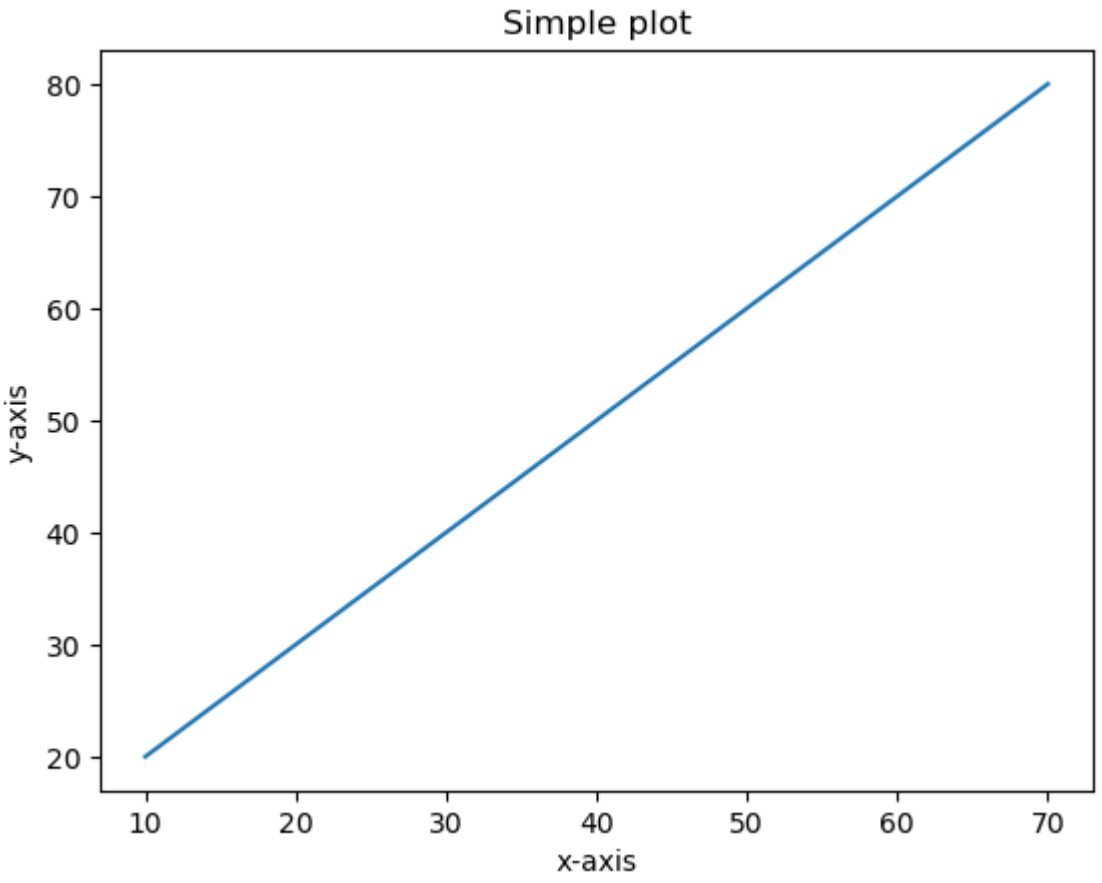
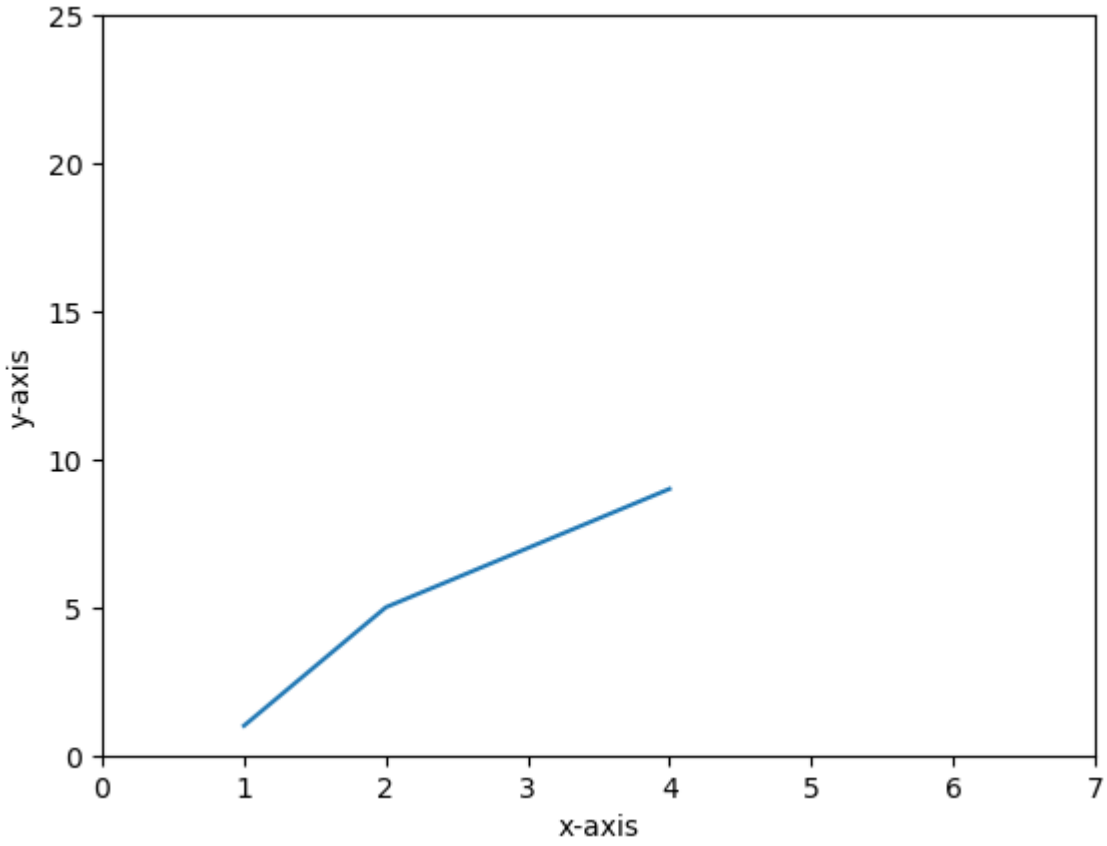
In [2]:
```
# Line Chart
# Creation of basic plot
#  import matplotlib.pyplot as plt
from matplotlib import pyplot as plt

# Data initialization
x = [10,20,30,40,50,60,70]
y = [20,30,40,50,60,70,80]

# ploting the data
plt.plot(x,y)

#Adding title
plt.title('Simple plot')

# # Adding the labels
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.show()
```




In [3]:
```
# Example of coordinates
from matplotlib import pyplot as plt
plt.plot([1,2,3,4], [1,5, 7, 9])
plt.axis([0, 7, 0, 25])
plt.xlabel('x-axis')
plt.ylabel('y-axis')
```

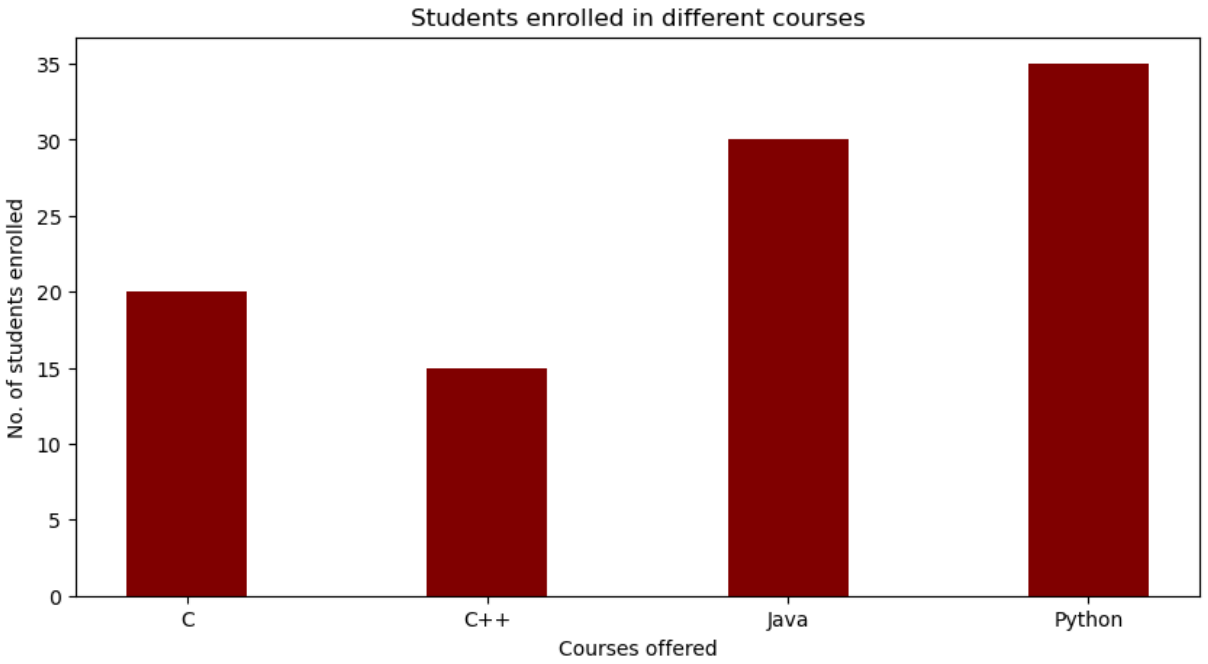
Out[3]: Text(0, 0.5, 'y-axis')




In [ ]:
```
##########################################  Bar plot ###################################
'''
A bar plot or bar chart is a graph that represents the category of data with rectan
heights that is proportional to the values which they represent.
The bar plots can be plotted horizontally or vertically.
A bar chart describes the comparisons between the discrete categories.
One of the axis of the plot represents the specific categories being compared,
while the other axis represents the measured values corresponding to those categori

'''
```

In [5]:
```
# Example bar chart
import numpy as np
import matplotlib.pyplot as plt

  #creating the dataset
data = {'C':20, 'C++':15, 'Java':30,
        'Python':35}
courses = list(data.keys())
values = list(data.values())

fig = plt.figure(figsize = (10, 5))

# creating the bar plot
plt.bar(courses, values, color ='maroon',
        width = 0.4)
```

```
plt.xlabel("Courses offered")
plt.ylabel("No. of students enrolled")
plt.title("Students enrolled in different courses")
plt.show()
```

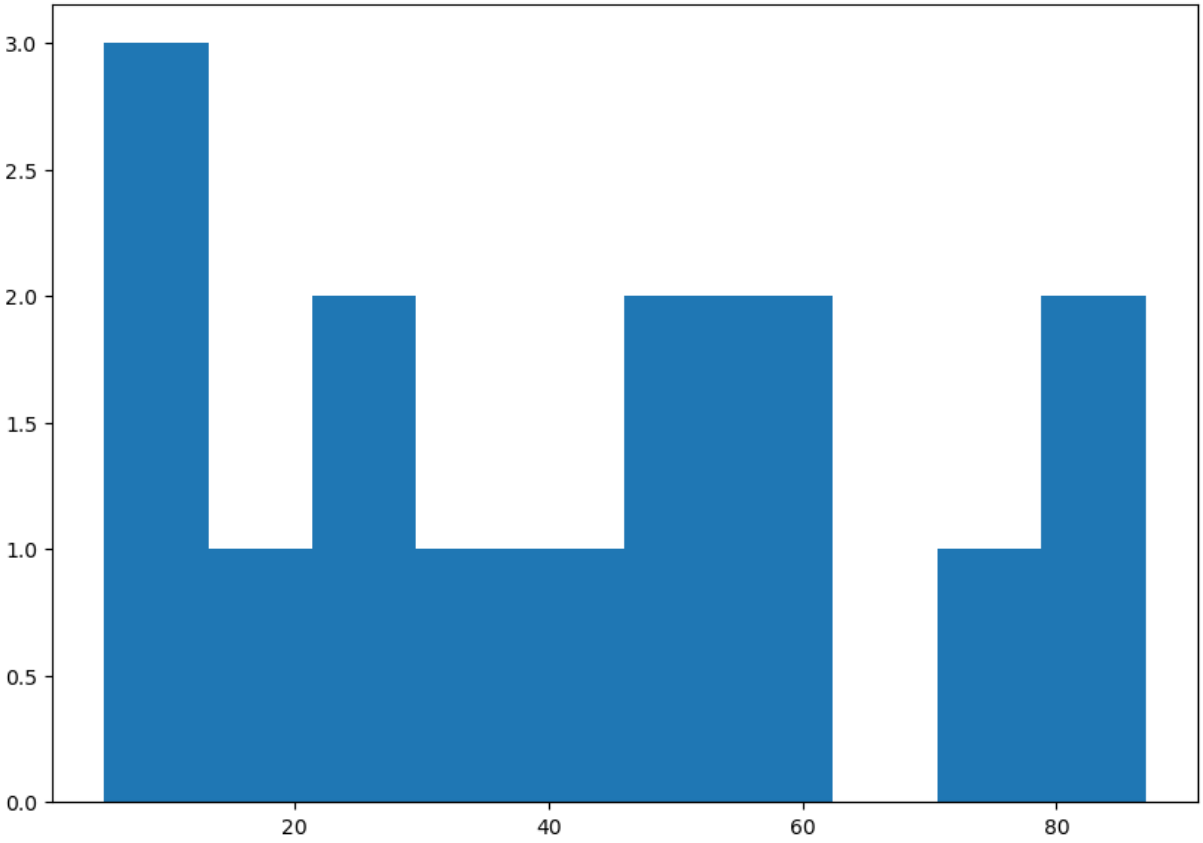



In [ ]:
```
################################        Histogram       ################################
'''

A histogram is basically used to represent data provided in a form of some groups.
It is accurate method for the graphical representation of numerical data distributi
It is a type of bar plot where X-axis represents the bin ranges while Y-axis gives
The following table shows the parameters accepted by matplotlib.pyplot.hist() funct

Attribute       parameter
x       array or sequence of array
bins    optional parameter contains integer or sequence or strings
density optional parameter contains boolean values
range   optional parameter represents upper and lower range of bins
histtype        optional parameter used to create type of histogram [bar, barstacke
align   optional parameter controls the plotting of histogram [left, right, mid]
weights optional parameter contains array of weights having same dimensions as x
bottom  location of the baseline of each bin
rwidth  optional parameter which is relative width of the bars with respect to bin
color   optional parameter used to set color or sequence of color specs
label   optional parameter string or sequence of string to match with multiple data
log     optional parameter used to set histogram axis on log scale

'''
```

In [ ]:
```
```

In [6]:
```
# Histogram... Need to verify values (to be covered in next class)
from matplotlib import pyplot as plt
import numpy as np
```

```
 #Creating dataset
a = np.array([22, 87, 5, 43, 56,
              73, 55, 54, 11,
              20, 51, 5, 79, 31,
              27])

# Creating histogram
fig, ax = plt.subplots(figsize =(10, 7))
ax.hist(a)

# Show plot
plt.show()
```




################## Scatter plot
##########################################################################
Scatter plots are used to observe relationship between variables and uses dots to represent the relationship between them. The scatter() method in the matplotlib library is used to draw a scatter plot. Scatter plots are widely used to represent relation among variables and how change in one affects the other. Syntax The syntax for scatter() method is given below:

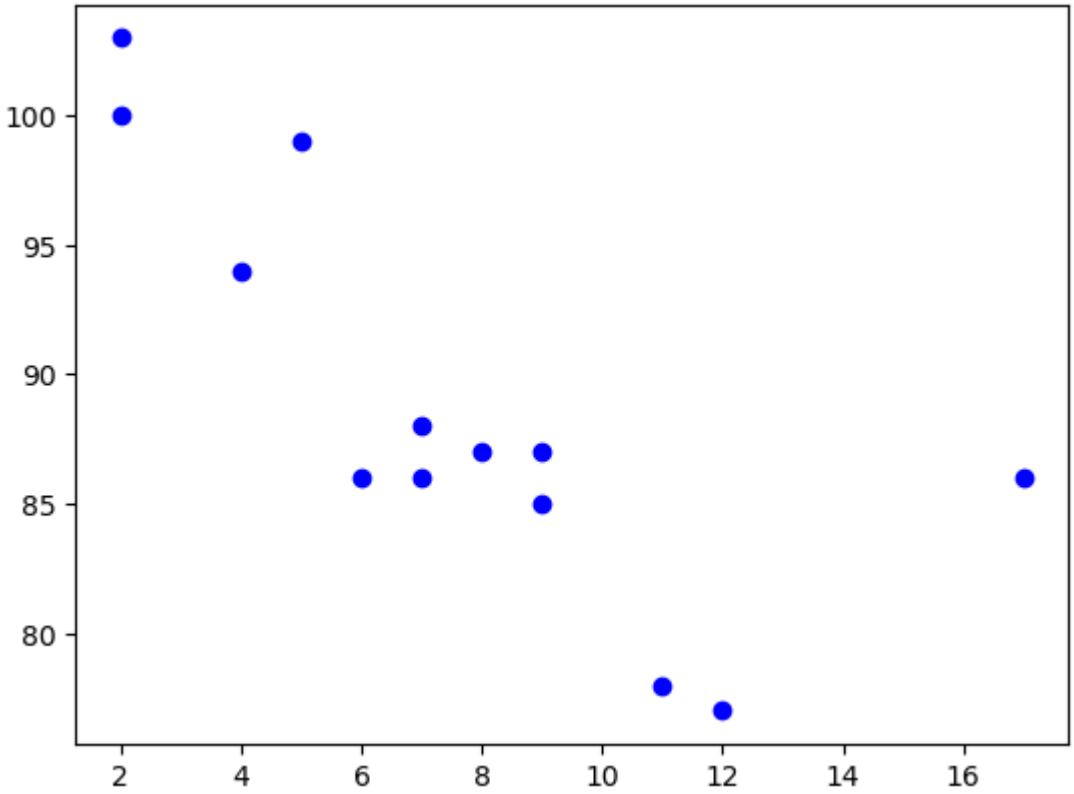
matplotlib.pyplot.scatter(x_axis_data, y_axis_data, s=None, c=None, marker=None, cmap=None, vmin=None, vmax=None, alpha=None, linewidths=None, edgecolors=None)

The scatter() method takes in the following parameters: x_axis_data- An array containing x-axis data y_axis_data- An array containing y-axis data s- marker size (can be scalar or array of size equal to size of x or y) c- color of sequence of colors for markers marker- marker style

cmap- cmap name linewidths- width of marker border edgecolor- marker border color alpha- blending value, between 0 (transparent) and 1 (opaque)

Except x_axis_data and y_axis_data all other parameters are optional and their default value is None.

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


x =[5, 7, 8, 7, 2, 17, 2, 9,4, 11, 12, 9, 6]

y =[99, 86, 87, 88, 100, 86, 103, 87, 94, 78, 77, 85, 86]

plt.scatter(x, y, c ="blue")

# To show the plot
plt.show()
```

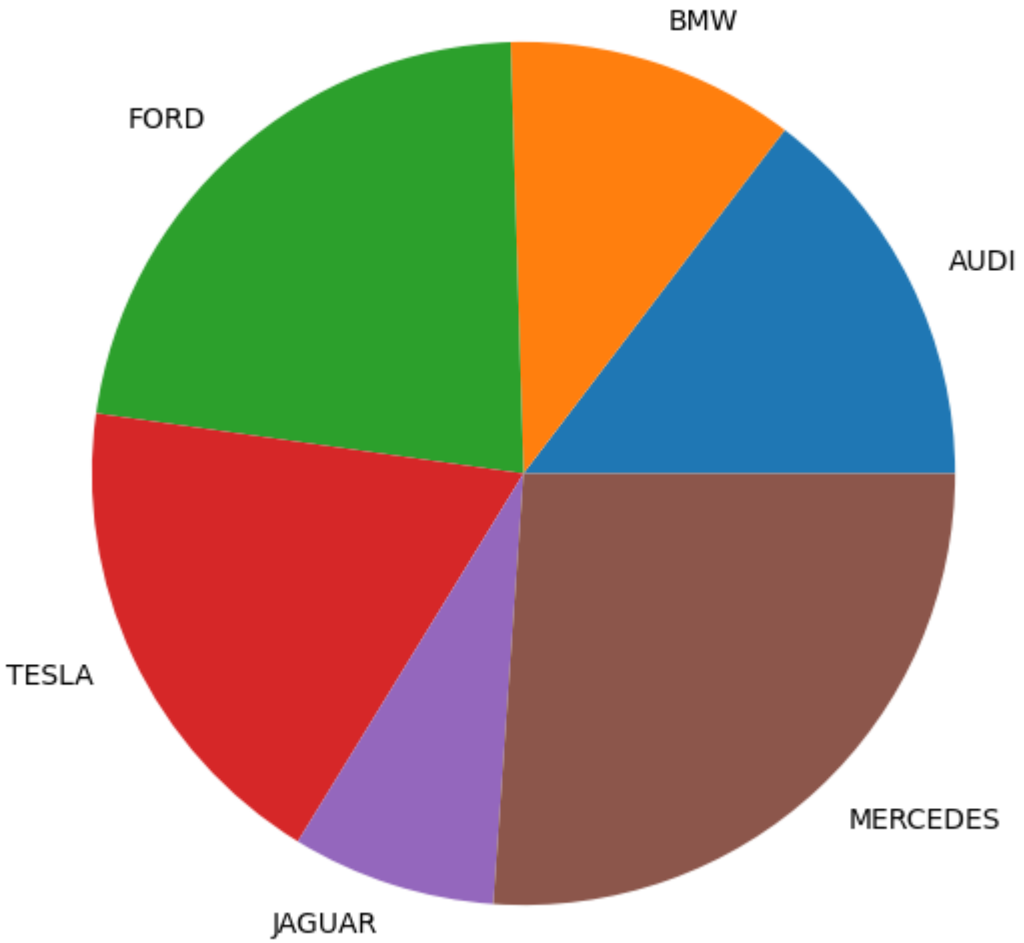



In [ ]:
```
#####################           Pie Chart          ##############################
'''
A Pie Chart is a circular statistical plot that can display only one series of data
The area of the chart is the total percentage of the given data. The area of slices
of the parts of the data.
The slices of pie are called wedges.
The area of the wedge is determined by the length of the arc of the wedge.
The area of a wedge represents the relative percentage of that part with respect to
Pie charts are commonly used in business presentations like sales, operations, surv
Matplotlib API has pie() function in its pyplot module which create a pie chart rep
```

```
Syntax: matplotlib.pyplot.pie(data, explode=None, labels=None, colors=None, autopct
Parameters:
data represents the array of data values to be plotted, the fractional area of each
labels is a list of sequence of strings which sets the label of each wedge.
color attribute is used to provide color to the wedges.
autopct is a string used to label the wedge with their numerical value.
shadow is used to create shadow of wedge.
'''
```

In [8]:

```
# Import libraries
from matplotlib import pyplot as plt
import numpy as np


# Creating dataset
cars = ['AUDI', 'BMW', 'FORD','TESLA', 'JAGUAR', 'MERCEDES']

data = [23, 17, 35, 29, 12, 41]

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

# show plot
plt.show()
```




In [ ]: