In [3]:
```
from sklearn import datasets
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.cluster import KMeans
```

In [4]:
```
#Load the data
iris  = datasets.load_iris()
iris
```

Out[4]:
```
{'DESCR': 'Iris Plants Database\n====================\n\nNotes\n-----\nData S
et Characteristics:\n    :Number of Instances: 150 (50 in each of three class
es)\n    :Number of Attributes: 4 numeric, predictive attributes and the clas
s\n    :Attribute Information:\n        - sepal length in cm\n        - sepal
width in cm\n        - petal length in cm\n        - petal width in cm\n
- class:\n                - Iris-Setosa\n                - Iris-Versicolour\n
- Iris-Virginica\n    :Summary Statistics:\n\n    ============== ==== ==== ==
===== ===== ====================\n                    Min  Max   Mean    SD
Class Correlation\n    ============== ==== ==== ======= ===== ==============
=====\n    sepal length:   4.3  7.9   5.84   0.83    0.7826\n    sepal width:
2.0  4.4   3.05   0.43   -0.4194\n    petal length:   1.0  6.9   3.76   1.76
0.9490  (high!)\n    petal width:    0.1  2.5   1.20  0.76     0.9565  (hig
h!)\n    ============== ==== ==== ======= ===== ====================\n\n    :
Missing Attribute Values: None\n    :Class Distribution: 33.3% for each of 3
classes.\n    :Creator: R.A. Fisher\n    :Donor: Michael Marshall (MARSHALL%P
LU@io.arc.nasa.gov)\n    :Date: July, 1988\n\nThis is a copy of UCI ML iris d
atasets.\nhttp://archive.ics.uci.edu/ml/datasets/Iris\n\nThe famous Iris data
base, first used by Sir R.A Fisher\n\nThis is perhaps the best known database
to be found in the\npattern recognition literature.  Fisher\'s paper is a cla
```

In [5]:
```
#type(df)
df = pd.DataFrame(iris.data)
df.head()
```

Out[5]:

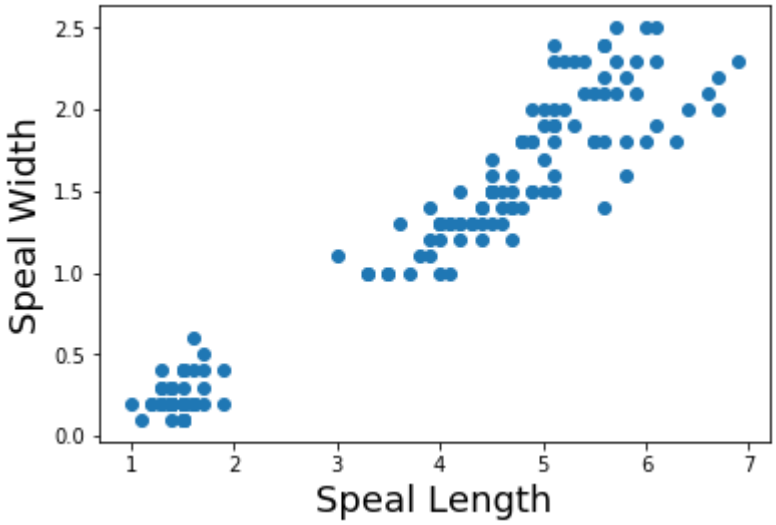
	0	1	2	3
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

In [6]:
```
x=df.iloc[:,2:4]
x.head()
```

Out[6]:

	2	3
0	1.4	0.2
1	1.4	0.2
2	1.3	0.2
3	1.5	0.2
4	1.4	0.2

In [7]:
```
plt.scatter(x.iloc[:,0],x.iloc[:,1]) #x.iloc contains the position
plt.xlabel('Speal Length',fontsize =18)
plt.ylabel('Speal Width',fontsize =18)
plt.show()
```



In [8]:
```
km = KMeans(n_clusters = 4, random_state=21)
km.fit(x) #To create the model and apply it on training data sets
```

Out[8]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=4, n_init=10, n_jobs=1, precompute_distances='auto',
    random_state=21, tol=0.0001, verbose=0)
```

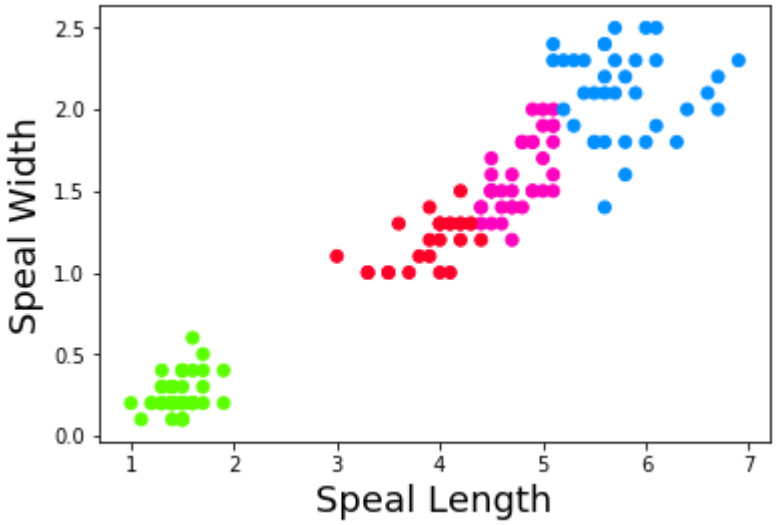
In [9]:
```
#centroid
centers=km.cluster_centers_
centers
```

Out[9]:
```
array([[3.90384615, 1.19230769],
       [1.464     , 0.244     ],
       [5.78611111, 2.10833333],
       [4.75789474, 1.59736842]])
```

In [10]:
```
z = km.labels_
#labeled it by different name using concat function pd.concat(x,z)
#Try prediction
z
```

Out[10]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 3, 3, 3, 0, 3, 3, 3, 0, 3, 0, 0, 0, 0, 3, 0, 3,
       3, 0, 3, 0, 3, 0, 3, 3, 0, 3, 3, 3, 3, 0, 0, 0, 0, 3, 3, 3, 3, 3,
       0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 2, 3, 2, 2, 2, 2, 3, 2, 2, 2,
       3, 2, 2, 3, 2, 2, 2, 2, 2, 3, 2, 3, 2, 3, 2, 2, 3, 3, 2, 2, 2, 2,
       2, 3, 2, 2, 2, 2, 3, 2, 2, 2, 3, 2, 2, 2, 3, 2, 2, 3])
```

In [11]:
```
plt.scatter(x.iloc[:,0],x.iloc[:,1], c=km.labels_ ,cmap ="gist_rainbow")
plt.xlabel('Speal Length',fontsize =18)
plt.ylabel('Speal Width',fontsize =18)
plt.show()
```



In [ ]:

In [ ]: