In [1]:
```
### pandas and matplotlib
import pandas as pd
filePath = 'data files/RegularSeasonCompactResults.csv'
def readCsvData(filePath):
    return pd.read_csv(filePath)
readCsvData(filePath)
```

Out[1]:

	Season	Daynum	Wteam	Wscore	Lteam	Lscore	Wloc	Numot
0	1985	20	1228	81	1328	64	N	0
1	1985	25	1106	77	1354	70	H	0
2	1985	25	1112	63	1223	56	H	0
3	1985	25	1165	70	1432	54	H	0
4	1985	25	1192	86	1447	74	H	0
5	1985	25	1218	79	1337	78	H	0
6	1985	25	1228	64	1226	44	N	0
7	1985	25	1242	58	1268	56	N	0
8	1985	25	1260	98	1133	80	H	0
9	1985	25	1305	97	1424	89	H	0
10	1985	25	1307	103	1288	71	H	0
11	1985	25	1344	75	1438	71	N	0
12	1985	25	1374	91	1411	72	H	0
13	1985	25	1412	70	1397	65	N	0
14	1985	25	1417	87	1225	58	H	0
15	1985	26	1116	65	1368	62	H	0
16	1985	26	1120	92	1391	50	H	0
17	1985	26	1135	65	1306	60	A	0
18	1985	26	1143	58	1388	53	H	0
19	1985	26	1153	50	1184	48	H	0
20	1985	26	1165	47	1159	40	A	0
21	1985	26	1171	55	1216	52	H	0
22	1985	26	1173	76	1134	56	H	0
23	1985	26	1177	59	1296	58	H	0
24	1985	26	1193	79	1265	76	A	0
25	1985	26	1196	106	1416	55	H	0

	Season	Daynum	Wteam	Wscore	Lteam	Lscore	Wloc	Numot
26	1985	26	1206	95	1137	77	H	0
27	1985	26	1210	79	1149	66	H	0
28	1985	26	1211	64	1102	59	A	0
29	1985	26	1234	76	1114	47	H	0
...	...	...	...	...	...	...	...	...
145259	2016	130	1452	69	1328	67	N	0
145260	2016	131	1114	72	1418	65	N	0
145261	2016	131	1138	64	1103	61	N	0
145262	2016	131	1163	77	1396	62	N	0
145263	2016	131	1167	57	1308	54	N	0
145264	2016	131	1201	68	1361	63	N	0
145265	2016	131	1214	81	1354	69	N	0
145266	2016	131	1218	64	1253	60	N	0
145267	2016	131	1242	81	1452	71	N	0
145268	2016	131	1246	93	1208	80	N	0
145269	2016	131	1272	74	1408	54	N	0
145270	2016	131	1277	64	1268	61	N	0
145271	2016	131	1292	55	1330	53	N	0
145272	2016	131	1314	61	1438	57	N	0
145273	2016	131	1332	88	1428	57	N	0
145274	2016	131	1345	76	1276	59	N	0
145275	2016	131	1371	69	1437	67	N	0
145276	2016	131	1372	82	1394	60	N	0
145277	2016	131	1380	54	1238	53	N	0
145278	2016	131	1386	82	1173	79	N	0
145279	2016	131	1392	80	1436	74	H	0
145280	2016	131	1401	71	1261	38	N	0

	Season	Daynum	Wteam	Wscore	Lteam	Lscore	Wloc	Numot
145281	2016	131	1419	82	1426	71	N	0
145282	2016	131	1433	76	1172	54	N	0
145283	2016	131	1451	62	1285	59	N	0
145284	2016	132	1114	70	1419	50	N	0
145285	2016	132	1163	72	1272	58	N	0
145286	2016	132	1246	82	1401	77	N	1
145287	2016	132	1277	66	1345	62	N	0
145288	2016	132	1386	87	1433	74	N	0

145289 rows × 8 columns

In [2]:
```
import pandas as pd
filePath = 'data files/RegularSeasonCompactResults.csv'
def readCsvData(filePath):
    return pd.read_csv(filePath)
df=readCsvData(filePath)
```

Group by

In [3]:
```
df.columns.tolist()
```

Out[3]:
```
['Season', 'Daynum', 'Wteam', 'Wscore', 'Lteam', 'Lscore', 'Wloc', 'Numot']
```

In [3]:
```
df.groupby('Wteam')['Wscore'].mean().head()
```

Out[3]:
```
Wteam
1101    78.111111
1102    69.893204
1103    75.839768
1104    75.825944
1105    74.960894
Name: Wscore, dtype: float64
```

In [4]:
```
df.groupby('Wteam')['Wscore'].mean().tail()
```

Out[4]:
```
Wteam
1460    75.531469
1461    75.170082
1462    79.906021
1463    71.720102
1464    73.926056
Name: Wscore, dtype: float64
```

Data cleaning

- whenever you have a lot of missing records values in the given dataset then you can isnull()
- isnull() will notify the any missing values in the dataframe

In [5]:
```
#Entire data set
df.isnull().sum()
```

Out[5]:
```
Season    0
Daynum    0
Wteam     0
Wscore    0
Lteam     0
Lscore    0
Wloc      0
Numot     0
dtype: int64
```

In [6]:
```
df['Season'].isnull().sum()
```

Out[6]:
```
0
```

- dropna() This function allows you to drop all or some rows that have missing values
- fillna() This function allows you to replace the rows that have missing values with the value what you want pass

In [7]:
```
len(df)
```

Out[7]: 145289

In [8]:
```
# Dropping rows with 1 null value
new_df = df.dropna(axis=0,how='any')
```

In [9]:
```
len(new_df)
```

Out[9]: 145289

In [10]:
```
df['Season'].fillna('Testing',inplace=True)
```

In [11]:
```
df
```

Out[11]:

	Season	Daynum	Wteam	Wscore	Lteam	Lscore	Wloc	Numot
0	1985	20	1228	81	1328	64	N	0
1	1985	25	1106	77	1354	70	H	0
2	1985	25	1112	63	1223	56	H	0
3	1985	25	1165	70	1432	54	H	0
4	1985	25	1192	86	1447	74	H	0
5	1985	25	1218	79	1337	78	H	0
6	1985	25	1228	64	1226	44	N	0
7	1985	25	1242	58	1268	56	N	0
8	1985	25	1260	98	1133	80	H	0
9	1985	25	1305	97	1424	89	H	0
10	1985	25	1307	103	1288	71	H	0
11	1985	25	1344	75	1438	71	N	0
12	1985	25	1374	91	1411	72	H	0
13	1985	25	1412	70	1397	65	N	0
14	1985	25	1417	87	1225	58	H	0
15	1985	26	1116	65	1368	62	H	0
16	1985	26	1120	92	1391	50	H	0
17	1985	26	1135	65	1306	60	A	0
18	1985	26	1143	58	1388	53	H	0
19	1985	26	1153	50	1184	48	H	0
20	1985	26	1165	47	1159	40	A	0
21	1985	26	1171	55	1216	52	H	0
22	1985	26	1173	76	1134	56	H	0
23	1985	26	1177	59	1296	58	H	0
24	1985	26	1193	79	1265	76	A	0
25	1985	26	1196	106	1416	55	H	0

	Season	Daynum	Wteam	Wscore	Lteam	Lscore	Wloc	Numot
26	1985	26	1206	95	1137	77	H	0
27	1985	26	1210	79	1149	66	H	0
28	1985	26	1211	64	1102	59	A	0
29	1985	26	1234	76	1114	47	H	0
...	...	...	...	...	...	...	...	...
145259	2016	130	1452	69	1328	67	N	0
145260	2016	131	1114	72	1418	65	N	0
145261	2016	131	1138	64	1103	61	N	0
145262	2016	131	1163	77	1396	62	N	0
145263	2016	131	1167	57	1308	54	N	0
145264	2016	131	1201	68	1361	63	N	0
145265	2016	131	1214	81	1354	69	N	0
145266	2016	131	1218	64	1253	60	N	0
145267	2016	131	1242	81	1452	71	N	0
145268	2016	131	1246	93	1208	80	N	0
145269	2016	131	1272	74	1408	54	N	0
145270	2016	131	1277	64	1268	61	N	0
145271	2016	131	1292	55	1330	53	N	0
145272	2016	131	1314	61	1438	57	N	0
145273	2016	131	1332	88	1428	57	N	0
145274	2016	131	1345	76	1276	59	N	0
145275	2016	131	1371	69	1437	67	N	0
145276	2016	131	1372	82	1394	60	N	0
145277	2016	131	1380	54	1238	53	N	0
145278	2016	131	1386	82	1173	79	N	0
145279	2016	131	1392	80	1436	74	H	0
145280	2016	131	1401	71	1261	38	N	0

	Season	Daynum	Wteam	Wscore	Lteam	Lscore	Wloc	Numot
145281	2016	131	1419	82	1426	71	N	0
145282	2016	131	1433	76	1172	54	N	0
145283	2016	131	1451	62	1285	59	N	0
145284	2016	132	1114	70	1419	50	N	0
145285	2016	132	1163	72	1272	58	N	0
145286	2016	132	1246	82	1401	77	N	1
145287	2016	132	1277	66	1345	62	N	0
145288	2016	132	1386	87	1433	74	N	0

145289 rows × 8 columns

NumPy Library

- Processing of N-dimensional arrays
- NumPy is an open source library available in python
- This used for Math,Scientific,DataScience programming

In [13]:
```
import numpy as np
li=[1,2,3,4,5]
c=np.array(li)
print(c)
print(type(c))
```

```
[1 2 3 4 5]
<class 'numpy.ndarray'>
```

In [15]:
```
print(c.shape)
print(c.dtype)
```

```
(5,)
int32
```

In [17]:
```
# 2 dimensional Representation
a=np.array([(1,2,3),(4,5,6)])
a
```

Out[17]:
```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [18]:
```
print(a.shape)
print(a.dtype)
```

```
(2, 3)
int32
```

np.zeros() and np.ones()

- to initialize the weights during the first iteration of Tensorflow and statistics steps

In [22]:
```
x=np.zeros((2,2))
print(x)
```

```
[[0. 0.]
 [0. 0.]]
```

In [23]:
```
x1=np.ones((3,3),dtype=np.int16)
print(x1)
```

```
[[1 1 1]
 [1 1 1]
 [1 1 1]]
```

In [25]:
```
x2=np.arange(15)
print(x2)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14]
```

In [26]:
```
# Convert the single dimensional to 2 dimensional array
x3=x2.reshape(5,3)
print(x3)
```

```
[[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]
 [12 13 14]]
```

In [27]:
```
x3 + 10
print(x3)
```

```
[[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]
 [12 13 14]]
```

In [28]:
```
x4=x3+10
print(x4)
```

```
[[10 11 12]
 [13 14 15]
 [16 17 18]
 [19 20 21]
 [22 23 24]]
```

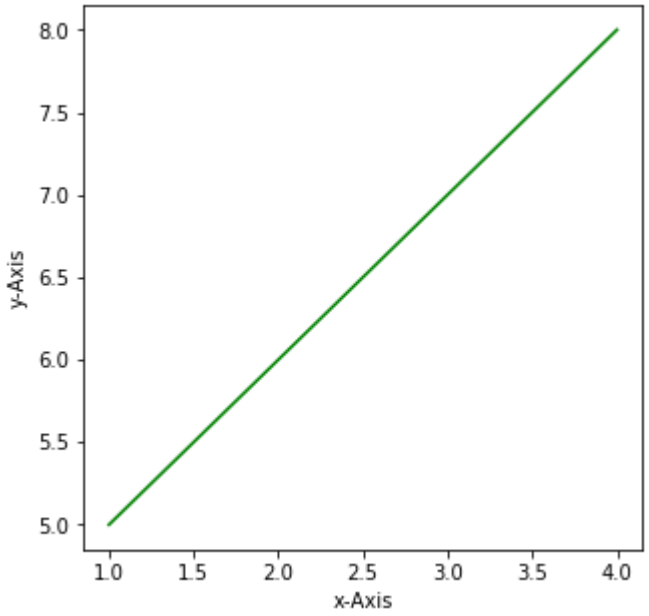
In [30]:
```
x4=x3/4
print(x4)
```

```
[[0.   0.25 0.5 ]
 [0.75 1.   1.25]
 [1.5  1.75 2.  ]
 [2.25 2.5  2.75]
 [3.   3.25 3.5 ]]
```

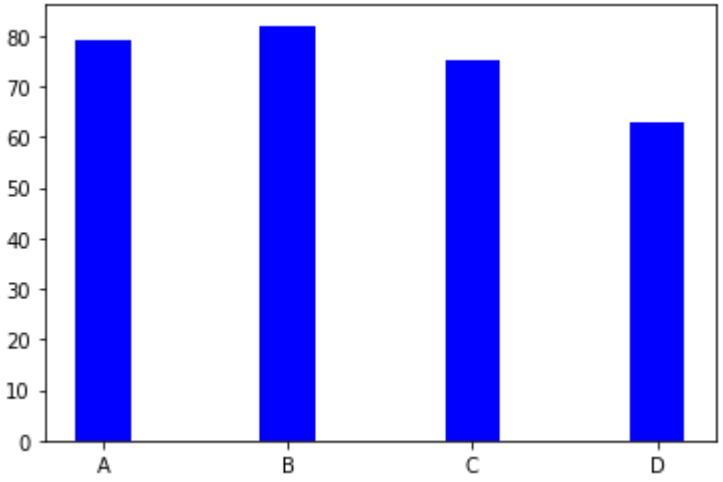
Visualizing Data

- An interesting way to display Dataframe
- With matplotlib

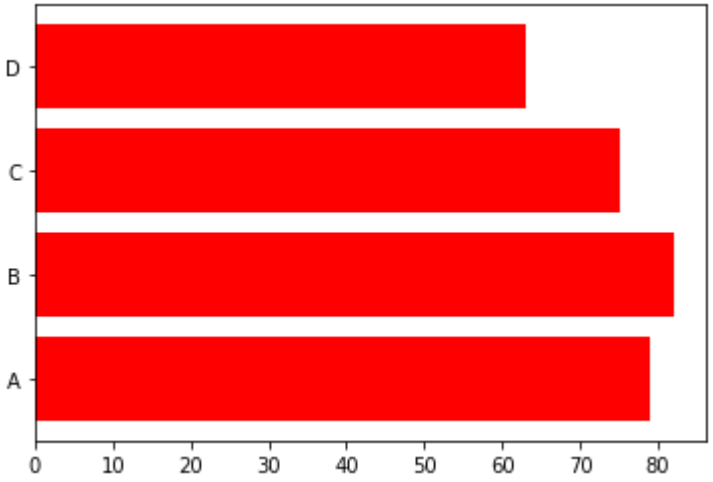
In [34]:
```
import matplotlib.pyplot as plt
a1 = [1,2,3,4]
a2 = [5,6,7,8]
plt.figure(figsize=(5,5))
plt.plot(a1,a2,color='green')
plt.xlabel('x-Axis')
plt.ylabel('y-Axis')
plt.show()
```



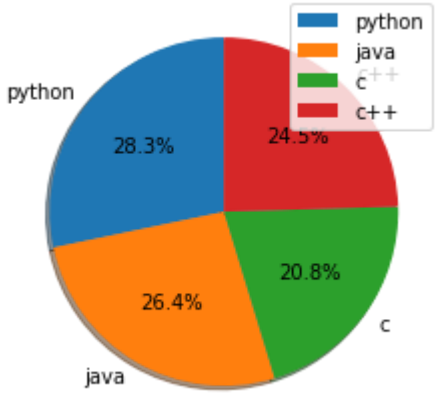
In [36]:
```
marks=[79,82,75,63]
divisions=['A','B','C','D']
avg_marks = [65,72,78,89]
width = 0.70
plt.bar(divisions,marks,color="blue",width=0.3)
plt.show()
```



In [37]:
```
plt.barh(divisions,marks,color='red')
plt.show()
```



In [39]:
```
lang = ['python','java','c','c++']
rating = [75,70,55,65]
plt.pie(rating,labels = lang,shadow=True,startangle=90,autopct='%1.1f%%')
plt.legend(loc='best')
plt.show()
```



In [ ]:
```

```