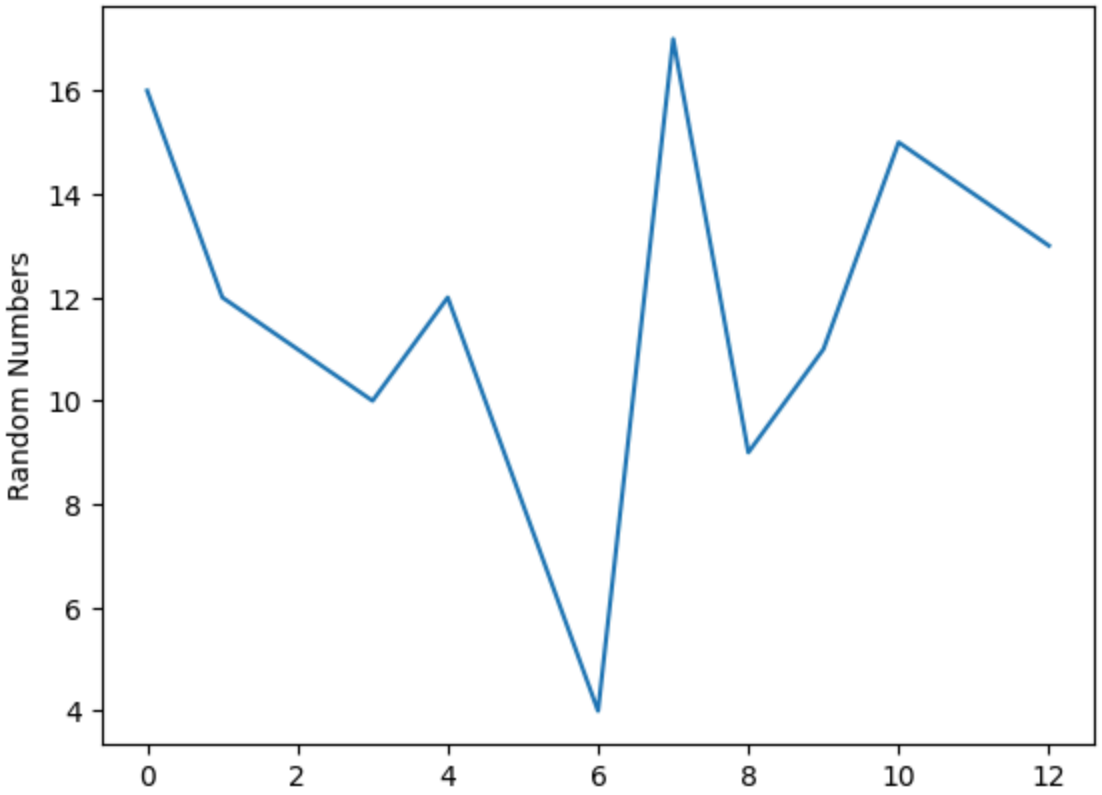
Charts using plot() Function

The plot() is a versatile command, and can take an arbitrary number of arguments. This means, we can plot figures corresponding to one axis, for two axes, considering single and multiple data etc.

In [5]:
```
#Program for drawing a line chart.
import matplotlib.pyplot as plt
#Creating a line chart of single list.
plt.plot([16,12,11,10,12,8,4,17,9,11,15,14,13])
#Adding label on y-axis.
plt.ylabel('Random Numbers')
#Displaying a line chart with the random numbers.
plt.show()
```

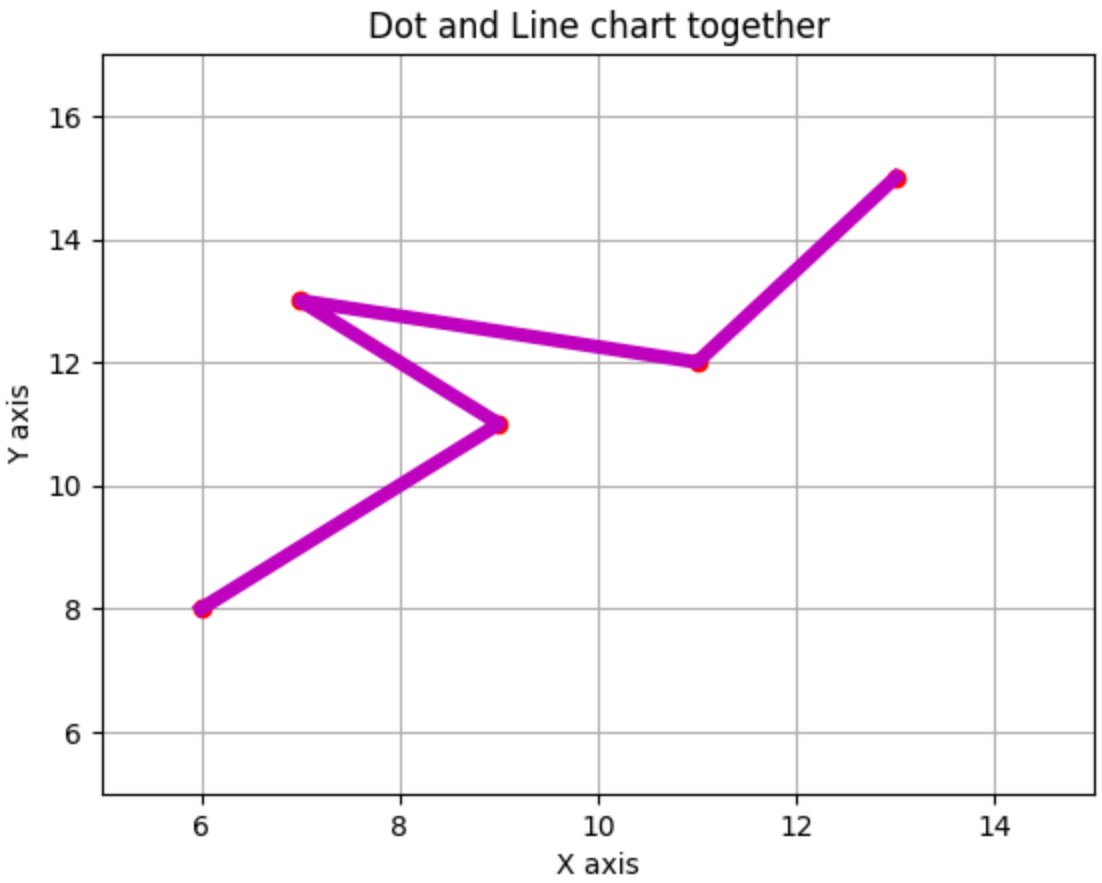


It is also possible to plot figures of different shapes and colours. For every x, y pair of arguments, there is an optional third argument which is the format string that indicates the colour and line type of the plot.

The letters and symbols of the format string are from MATLAB, for concatenating a colour string with a line style string. The default format string is 'b-', which is a solid blue line.

In [4]:
```
#Program to draw dot and lines on the same chart with axes limit.
import matplotlib.pyplot as plt
plt.plot([6,9,7,11,13], [8,11,13,12,15], 'ro', [6,9,7,11,13], [8,11,13,12,15],
```

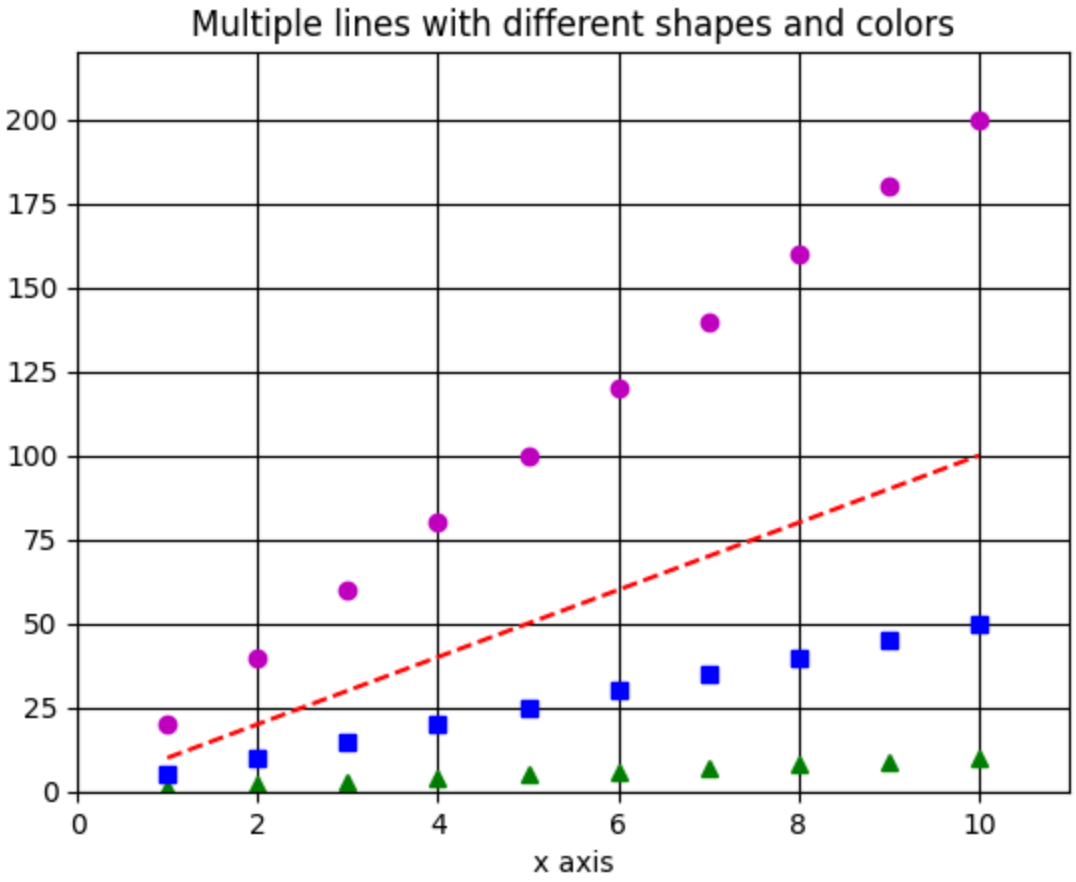
```
         'm',linewidth=5)
#Adding label to the x-axis.
plt.xlabel('X axis')
#Adding label to the y-axis.
plt.ylabel('Y axis')
#Adding title to the chart.
plt.title('Dot and Line chart together')
#Setting the limit of both the axes.
plt.axis ([5,15,5,17])
plt.grid(True)
#Displaying the chart.
plt.show()
```



In [6]:
```
#Creating four lists.
a=list (range (1, 11))
b=list (range (5, 55, 5))
c=list (range (10,110, 10))
d=list (range (20,210,20))
print ("First list: ", a)
print ("Second list:",b)
print ("Third list:", c)
print ("Fourth list:",d)
#Creating a chart with different colors and effects for different data.
plt.plot (a, a, 'g^', a, b, 'bs', a, c, 'r--', a, d, 'mo')
#Set limits of x-axis.
plt.xlim (0,11)
#Set limits of y-axis.
plt.ylim (0,220)
```

```
plt.xlabel('x axis')
plt.title('Multiple lines with different shapes and colors')
#Displaying grid in chart.
plt.grid (True, color='k')
#Displaying the chart.
plt.show()
```

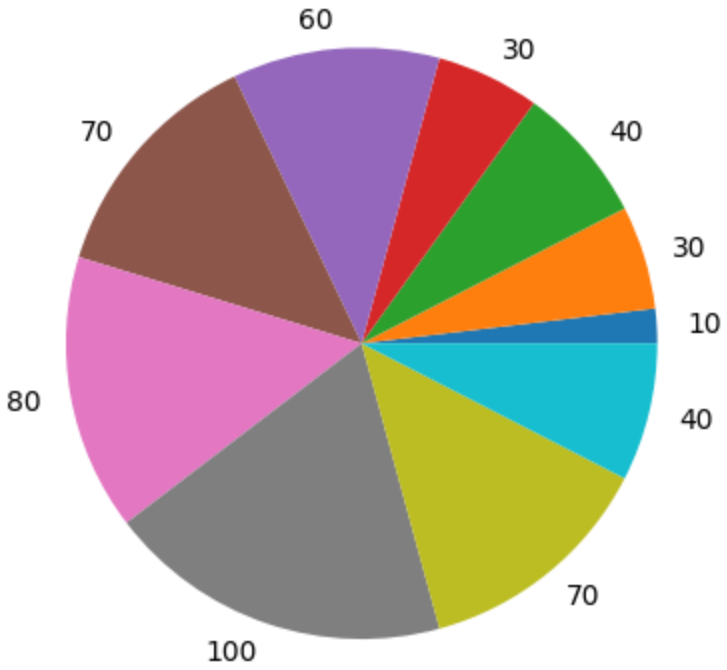
```
First list:  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Second list: [5, 10, 15, 20, 25, 30, 35, 40, 45, 50]
Third list: [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
Fourth list: [20, 40, 60, 80, 100, 120, 140, 160, 180, 200]
```



Pie chart

Pie charts visualize the absolute and relative frequencies. A pie chart is a circle partitioned into segments where each of the segments represents a category. The size of each segment depends upon the relative frequency and is determined by the angle. A pie-chart is a representation of values as slices of a circle with different colours. The pie chart is drawn using pie() function and considering single list as an argument.

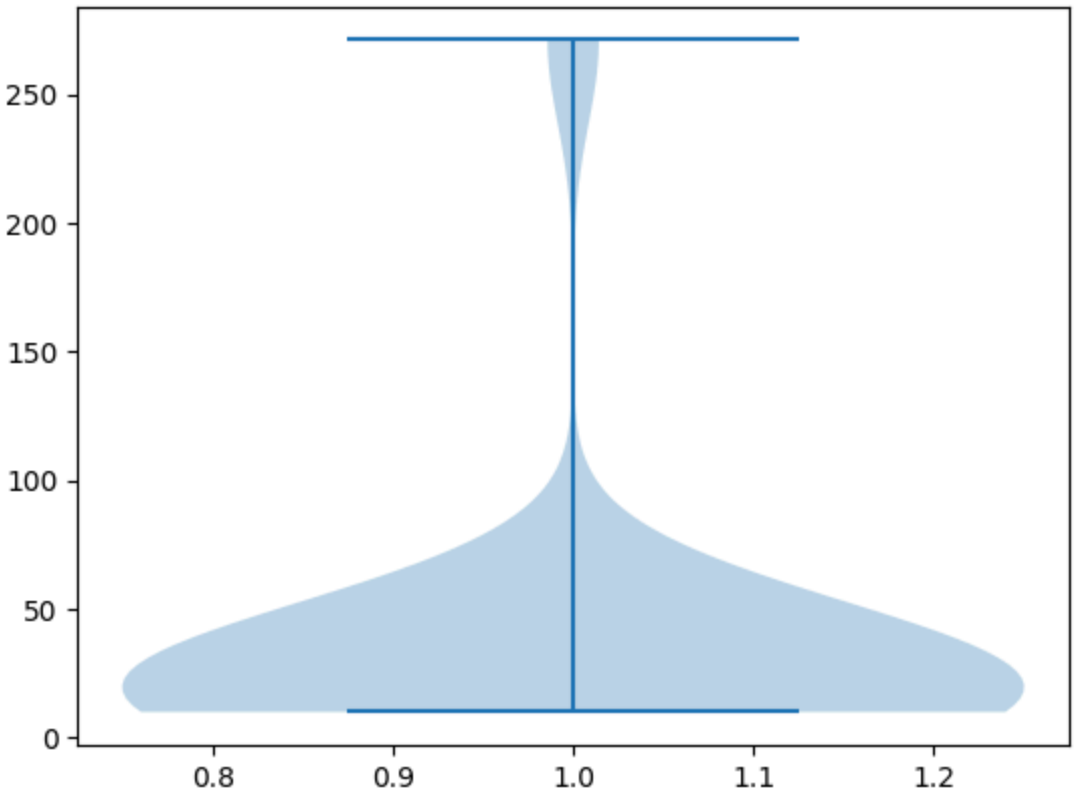
In [8]:
```
list1= [10, 30, 40, 30, 60, 70, 80, 100,70,40]
#Creating and displaying a pie chart.
plt.pie (list1, labels=list1)
plt.show()
```



Violin Plot

This plot is a combination of box and Kernel density plot. It is drawn using violinplot() from matplotlib.pyplot

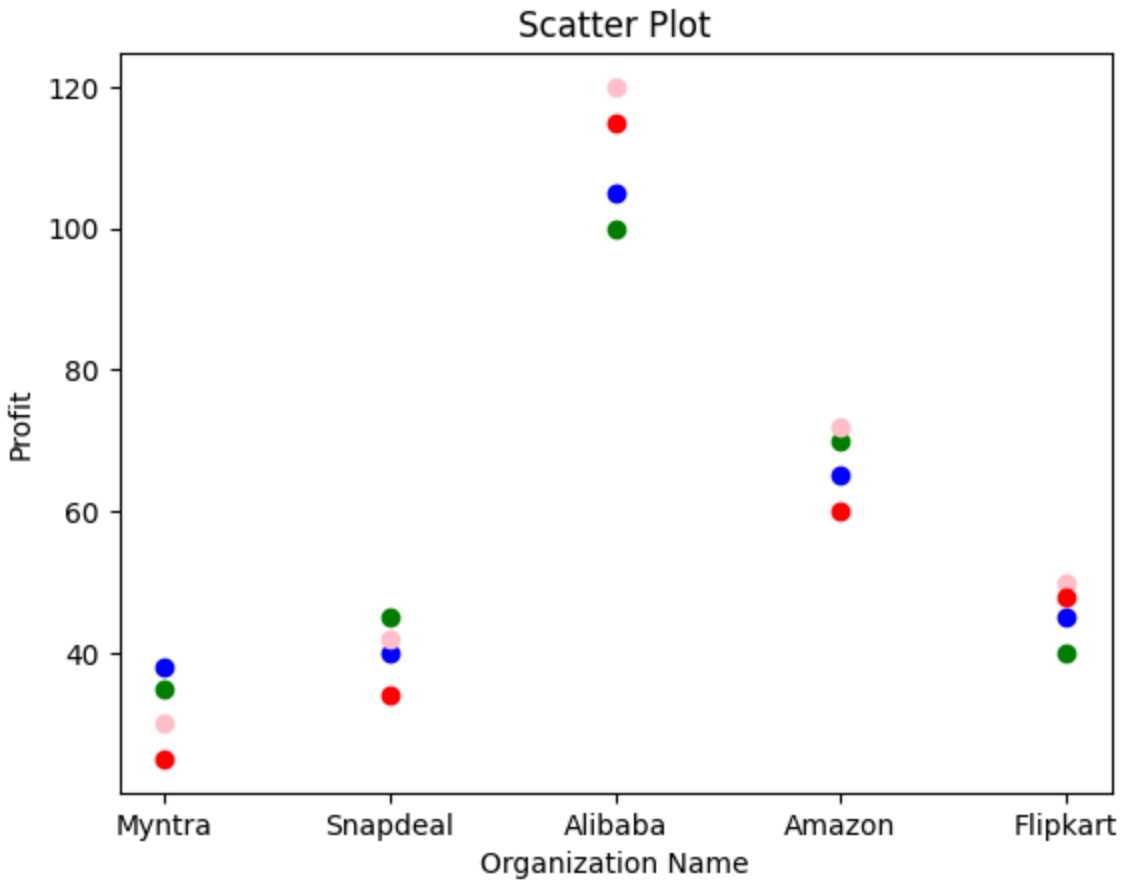
In [10]:
```
plt.violinplot([10,11,12,18,23,27,29,30,26,28,24,23,14,12,16,12,12,27,271])
#Displaying the chart.
plt.show()
```



Scatter Plot

The scatter plot is created using the scatter() function and is helpful in displaying bivariate data. Scatter plots show many points plotted in the Cartesian plane. Each point represents the values of two variables. One variable is chosen in the horizontal axis and another in the vertical axis. The "color" is the default argument in the function which represents the color of the dot.

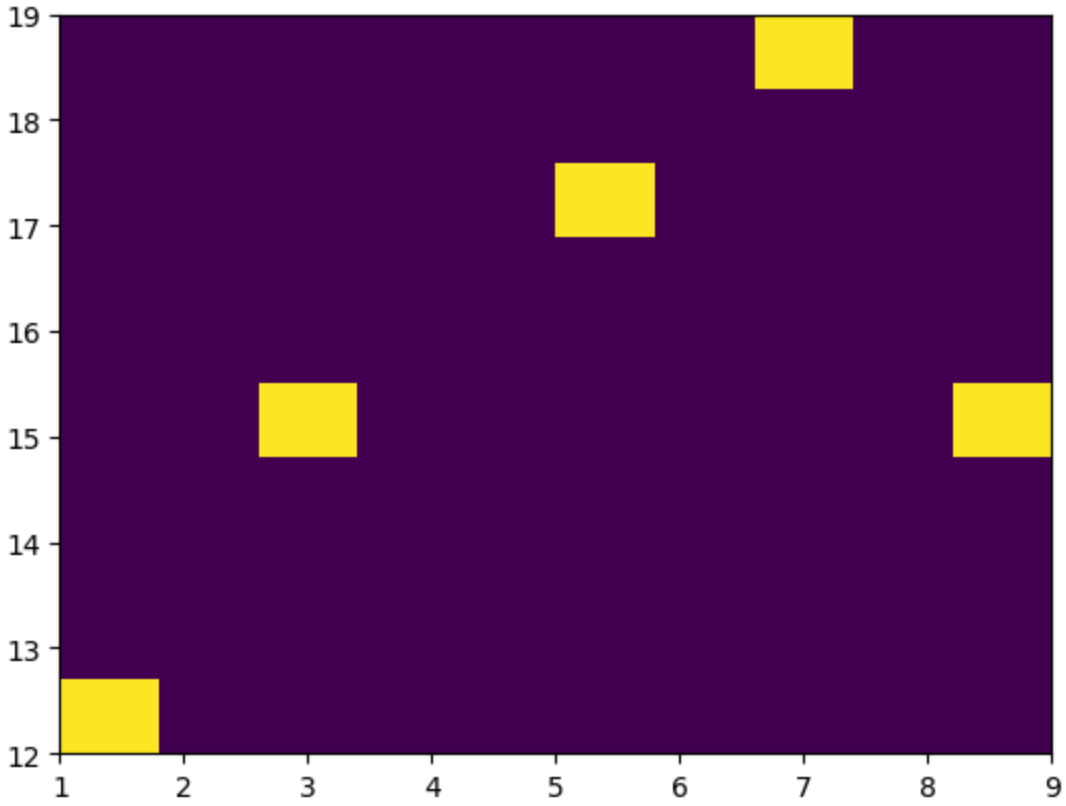
In [13]:
```
#Program to draw a scatter plot.
ecommerce=['Myntra', 'Snapdeal', 'Alibaba', 'Amazon', 'Flipkart']
Q1_Profit=[35, 45, 100, 70, 40]
Q2_Profit=[38, 40,105, 65, 45]
Q3_Profit=[30, 42, 120, 72, 50]
Q4_Profit=[25, 34, 115, 60, 48]
#Creating a scatter plot.
plt.scatter (ecommerce, Q1_Profit, color='green')
plt.scatter (ecommerce, Q2_Profit, color='blue')
plt.scatter (ecommerce, Q3_Profit, color='pink')
plt.scatter (ecommerce, Q4_Profit, color='red')
#Adding title to the chart and labels to the axis.
plt.xlabel('Organization Name')
plt.ylabel('Profit')
plt.title('Scatter Plot')
#Displaying a scatter plot.
plt.show()
```



