Data Analytics & Machine Learning using Python

Session : 08 June 2020

Data Virtualization

- Data Visualization allows us to quickly interpret the data and adjust different variables to see their effect.
- Briefing Matplotlib
- Methods of Plotting
- Bar Graph

Data Visualization Libraries

1. Matplotlib
2. Ggplot
3. Plotly
4. Geoplotlib
5. Seaborn

Matplotlib

- Matplotlib is a multiplatform data virtualization library built on Numpy Arrays
- One of Matplotlib's most important features is its ability to play well with many operating systems and graphics backends.

PyPlot

- Pyplot is a shell-like interface to Matplotlib, to make it easir for the people who are used to Matlab.

Importing Matplotlib

In [98]:
```
# 1st Method
import matplotlib as mpl
# mpl is shorthand for matplotlib
```

In [1]:
```
# 2nd Method
import matplotlib.pyplot as plt
import numpy as np
# plt is shorthand for matplotlib.pyplot
```

Types of Graph

- Sinusoidal Graph
- Line Graph
- Bar Graph
- Box Plot
- Scatter Plot
- Pie Chart
- Histogram

Plotting from an IPython notebook

- Plotting with an IPython notebook can be done with %matplotlib command. You also have the options of embedding graphics directly in the notebook, with two possible options
  - %matplotlib inline will lead to static images of your plot embedded in the notebook

In [202]:
```
%matplotlib inline
```

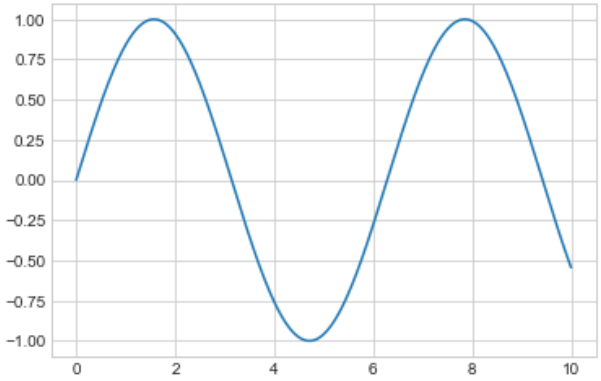
In [3]:
```
plt.plot?
```

Sinusoidal Graph

- figure() - used to hold the graph as an figure object
- axes() - used to create the axes in graph
- linspace() - used to generate the numerical sequence with help of numpy array
- plot() - used to plot the graph

In [205]:
```
# Sine Wave
x=np.linspace(0,10,1000)
plt.plot(x,np.sin(x))
```

Out[205]:
```
[<matplotlib.lines.Line2D at 0x1cd209a3888>]
```



In [94]:
```
plt.plot?
```

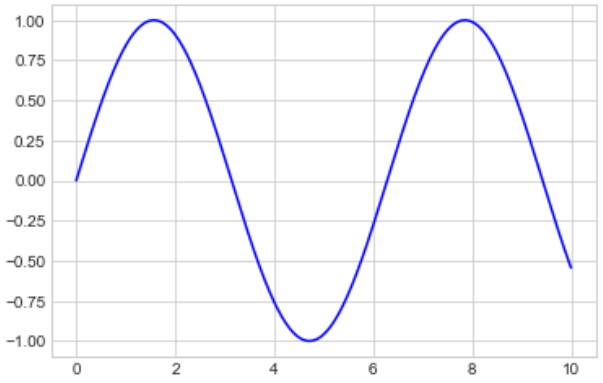
Adjusting the Color

In [206]:
```
# Specify color by name
plt.plot(x,np.sin(x),color='red')
```

Out[206]:
```
[<matplotlib.lines.Line2D at 0x1cd20a516c8>]
```

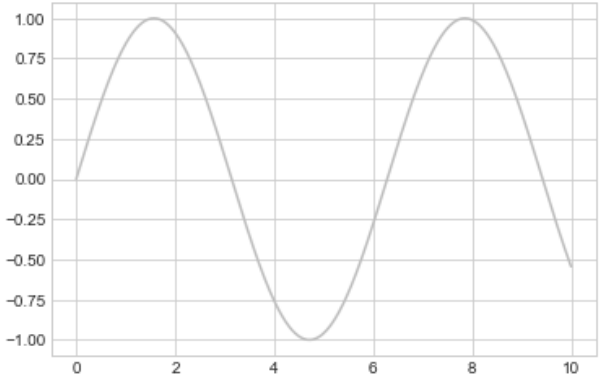
In [207]:
```
# Specify color code (rgbcmyk)
plt.plot(x,np.sin(x),color='b')
```

Out[207]:
```
[<matplotlib.lines.Line2D at 0x1cd20abc948>]
```



In [208]:
```
# specify with Grayscale between 0 & 1
plt.plot(x,np.sin(x),color='0.74')
```

Out[208]:
```
[<matplotlib.lines.Line2D at 0x1cd20b26fc8>]
```

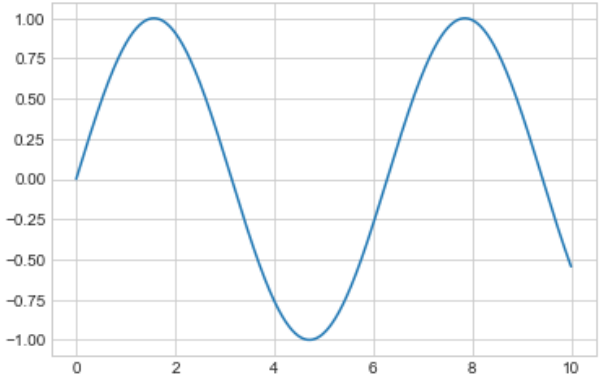


Adjusting the Line

- You can adjust the line style using the linestyle keyword

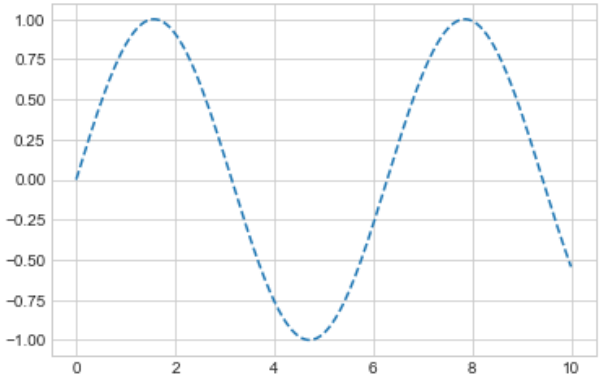
In [209]:
```
# Method
plt.plot(x,np.sin(x), linestyle='solid') #Solid Line
```

Out[209]:
```
[<matplotlib.lines.Line2D at 0x1cd20bbf748>]
```



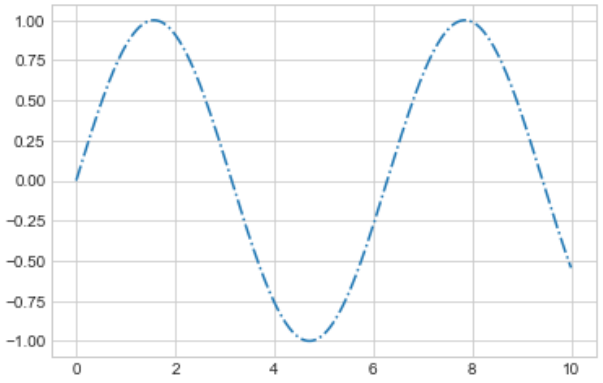
In [211]:
```
# Method 2
plt.plot(x,np.sin(x), linestyle='dashed')      # Dashed Line
```

Out[211]: [<matplotlib.lines.Line2D at 0x1cd219b5dc8>]



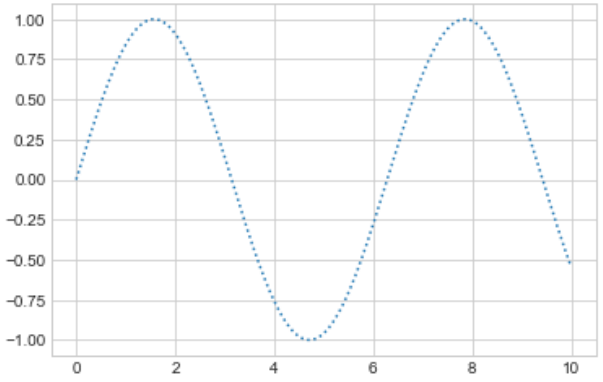
In [218]:
```
# Method 3
plt.plot(x,np.sin(x), linestyle='dashdot')              #DashDot
```

Out[218]: [<matplotlib.lines.Line2D at 0x1cd20eca3c8>]



In [219]:
```
# Method 4
plt.plot(x,np.sin(x), linestyle='dotted')                       # Dotted
```

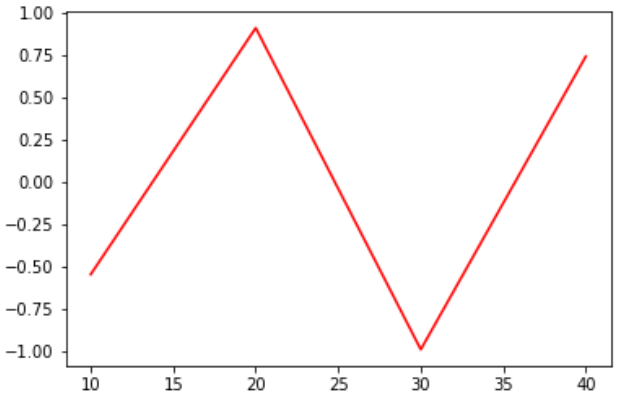
Out[219]: [<matplotlib.lines.Line2D at 0x1cd20f5ca08>]



These linestyle and color codes can be combined into a single nonkeyword argument to be built in plt.plot() function

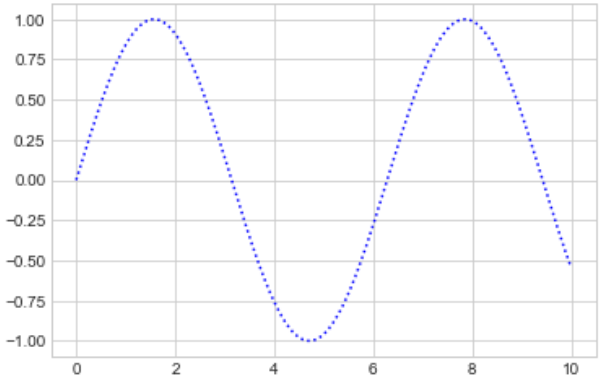
In [18]:
```
# Time & Amplitude
# Linestyle & Color 1
time=[10,20,30,40]
amplitude=np.sin(time)
plt.plot(time,amplitude,'r-')
```

Out[18]: [<matplotlib.lines.Line2D at 0x198990eb988>]



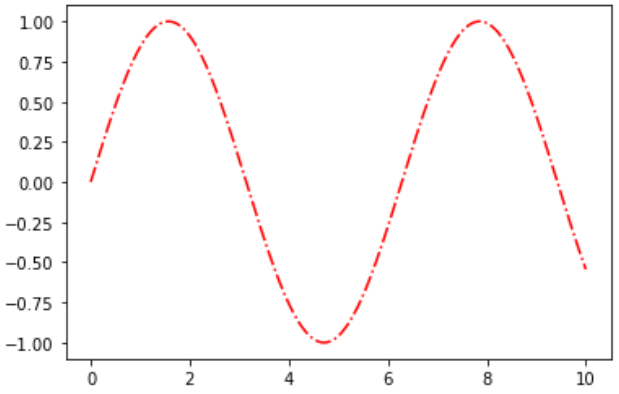
In [225]:
```
plt.plot(x,np.sin(x),'b:')
```

Out[225]: [<matplotlib.lines.Line2D at 0x1cd2131ef48>]

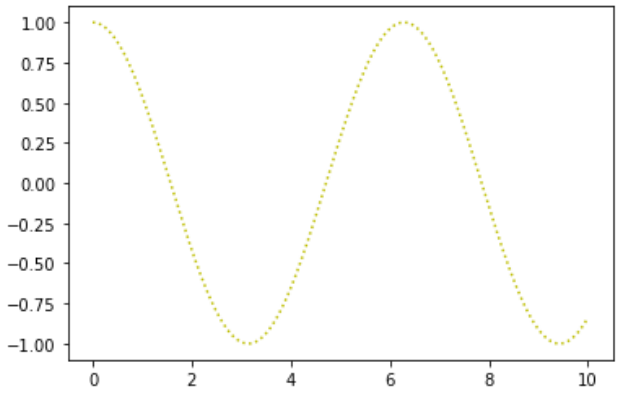


In [33]:
```
# Linestyle & Color 3
plt.plot(x,np.sin(x), 'r-.')
```

Out[33]: [<matplotlib.lines.Line2D at 0x1cd19b24908>]



In [42]:
```
# Linestyle & Color 4
plt.plot(x,np.cos(x), 'y:')
plt.show()
```



Marker Style

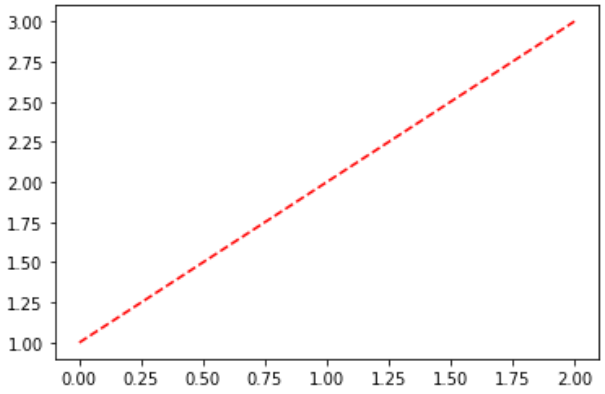
- Markers are, by default, drawn as point markers. They are just a location on the figure where segments join

Marker abbreviation	Marker style
.	Point marker
,	Pixel marker
o	Circle marker
v	Triangle down marker
^	Triangle up marker
<	Triangle left marker
>	Triangle right marker
1	Tripod down marker
2	Tripod up marker
3	Tripod left marker
4	Tripod right marker
s	Square marker
p	Pentagon marker
*	Star marker
h	Hexagon marker
H	Rotated hexagon marker
+	Plus marker
x	Cross (x) marker
D	Diamond marker
d	Thin diamond marker
|	Vertical line (vline symbol) marker
_	Horizontal line (hline symbol) marker

To Display the Plot

In [46]:
```
#Example 1
plt.plot([1,2,3], color='red', linestyle='--')
```

Out[46]:
```
[<matplotlib.lines.Line2D at 0x1cd1af107c8>]
```



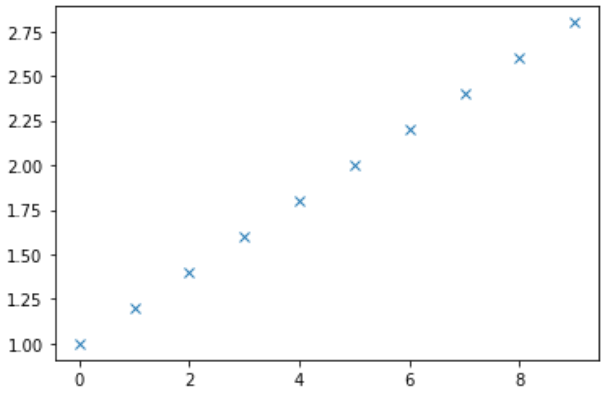
plt.show()

- if you are using Matplotlib from within a script, the functions "plt.show()" is your friend.
- "plt.show()" starts and event loop, looks for all currently active figure objects, and opens window that display your figure
- The "plt.show()" command does a lot under the hood, as it must interact with your system's interactive graphical backend

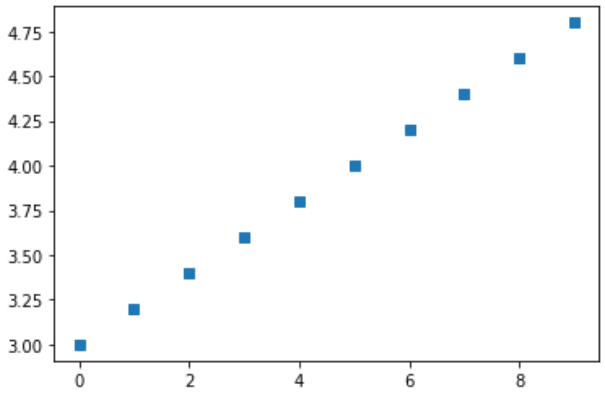
In [47]:
```
# Creating the Numpy Array
y = np.arange(1, 3, 0.2)
```

In [48]:
```
plt.plot(y, 'x')       # Cross Marker
```

Out[48]:
```
[<matplotlib.lines.Line2D at 0x1cd1af83d48>]
```

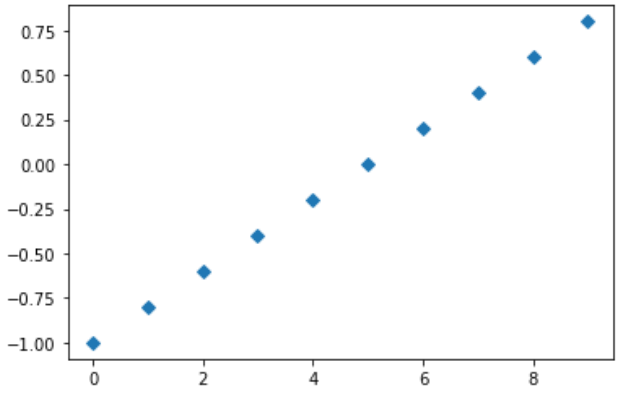


In [49]:
```
plt.plot(y+2, 's');       # Square
```



In [50]:
```
plt.plot(y-2,'D')         # Diamond
```

Out[50]:
```
[<matplotlib.lines.Line2D at 0x1cd1b052248>]
```



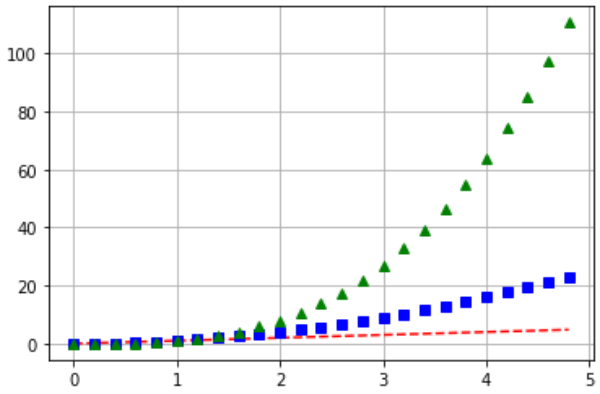
Labelling a Plot

- Labeling of the plot is done : **Grid, Title, Axis, Labels and Legends**

Plot Grid

- We can add a grid to the plot by calling the grid() function; it takes one parameter, a Boolean value, to enable (if True) or disable (if False) the grid

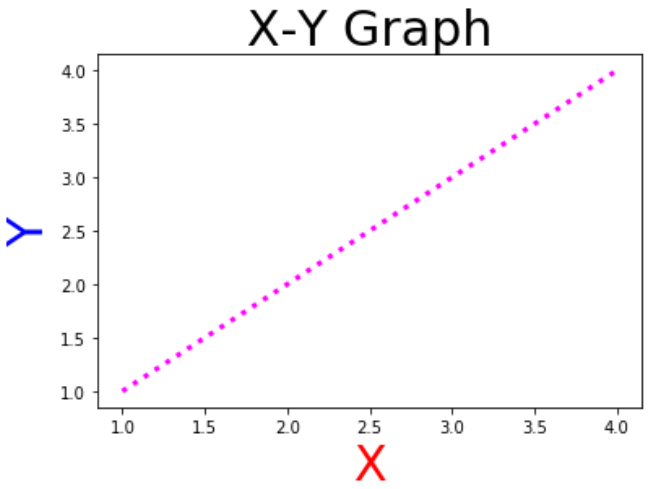
In [78]:
```
# Example 1
t = np.arange(0., 5., 0.2)
plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3, 'g^')
plt.grid(True)
```



Plot Title and Labels

In [10]:
```
# Example 1
x=np.arange(1,5)
y=np.arange(1,5)
plt.plot(x,y,color='magenta',linestyle=':',linewidth=3.0, visible=True,markersize=100)
plt.xlabel('X',size=30,color='red')
plt.ylabel('Y',size=30,color='blue')
plt.title('X-Y Graph',size=30,color='black')
```

Out[10]: `Text(0.5, 1.0, 'X-Y Graph')`

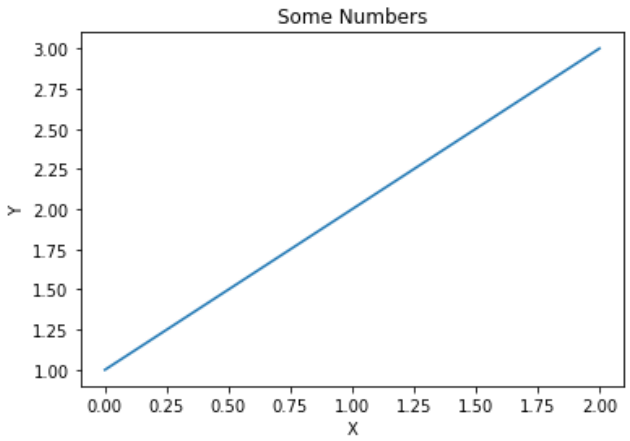


Line Graph

- A line chart (aka line plot, line graph) uses points connected by line segments from left to right to demonstrate changes in value
- Line graphs can be used to show how data changes over time and are often used to communicate trends, such as how household income changes each year

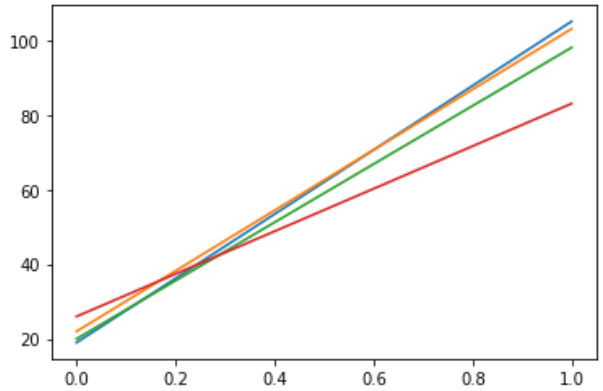
In [96]:
```
# Example 1
plt.plot([1,2,3])
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Some Numbers')
```

Out[96]:
```
Text(0.5, 1.0, 'Some Numbers')
```



In [97]:
```
guests=[19,22,20,26]
subscribers=[105,103,98,83]
plt.plot([guests,subscribers])
```

Out[97]:
```
[<matplotlib.lines.Line2D at 0x1cd1c3a4c48>,
 <matplotlib.lines.Line2D at 0x1cd1c396688>,
 <matplotlib.lines.Line2D at 0x1cd1c3963c8>,
 <matplotlib.lines.Line2D at 0x1cd1c3967c8>]
```

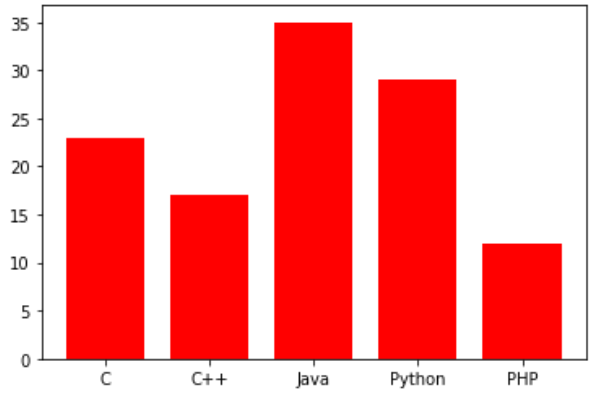


Bar Graph

- A bar chart or bar graph is a chart or graph that presents categorical data with rectangular bars with heights or lengths proportional to the values that they represent. The bars can be plotted vertically or horizontally.

x	sequence of scalars representing the x coordinates of the bars. align controls if x is the bar center (default) or left edge.
height	scalar or sequence of scalars representing the height(s) of the bars.
width	scalar or array-like, optional. the width(s) of the bars default 0.8
bottom	scalar or array-like, optional. the y coordinate(s) of the bars default None.
align	{'center', 'edge'}, optional, default 'center'

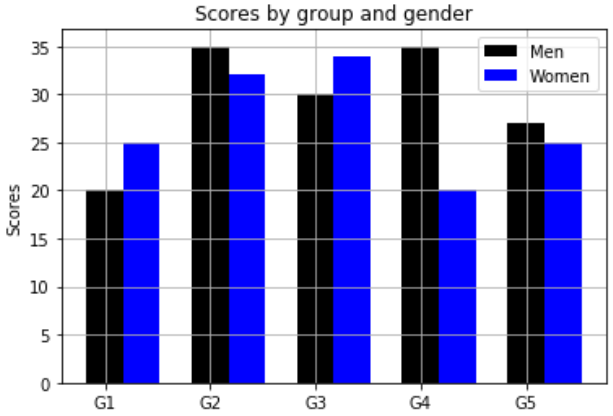
In [11]:
```
# Example 1
langs = ['C', 'C++', 'Java', 'Python', 'PHP']
students = [23,17,35,29,12]
x=plt.bar(langs,students,width=0.75,color='red')
plt.show()

for i in x:
    x.annotate(str(i.get_height()))
```



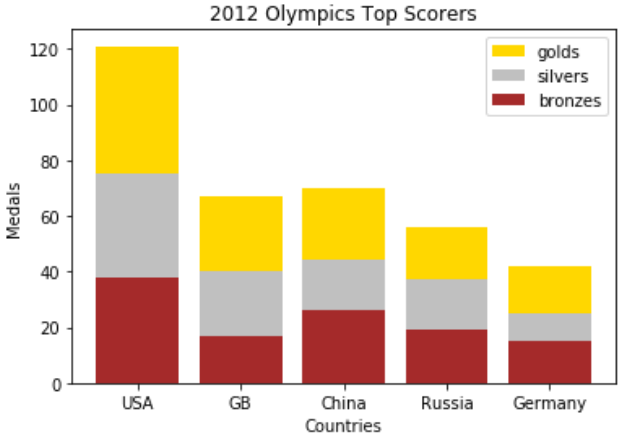

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-11-4b0dea404763> in <module>
      6
      7 for i in x:
----> 8     x.annotate(str(i.get_height()))

AttributeError: 'BarContainer' object has no attribute 'annotate'
```

In [99]:
```
plt.figure?
```

In [122]:
```
# Example 2
N = 5
men_means = (20, 35, 30, 35, 27)
women_means = (25, 32, 34, 20, 25)

ind = np.arange(N)
width = 0.35
plt.bar(ind, men_means, width, label='Men',color='black')
plt.bar(ind + width, women_means, width,label='Women',color='blue')
plt.grid(True)
plt.ylabel('Scores')
plt.title('Scores by group and gender')

plt.xticks(ind,('G1', 'G2', 'G3', 'G4', 'G5'))
plt.legend()
plt.show()

# ticks are used to show specific points on the coordinate axis
```



In [20]:
```
# Example 3
countries = ['USA', 'GB', 'China', 'Russia', 'Germany']
bronzes = np.array([38, 17, 26, 19, 15])
silvers = np.array([37, 23, 18, 18, 10])
golds = np.array([46, 27, 26, 19, 17])
ind = [x for x, _ in enumerate(countries)]

plt.bar(ind, golds, width=0.8, label='golds', color='gold', bottom=silvers+bronzes)
plt.bar(ind, silvers, width=0.8, label='silvers', color='silver', bottom=bronzes)
plt.bar(ind, bronzes, width=0.8, label='bronzes', color='brown')

plt.xticks(ind, countries)
plt.ylabel("Medals")
plt.xlabel("Countries")
plt.legend(loc="upper right")
plt.title("2012 Olympics Top Scorers")
plt.grid(False)
plt.show()
```



Horizontal Bar Graph

In [136]:
```
# Example 4
langs = ['C', 'C++', 'Java', 'Python', 'PHP']
students = [23,17,35,29,12]
plt.barh(langs,students,color='red')
plt.show()
```