In [21]:
```
#Import scikit libraries
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics

#Import numpy library
import numpy as np

#Import scikit-learn dataset library
from sklearn import datasets
```

In [25]:
```
#Load dataset
iris = datasets.load_iris()
```

In [24]:
```
type(iris)
```

Out[24]:
```
sklearn.utils.Bunch
```

In [9]:
```
print(iris.feature_names)
```

```
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
```

In [10]:
```
print(iris.target)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
```

In [11]:
```
print(iris.target_names)
```

```
['setosa' 'versicolor' 'virginica']
```

In [12]:
```
# first 5 data rows
print(iris.data[0:5])
```

```
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]]
```

In [13]:
```
print(iris.data.shape)
X = iris.data
```

```
(150, 4)
```

In [14]:
```
print(iris.target.shape)
y = iris.target
```

```
(150,)
```

In [15]:
```
#Split into train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state =4) # 70% training and 30% test

# Create a k-NN classifier with 5 neighbors
knn = KNeighborsClassifier(n_neighbors=1)

# Fit the classifier to the training data
knn.fit(X, y)

#Predict the response for test dataset
y_pred = knn.predict(X_test)

print(y_pred)
```

```
[2 0 2 2 2 1 1 0 0 2 0 0 0 1 2 0 1 0 0 2 0 2 1 0 0 0 0 0 0 2 1 0 2 0 1 2 2
 1 1 0 2 0 1 0 2 0 0 1 1 2 0 1 2 2 1 1 0 1 2 1]
```

In [16]:
```
print(X_train.shape)
print(y_train.shape)
```

```
(90, 4)
(90,)
```

In [17]:
```
#Calculating accuracy of algorithm
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 1.0
```

In [148]:
```
knn.predict(np.array([[3,5,4,2]]))
```

Out[148]:
```
array([2])
```

In [149]:
```
X_new = [[3,5,4,2],[5,4,3,2]]
knn.predict(X_new)
```

Out[149]:
```
array([2, 1])
```

In [165]:
```
knn = KNeighborsClassifier(n_neighbors = 5)
knn.fit(X_train,y_train)
y_pred = knn.predict(X_test)
print (metrics.accuracy_score(y_test, y_pred))
```

```
0.9666666666666667
```

In [164]:
```
knn = KNeighborsClassifier(n_neighbors = 1)
knn.fit(X_train,y_train)
y_pred = knn.predict(X_test)
print (metrics.accuracy_score(y_test, y_pred))
```

```
0.95
```

In [167]:
```
#Finding a the best value for K
#For to to calculate K from 1 through 25 and record testing accuracy

k_range = range(1,25)
scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors = k)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    scores.append(metrics.accuracy_score(y_test,y_pred))
```

In [168]:
```
# Plot relationship between K and testing accuracy

import matplotlib.pyplot as plt

%matplotlib inline

plt.plot(k_range, scores)
plt.xlabel('Value of K for KNN')
plt.ylabel('Testing Accuracy')
```

Out[168]:
```
Text(0,0.5,'Testing Accuracy')
```