Histogram

Histogram is based on the idea to categorize the data into different groups and plot the bars for each category with height. A histogram represents the frequencies of values of a variable gathered into ranges. Each bar in histogram represents the height of the number of values present in that range.

In [15]:
```
#Creating a histogram using hist2d() function.
list1=list (range (1,10,2))
list2=[12,15,17,19,15]
plt.hist2d(list1, list2)
#Displaying the histogram.
plt.show()
```



Bar chart

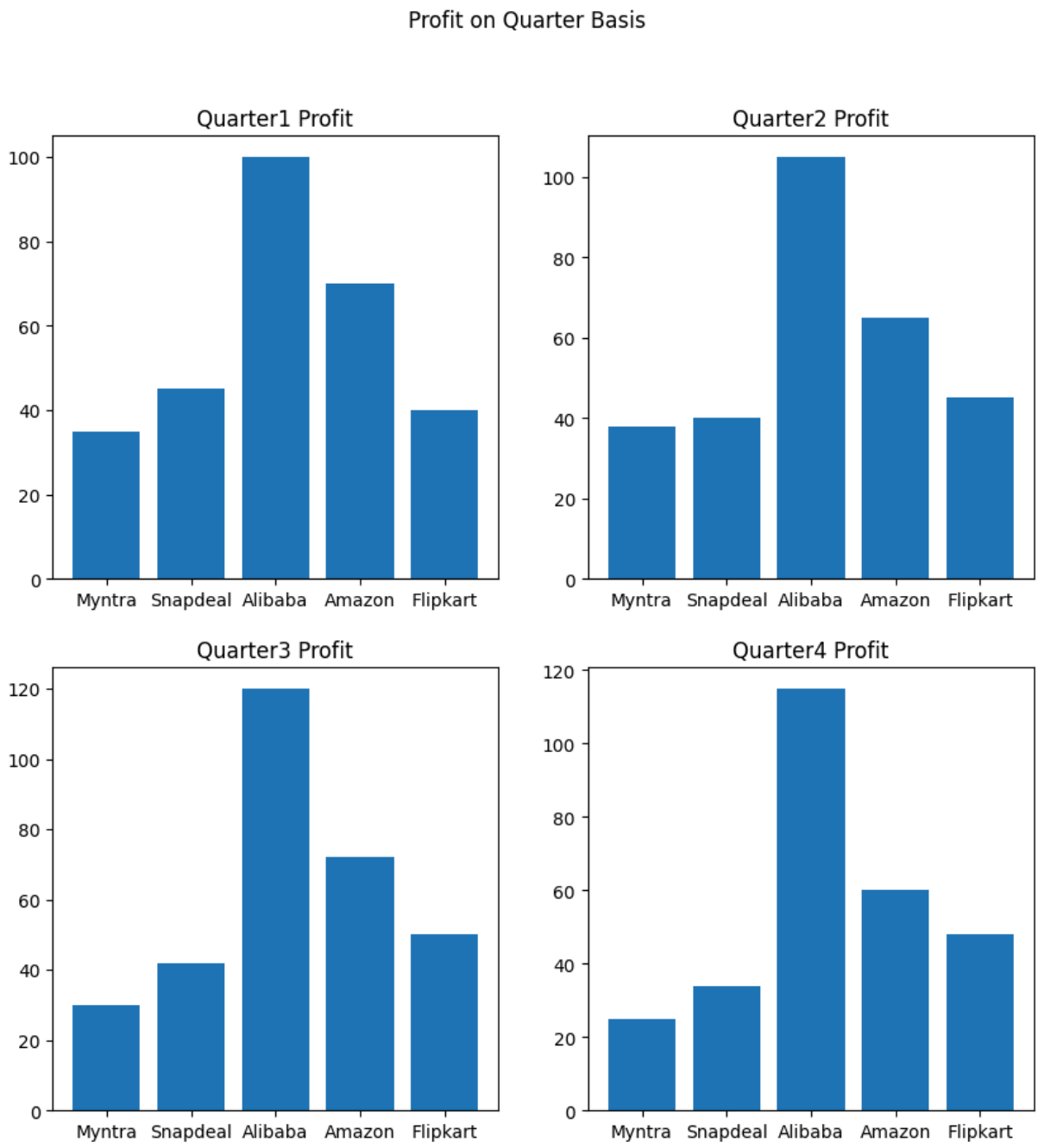
A bar chart visualizes the relative or absolute frequencies of observed values of a variable. It consists of one bar for each category. A bar chart represents data in rectangular bars with length of the bar proportional to the value of the variable. Python uses the function bar() to create bar charts. The height of each bar is determined by either the absolute frequency or the relative frequency of the respective category and is shown on the y-axis. However, the horizontal bar chart can be created by using the barh() function

Creating Multiple charts on one image

It is possible to have more than one charts on one chart in the form of m × n array of charts using the function subplot().

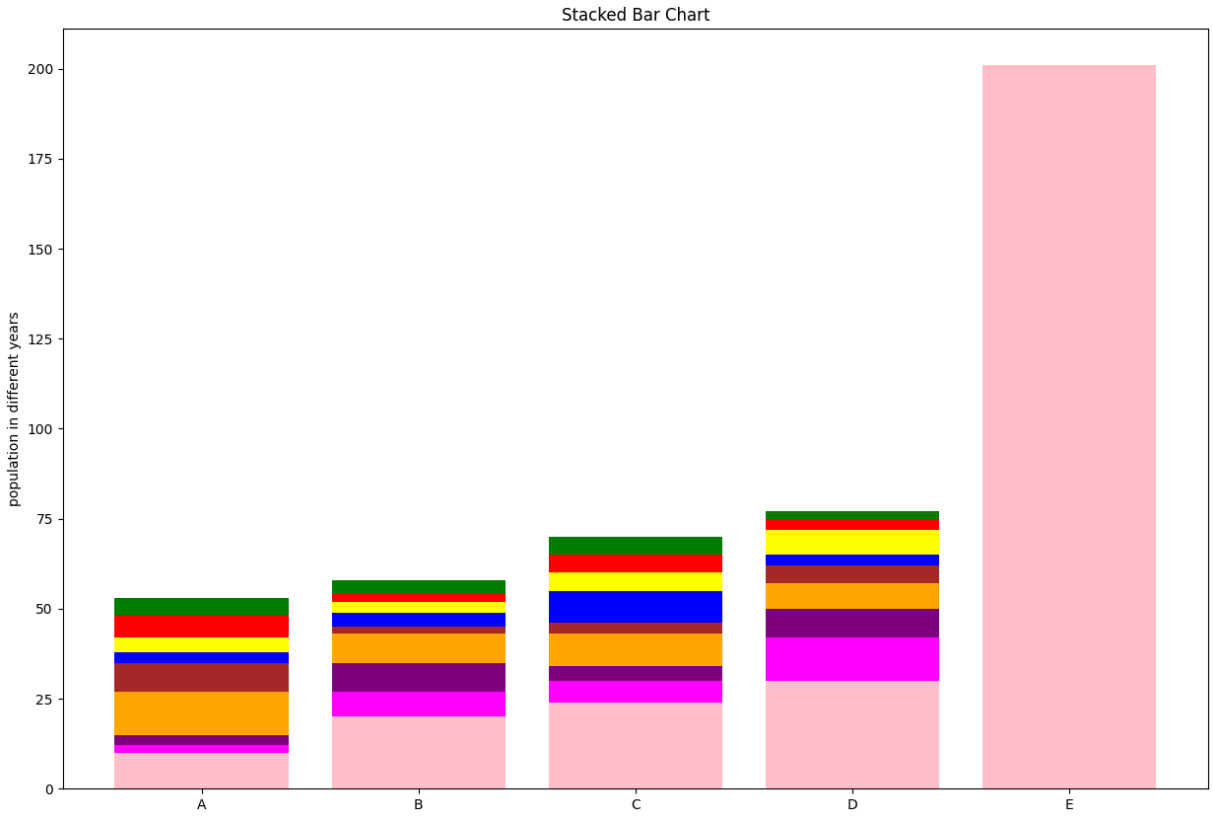
In [16]:
```
#Program to draw a scatter plot.
ecommerce=['Myntra', 'Snapdeal', 'Alibaba', 'Amazon', 'Flipkart']
Q1_Profit=[35, 45, 100, 70, 40]
Q2_Profit=[38, 40,105, 65, 45]
Q3_Profit=[30, 42, 120, 72, 50]
Q4_Profit=[25, 34, 115, 60, 48]
#Creating different bar charts on one image using subplot() function.
plt.figure(1, figsize=(10,10))
#Creating bar chart in first cell of figure having 2 rows, 3 columns.
plt.subplot (221)
```

```
plt.bar (ecommerce, Q1_Profit)
plt.title('Quarter1 Profit')
#Creating a bar chart in second cell.
plt.subplot (222)
plt.bar (ecommerce, Q2_Profit)
plt.title('Quarter2 Profit')
#Creating a bar chart in third cell.
plt.subplot (223)
plt.bar (ecommerce, Q3_Profit)
plt.title('Quarter3 Profit')
# Creating a bar chart in fourth cell.
plt.subplot (224)
plt.bar (ecommerce, Q4_Profit)
plt.title('Quarter4 Profit')
#Adding a Main title for the figure.
plt.suptitle('Profit on Quarter Basis')
#Displaying the chart.
plt.show()
```



In [20]:
```
#Program to create a stacked bar chart.
plt.figure(1, figsize=(15,10))
countries=['A', 'B', 'C', 'D', 'E']
Population_1930=[10,20,24,30,201]
Population_1940=[12, 27, 30, 42, 26]
Population_1950=[15, 35, 34, 50,33]
Population_1960=[27,43,43,57,38]
Population_1970=[35, 45, 46, 62, 40]
Population_1980=[38, 49, 55, 65, 45]
Population_1990=[42, 52, 60, 72, 50]
Population_2000=[48,54, 65, 75,58]
Population_2010=[53,58,70,77,63]
#Creating a stacked bar chart.
plt.bar (countries, Population_2010, color='green')
plt.bar (countries, Population_2000, color='red')
```

```
plt.bar (countries, Population_1990, color='yellow')
plt.bar (countries, Population_1980, color='blue')
plt.bar (countries, Population_1970, color='brown')
plt.bar (countries, Population_1960, color='orange')
plt.bar (countries, Population_1950, color='purple')
plt.bar (countries, Population_1940, color='magenta')
plt.bar (countries, Population_1930, color='pink')
#Adding labels and title to the chart. plt.xlabel('Countries')
plt.ylabel('population in different years')
plt.title('Stacked Bar Chart')
#Displaying a bar chart.
plt.show()
```

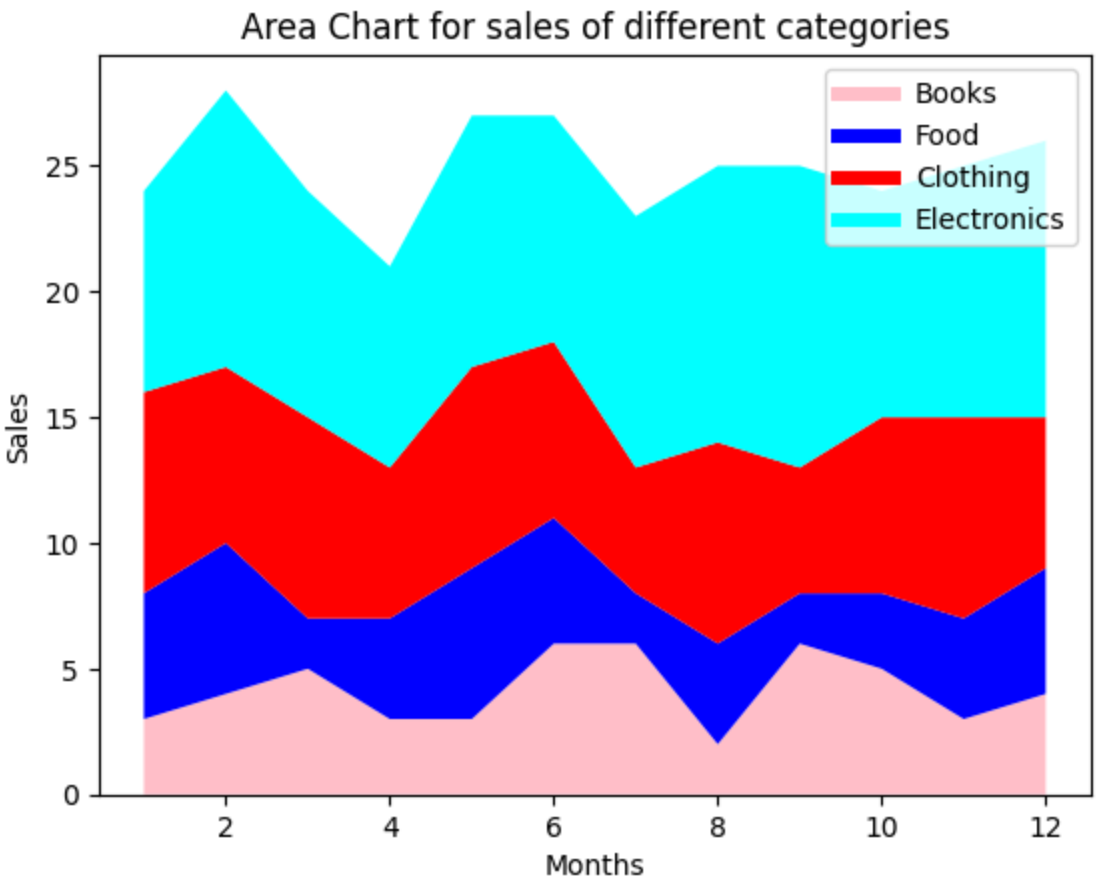


Area Plot

Area plots are pretty much similar to the line plot. They are also known as stack plots. These plots can be used to track changes over time for two or more related groups that make up one whole category.

In [22]:
```
month_sales = [1,2,3,4,5,6,7,8,9,10,11,12]
Electronics = [8,11,9,8,10,9,10,11,12,9,10,11]
Clothing=[8,7,8,6,8,7,5,8,5,7,8,6]
Food=[5,6,2,4,6,5,2,4,2,3,4,5]
Books=[3,4,5,3,3,6,6,2,6,5,3,4]
plt.plot([], [], color='pink', label='Books', linewidth=5)
plt.plot([], [], color='blue', label='Food', linewidth=5)
plt.plot([], [], color='red', label='Clothing', linewidth=5)
plt.plot([], [], color='cyan', label='Electronics', linewidth=5)
```

```
plt.stackplot (month_sales, Books, Food, Clothing, Electronics,
               colors= ['pink', 'blue', 'red', 'cyan'])
plt.xlabel('Months')
plt.ylabel('Sales')
plt.title('Area Chart for sales of different categories')
plt.legend()
#Displaying the chart.
plt.show()
```



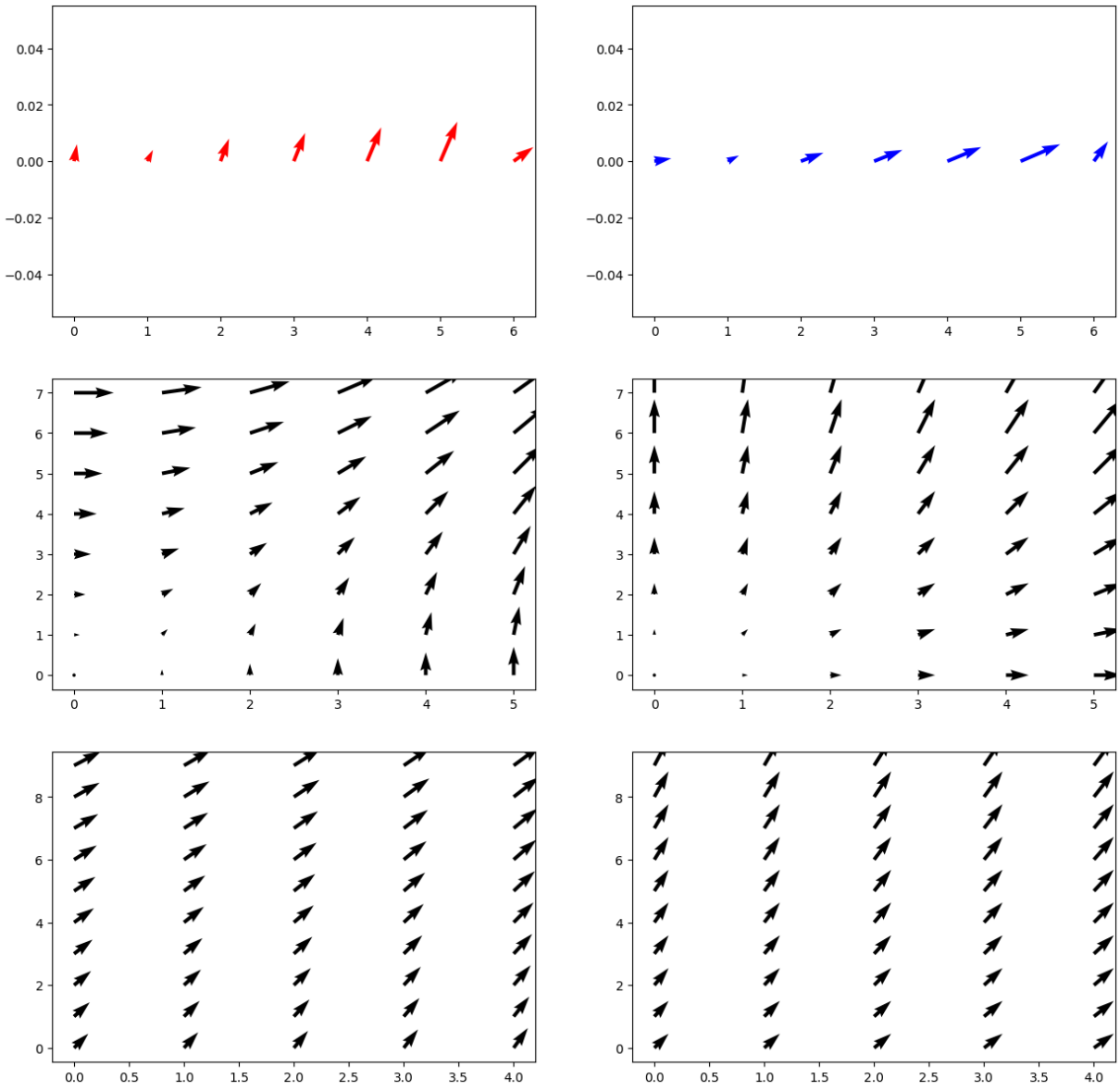
Quiver Plot

The quiver() function takes two important arguments that represents 1-D or 2-D arrays or sequences. The quiver plots a 2-D field of arrows representing the two data sets. Color is an optional argument which represents the color of the arrow.

In [24]:
```
#Program to draw multiple quiver plots in single image.
import matplotlib.pyplot as plt
import numpy as np
a=[10, 20, 30, 40, 50, 60, 70]
b=[60,40, 80, 100, 120, 140, 50]
plt.figure(1, figsize=(15,15))
#Creating a quiver plot in first cell of image having 3 rows, 2 columns.
plt.subplot (3,2,1)
plt.quiver (a,b,color='r')
# Creating a quiver plot in second cell.
```

```
plt.subplot (3,2,2)
plt.quiver (b, a, color='b')
#Creating a quiver plot in third cell.
plt.subplot (3,2,3)
x=8
y=6
M, N= np.mgrid[0:x, 0:y]
plt.quiver (M,N)
# Creating a quiver plot in fourth cell.
plt.subplot (3,2,4)
plt.quiver (N,M)
#Creating a quiver plot in fifth cell.
plt.subplot (3,2,5)
x=20
y=15
M, N= np.mgrid[10:x, 10:y]
plt.quiver (M, N)
#Creating a quiver plot in sixth cell.
plt.subplot (3,2,6)
plt.quiver (N,M)
plt.suptitle('Quiver Plots')
#Displaying the image.
plt.show()
```

Quiver Plots



In [ ]: