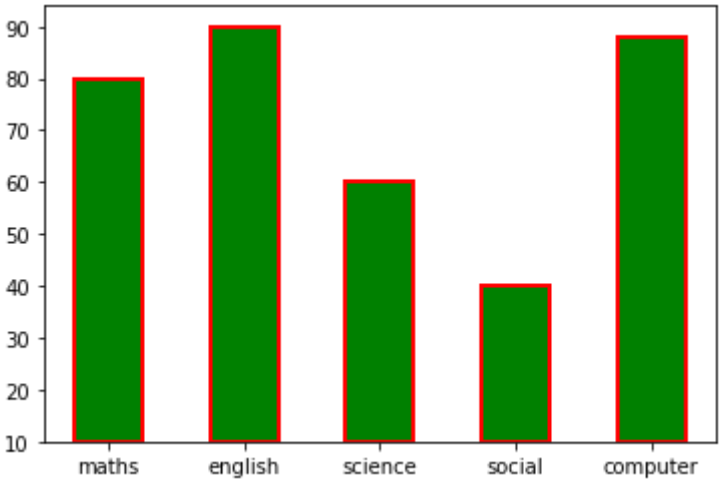
In [17]:

```
# lets create a simple bar chart
#x-axis is shows the subject and y -axis shows the markers in each subject
subject = ['maths','english','science','social','computer']
marks =[70,80,50,30,78]
plt.bar(subject,marks)
plt.show()
```



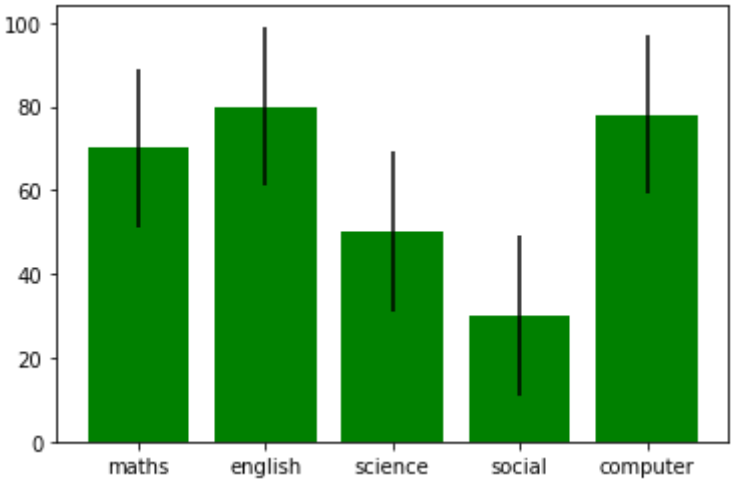
In [18]:
```
#let's do some customizations
#width - shows the bar width and default value is 0.8
#color - shows the bar color
#bottom - value from where the y - axis starts in the chart i.e., the lowes
#align -move the position of x-label, has two options 'edge' or 'center'
#edgecolor - used to color the borders of the bar
#linewidth - used to adjust the width of the line around the bar
#tick_label - to set the customized labels for the x-axis
plt.bar(subject,marks,color='g',width=0.5,bottom=10,align='center',edgecolo
tick_label=subject)
```

Out[18]:
```
<BarContainer object of 5 artists>
```



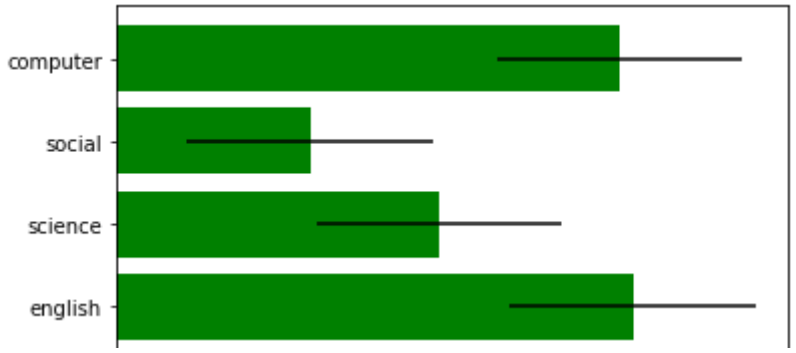
In [19]:
```
# errors bars could be added to represent the error values referring to an
# here in this example we used standard deviation to show as error bars
plt.bar(subject,marks,color ='g',yerr=np.std(marks))
```

Out[19]:
```
<BarContainer object of 5 artists>
```



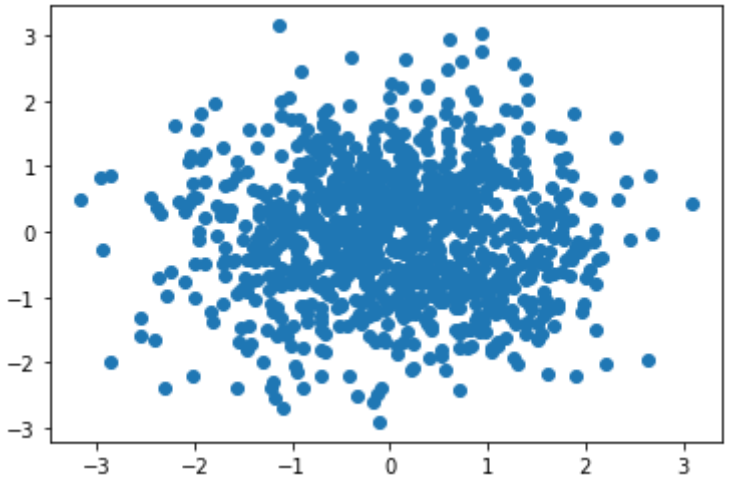
In [20]:
```
# to plot horizontal bar plot use plt.barh() function
plt.barh(subject,marks,color ='g',xerr=np.std(marks))
```

Out[20]:
```
<BarContainer object of 5 artists>
```



In [21]:
```
x = np.random.randn(1000)
y = np.random.randn(1000)
plt.scatter(x,y)
```

Out[21]:
```
<matplotlib.collections.PathCollection at 0x29cdd4de820>
```



In [22]:
```
# as you observe there is no correlation exists between x and y
# let's try to add additional parameters
# size - to manage the size of the points
#color - to set the color of the points
#marker - type of marker
#alpha - transparency of point
size = 150*np.random.randn(1000)
colors = 100*np.random.randn(1000)
plt.scatter(x, y, s=size, c = colors, marker ='*', alpha=0.7)
```

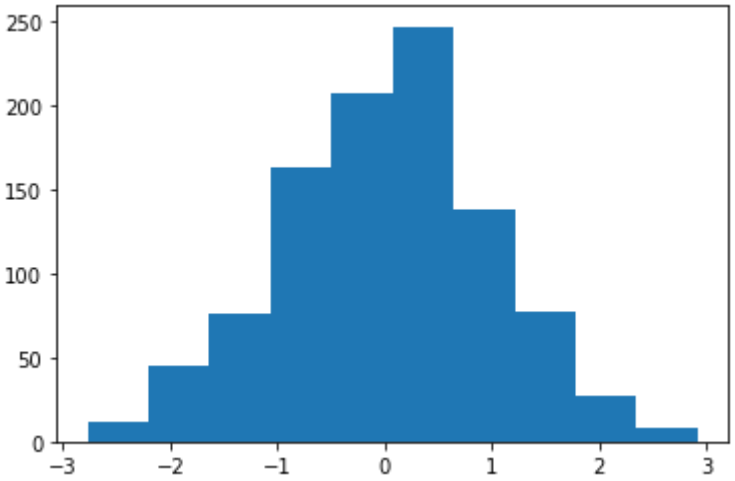
Out[22]:
```
E:\anoconda\lib\site-packages\matplotlib\collections.py:980: RuntimeWarnin
g: invalid value encountered in sqrt
  scale = np.sqrt(self._sizes) * dpi / 72.0 * self._factor
<matplotlib.collections.PathCollection at 0x29cdd54eac0>
```

In [23]:

```
# let's generate random numbers and use the random numbers to generate hist
data = np.random.randn(1000)
plt.hist(data)
```

Out[23]:

```
(array([ 12.,  45.,  76., 163., 207., 247., 138.,  77.,  27.,   8.]),
 array([-2.77321913, -2.20429454, -1.63536995, -1.06644536, -0.49752077,
         0.07140382,  0.64032841,  1.20925299,  1.77817758,  2.34710217,
         2.91602676]),
 <BarContainer object of 10 artists>)
```



In [24]:

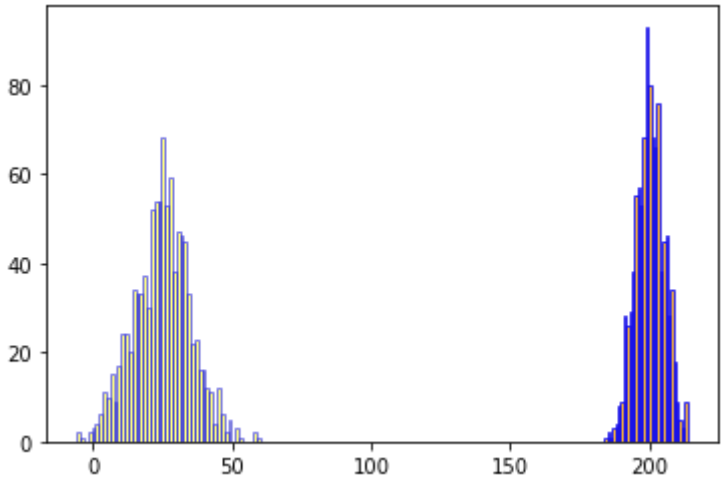
```
# let's add additional parameters
# facecolor
# alpha
# edgecolor
# bins
data = np.random.randn(1000)
plt.hist(data, facecolor ='y',linewidth=2,edgecolor='k', bins=30, alpha=0.6
```

Out[24]:

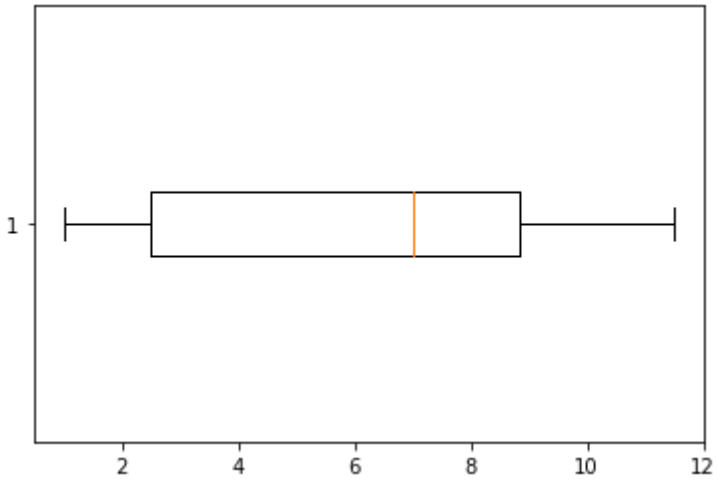
```
(array([ 4.,  1.,  7.,  6.,  6., 15., 17., 27., 35., 39., 48., 62., 60.,
        68., 56., 85., 82., 72., 72., 42., 45., 35., 33., 28., 14., 14.,
         9.,  7.,  8.,  3.]),
 array([-2.87063823, -2.68335529, -2.49607234, -2.3087894 , -2.12150646,
        -1.93422351, -1.74694057, -1.55965762, -1.37237468, -1.18509174,
        -0.99780879, -0.81052585, -0.6232429 , -0.43595996, -0.24867702,
        -0.06139407,  0.12588887,  0.31317181,  0.50045476,  0.6877377 ,
         0.87502065,  1.06230359,  1.24958653,  1.43686948,  1.62415242,
         1.81143537,  1.99871831,  2.18600125,  2.3732842 ,  2.56056714,
         2.74785009]),
 <BarContainer object of 30 artists>)
```

In [25]:
```
# lets create multiple histograms in a single plot
# Create random data
hist1 = np.random.normal(25,10,1000)
hist2 = np.random.normal(200,5,1000)
#plot the histogram
plt.hist(hist1,facecolor = 'yellow',alpha = 0.5, edgecolor ='b',bins=50)
plt.hist(hist2,facecolor = 'orange',alpha = 0.8, edgecolor ='b',bins=30)
```

Out[25]:
```
(array([ 1.,  2.,  1.,  3.,  4.,  8.,  9., 28., 26., 29., 38., 55., 57.,
        53., 68., 93., 80., 68., 66., 76., 38., 45., 46., 28., 34., 18.,
         9.,  5.,  3.,  9.]),
 array([183.67673689, 184.66911312, 185.66148935, 186.65386558,
        187.64624182, 188.63861805, 189.63099428, 190.62337051,
        191.61574674, 192.60812298, 193.60049921, 194.59287544,
        195.58525167, 196.5776279 , 197.57000414, 198.56238037,
        199.5547566 , 200.54713283, 201.53950907, 202.5318853 ,
        203.52426153, 204.51663776, 205.50901399, 206.50139023,
        207.49376646, 208.48614269, 209.47851892, 210.47089516,
        211.46327139, 212.45564762, 213.44802385]),
 <BarContainer object of 30 artists>)
```



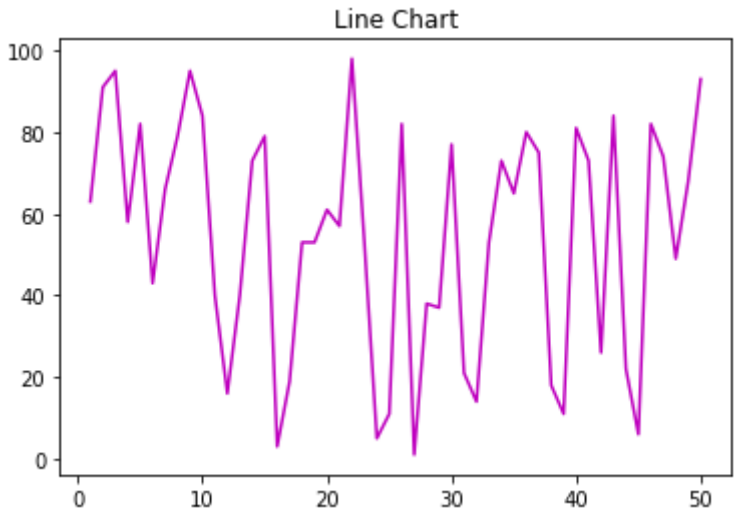
In [26]:
```
import matplotlib.pyplot as plt
data=[1, 1, 2, 2, 4, 6, 6.8, 7.2, 8, 8.3, 9, 10, 10, 11.5]
plt.boxplot(data, vert=False)
plt.show()
```

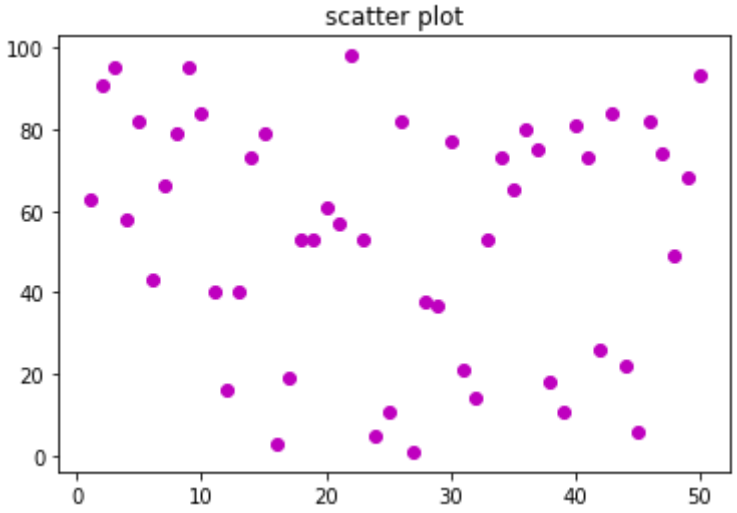


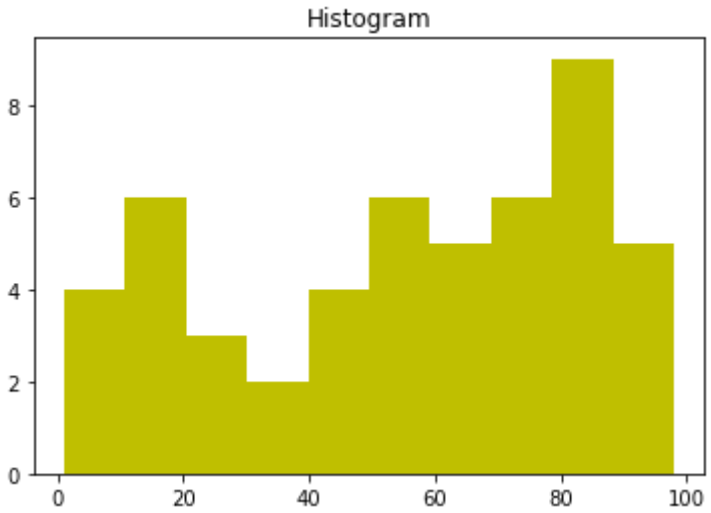
Set A

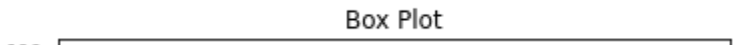
In [27]:

```
#1.Generate a random array of 50 integers and display them using a line cha
#Apply appropriate color, labels and styling options.
import pandas as pd
import numpy as np
from matplotlib.pyplot import*
n=np.random.randint(1,100,50)
x=np.arange(1,51)
y=np.array(n)
title("Line Chart")
plot(x,y,color="m")
show()

title("scatter plot")
scatter(x,y,color="m")
show()

n1=np.random.randint(1,100,50)
#x=np.arange(1,51)
x=np.array(n)
title("Histogram")
hist(x,color="y")
show()

title("Box Plot")
boxplot(n)
show()
```



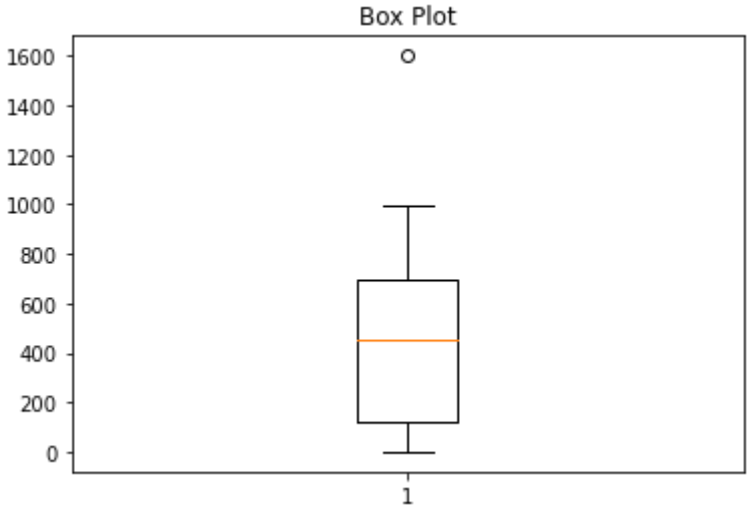






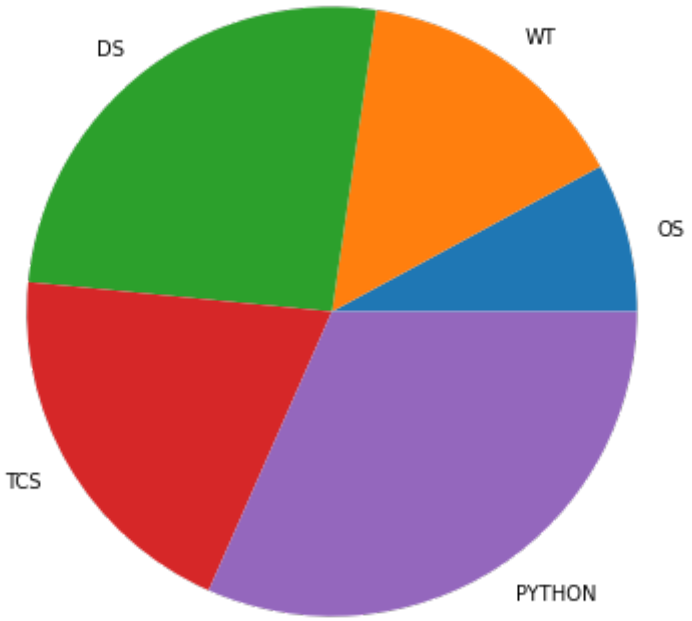
In [28]:

```
#2.Add two outliers to the above data and display the box plot.
import pandas as pd
import numpy as np
from matplotlib.pyplot import*
outliers=[100,458,500,85,555,78,123,457,852,321,114,445,635,887,999,124,456
outliers.append(1600)
outliers.append(1)
b=[outliers]
boxplot(b)
title("Box Plot")
show()
```



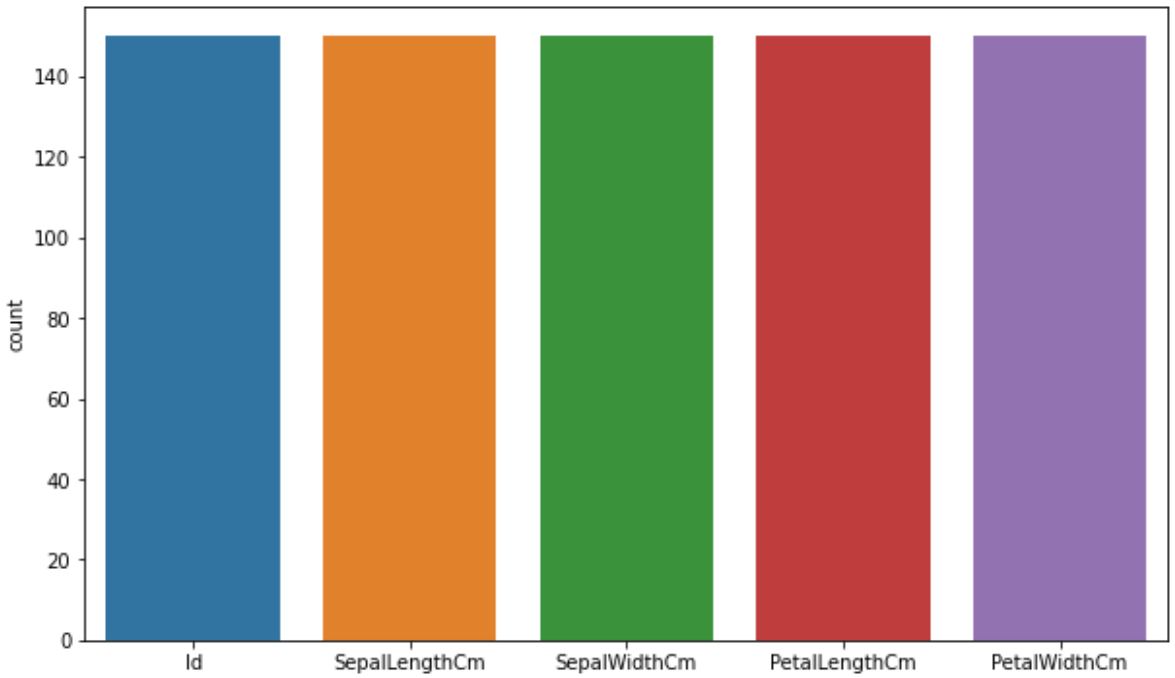
In [29]:

```
#3.Create two lists, one representing subject names and the other represent
#Display the data in a pie chart and bar chart.
# Import libraries
from matplotlib import pyplot as plt
import numpy as np
sub = ['OS', 'WT', 'DS','TCS', 'PYTHON']

data = [23, 43, 75, 58,92]

# Creating plot
fig = plt.figure(figsize =(10, 7))
plt.pie(data, labels = sub)

# show plot
plt.show()
```



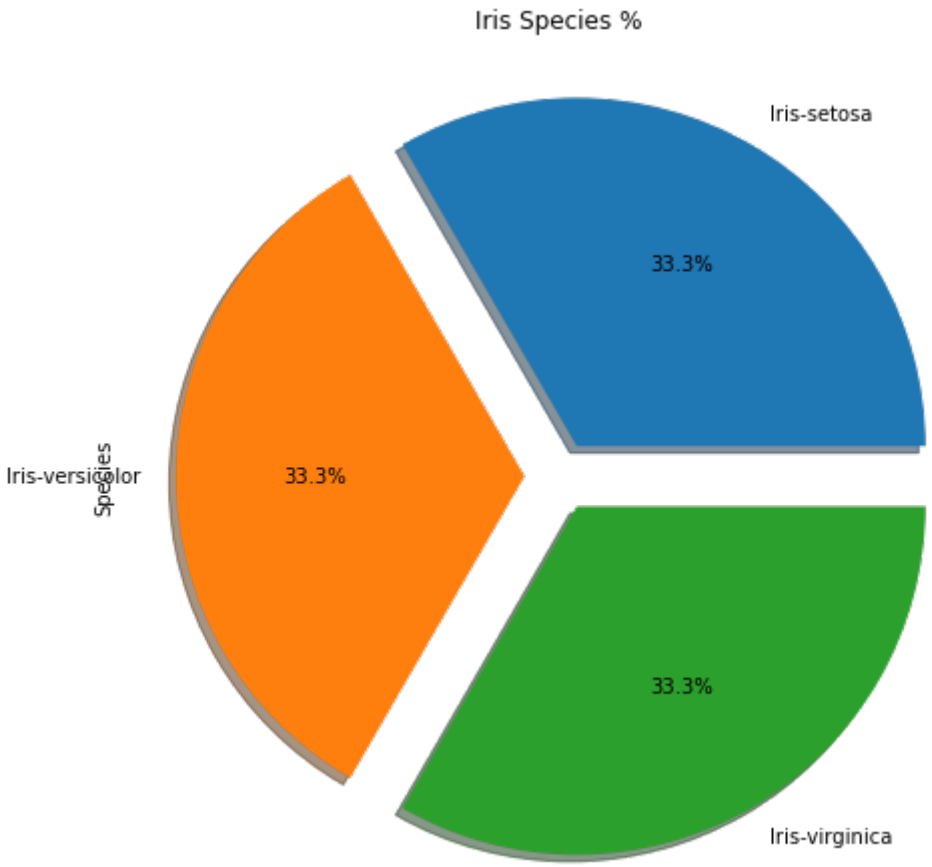
In [30]:

```
#4.Write a Python program to create a Bar plot to get the frequency of the
import pandas as pd
import seaborn as sns
from matplotlib.pyplot import*
df=pd.read_csv("iris.csv")
x=subplots(1,1,figsize=(10,6))
sns.countplot(data=df)
show()
```



In [31]:

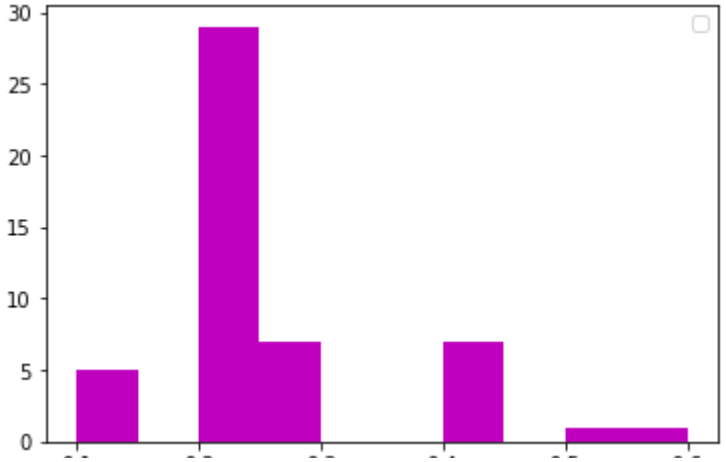
```
#5.Write a Python program to create a Pie plot to get the frequency of the
import pandas as pd
import matplotlib.pyplot as plt
iris = pd.read_csv("iris.csv")
ax=plt.subplots(1,1,figsize=(10,8))
iris['Species'].value_counts().plot.pie(explode=[0.1,0.1,0.1],autopct='%1.1
plt.title("Iris Species %")
plt.show()
```



In [32]:

```
#6.Write a Python program to create a histogram of the three species of the
import pandas as pd
from matplotlib.pyplot import*
from sklearn.datasets import load_iris
iris=load_iris()
print(iris.data[iris.target==1][:5])
print(iris.data[iris.target==1,0][:5])
fig,x=subplots()
x_index=3
colors=['m','y','r']
for lable,color in zip(range(len(iris.target_names)),colors):
    x.hist(iris.data[iris.target==lable,x_index],
                    color=color)
    x.legend(loc='upper right')
    show()
```

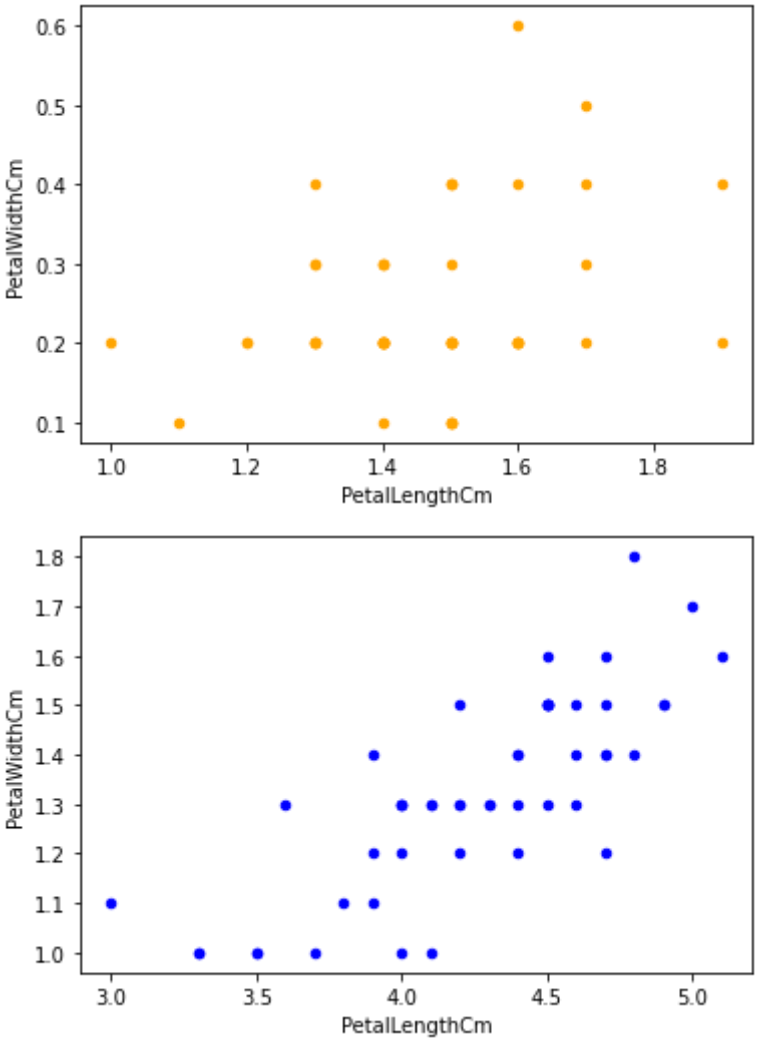
```
No artists with labels found to put in legend.  Note that artists whose lab
el start with an underscore are ignored when legend() is called with no arg
ument.
[[7.  3.2 4.7 1.4]
 [6.4 3.2 4.5 1.5]
 [6.9 3.1 4.9 1.5]
 [5.5 2.3 4.  1.3]
 [6.5 2.8 4.6 1.5]]
[7.  6.4 6.9 5.5 6.5]
```

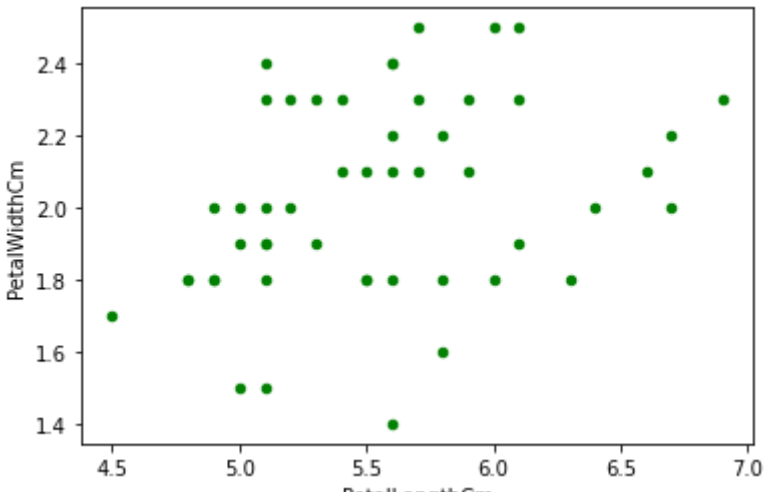


Set B

In [33]:

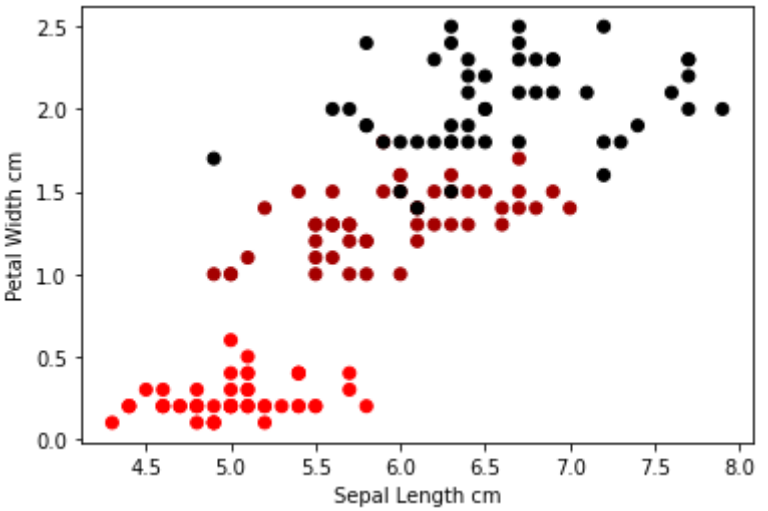
```
#1.Write a Python program to create a graph to find relationship between th
import pandas as pd
import matplotlib.pyplot as plt
iris=pd.read_csv("iris.csv")
iris[iris.Species=='Iris-setosa'].plot.scatter(x='PetalLengthCm',y='PetalWi
iris[iris.Species=='Iris-versicolor'].plot.scatter(x='PetalLengthCm',y='Pet
iris[iris.Species=='Iris-virginica'].plot.scatter(x='PetalLengthCm',y='Peta
fig=plt.gcf()
plt.show()
```





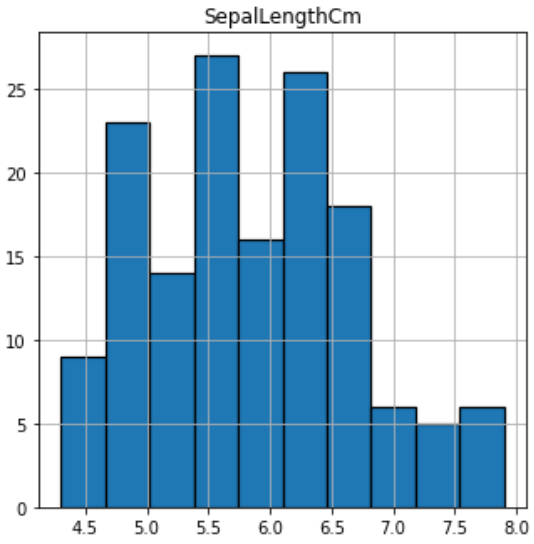
In [34]:

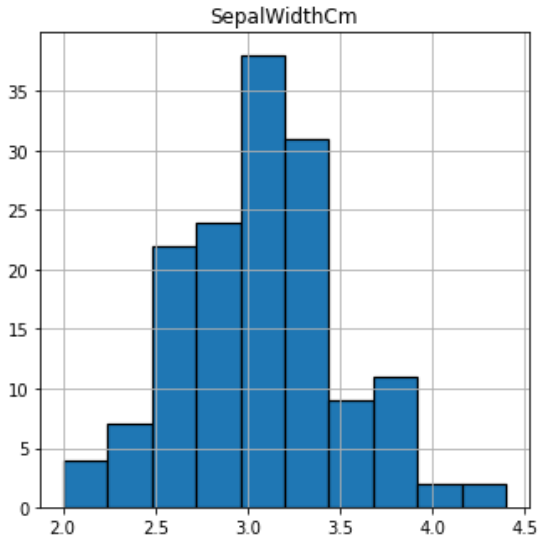
```
#2.Write a Python program to draw scatter plots to compare two features of
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import preprocessing
iris = pd.read_csv("iris.csv")
iris = iris.drop('Id',axis=1)
le = preprocessing.LabelEncoder()
iris.Species = le.fit_transform(iris.Species)
x = iris.iloc[:, :-1].values
y = iris.iloc[:, 4].values
plt.scatter(x[:,0], x[:, 3], c=y, cmap ='flag')
plt.xlabel('Sepal Length cm')
plt.ylabel('Petal Width cm')
plt.show()
```

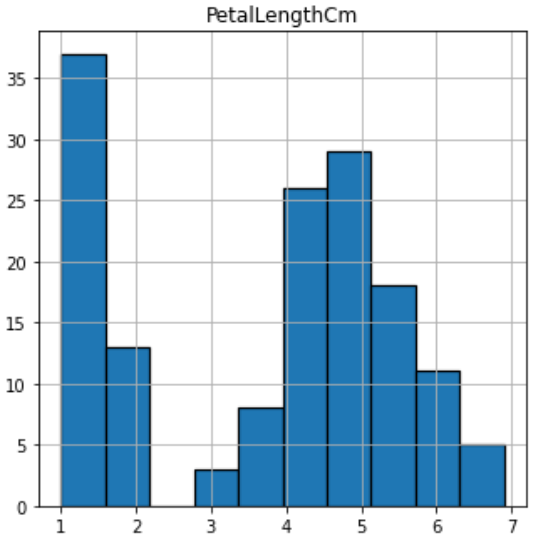


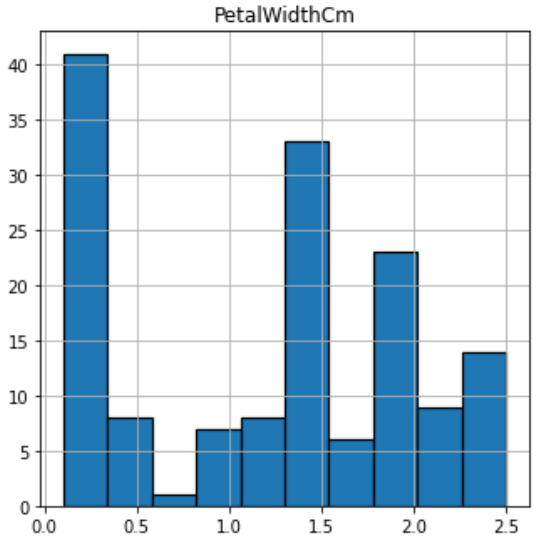
In [35]:

```
#3.Write a Python program to create box plots to see how each feature i.e.
#Petal Width are distributed across the three species
import pandas as pd
import matplotlib.pyplot as plt
iris = pd.read_csv("iris.csv")
# Drop id column
new_data = iris.drop('Id',axis=1)
new_data.hist(edgecolor='black', linewidth=1.2)
fig=plt.gcf()
fig.set_size_inches(12,12)
plt.show()
```





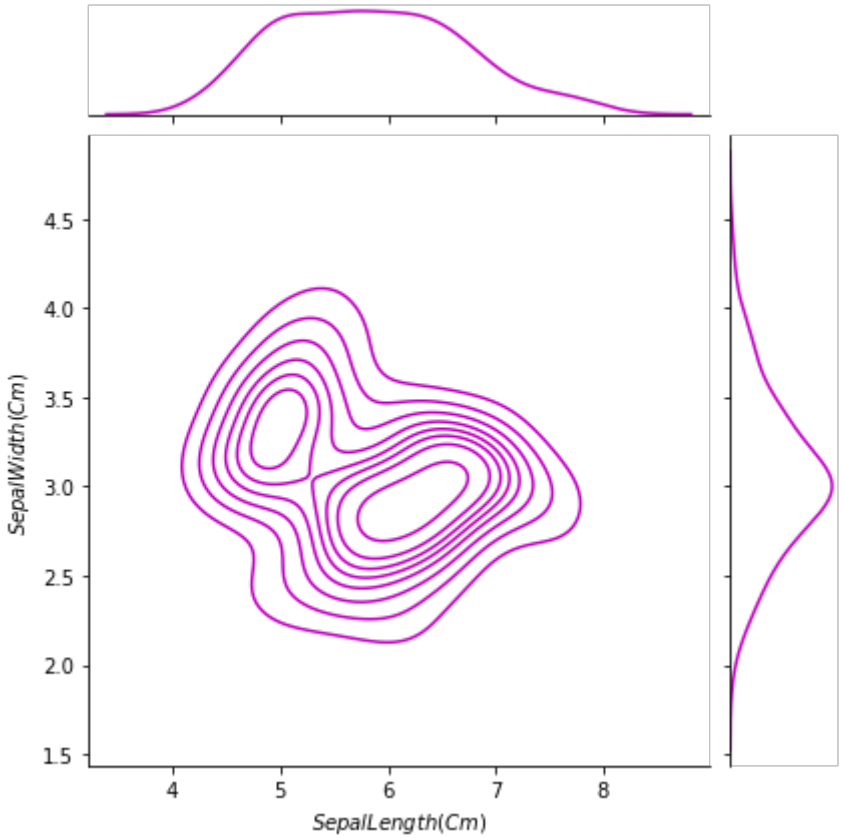




Set c

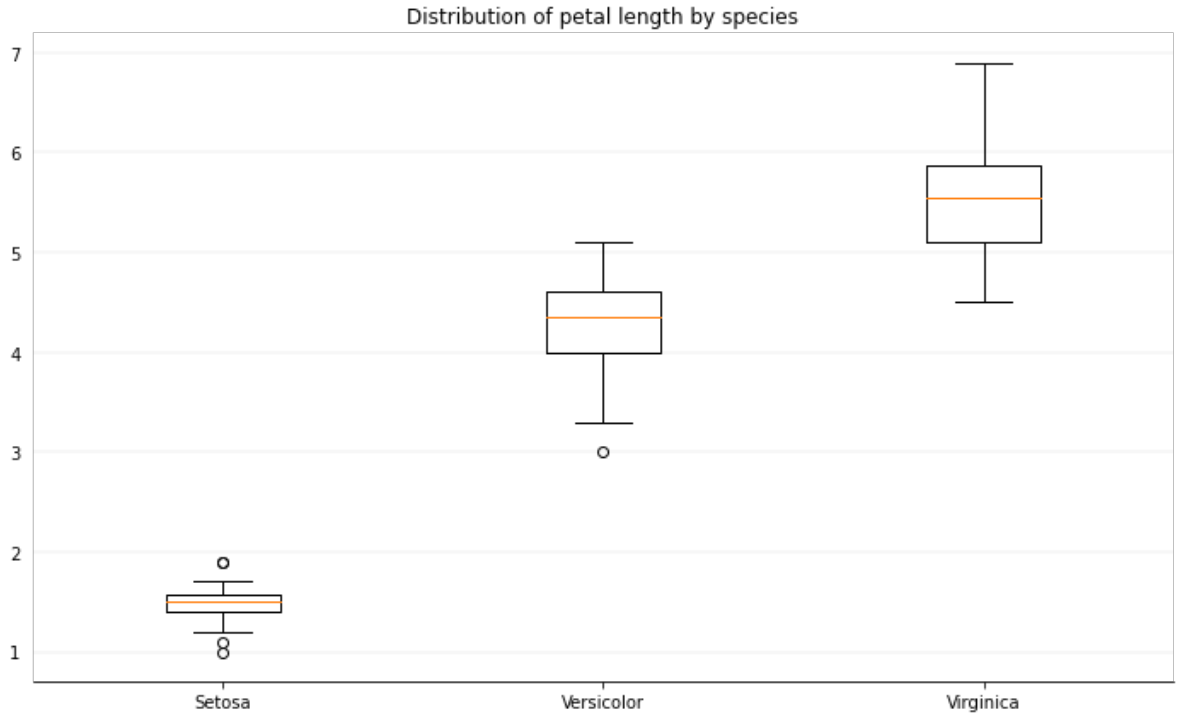
In [36]:

```
#1.Write a Python program to create a pairplot of the iris data set and che
#to be the most separable
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
iris = pd.read_csv("iris.csv")
g = sns.jointplot(x="SepalLengthCm", y="SepalWidthCm", data=iris, kind="kde
g.plot_joint(plt.scatter, c="w", s=40, linewidth=1, marker="+")
g.ax_joint.collections[0].set_alpha(0)
g.set_axis_labels("$SepalLength(Cm)$", "$SepalWidth(Cm)$")
plt.show()
```



In [37]:

```
#2.Write a Python program to generate a box plot to show the Interquartile
#outliers for the three species for each feature.
import numpy as np
import pandas as pd
from sklearn import datasets
import matplotlib.pyplot as plt
iris = datasets.load_iris()
iris_data = pd.DataFrame(data=iris.data, columns=['sepal_length', 'sepal_wi
iris_target = pd.DataFrame(data=iris.target, columns=['species'])
iris_df = pd.concat([iris_data, iris_target], axis=1)
iris_df['species_name'] = np.where(iris_df['species'] == 0, 'Setosa', None)
iris_df['species_name'] = np.where(iris_df['species'] == 1, 'Versicolor', i
iris_df['species_name'] = np.where(iris_df['species'] == 2, 'Virginica', ir
setosa_petal_length = iris_df[iris_df['species_name'] == 'Setosa']['petal_l
versicolor_petal_length = iris_df[iris_df['species_name'] == 'Versicolor'][
virginica_petal_length = iris_df[iris_df['species_name'] == 'Virginica']['p
fig, ax = plt.subplots(figsize=(12, 7))
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.yaxis.set_ticks_position('none')
ax.grid(color='grey', axis='y', linestyle='-', linewidth=0.25, alpha=0.5)
ax.set_title('Distribution of petal length by species')
dataset = [setosa_petal_length, versicolor_petal_length, virginica_petal_le
labels = iris_df['species_name'].unique()
ax.boxplot(dataset, labels=labels)
plt.show()
```



In [38]:

```
#3.Write a Python program to create a join plot using "kde" to describe inc
#between Sepal length and Sepal width. Note: The kernel density estimation
#distribution. In seaborn, this kind of plot is shown with a contour plot a
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
iris = pd.read_csv("iris.csv")
fig=sns.jointplot(x='SepalLengthCm', y='SepalWidthCm', kind="kde", color='c
plt.show()
```