In [26]:
```
#Setting the working Directory
import os
path="/Users/amanmahajan/Desktop/ML/Assignment1"
os.chdir(path)

#Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

#Importing the dataset "Iris"
mydata = pd.read_csv('Iris.csv')
x = mydata.iloc[:, [1,2,3,4]].values
y = mydata.iloc[:, 5].values

#Splitting the dataset into the Training set and Test set
from sklearn.cross_validation import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size =
0.25, random_state = 0)

#Scaling
from sklearn.preprocessing import StandardScaler
scale = StandardScaler()
x_train = scale.fit_transform(x_train)
x_test = scale.transform(x_test)

#Fitting K-NN to the Training set
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski
', p = 2)
classifier.fit(x_train, y_train)

#Predicting the Test set results
y_pred = classifier.predict(x_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
conMat = confusion_matrix(y_test, y_pred)
print(conMat)

#Calculating the accuracy
accuracy = ((13+15+9)*100)/(13+15+9+1)
print("Percentage of Confusion matrix accuracy is:")
print(accuracy)
```

```
[[13  0  0]
 [ 0 15  1]
 [ 0  0  9]]
Percentage of Confusion matrix accuracy is:
97.36842105263158
```

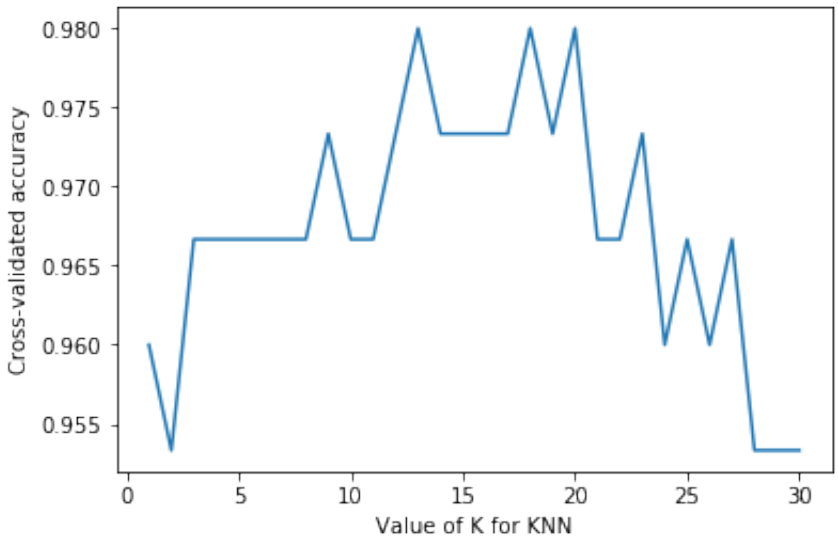
In [44]:
```
#Finding Best Value for K
from sklearn.cross_validation import cross_val_score
knn_opt = range(1, 31)

#Empty list to store scores
knn_store = []

#Looping through reasonable values of k
for k in knn_opt:
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, x, y, cv=10, scoring='accuracy')
    knn_store.append(scores.mean())


print(knn_store)
```

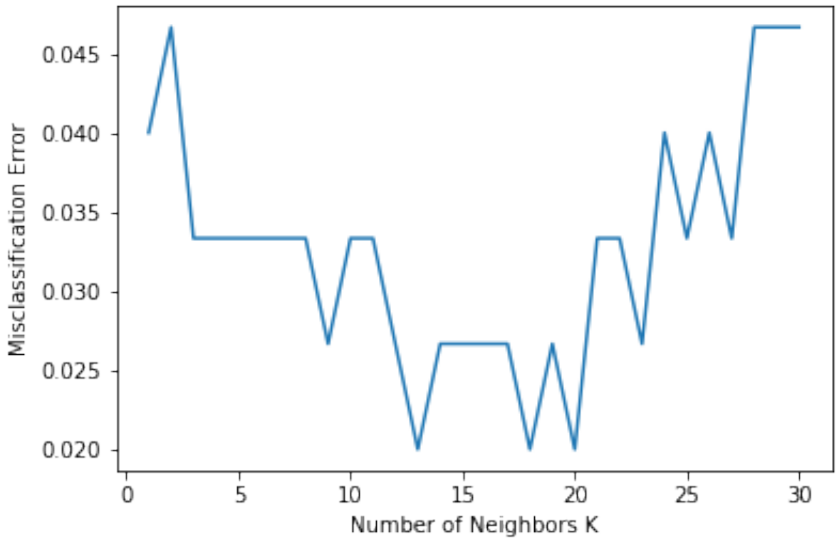
```
[0.95999999999999996, 0.95333333333333337, 0.96666666666666656, 0.96
666666666666656, 0.96666666666666679, 0.96666666666666679, 0.9666666
6666666679, 0.96666666666666679, 0.97333333333333338, 0.966666666666
66679, 0.96666666666666679, 0.97333333333333338, 0.98000000000000009
, 0.97333333333333338, 0.97333333333333338, 0.97333333333333338, 0.9
7333333333333338, 0.98000000000000009, 0.97333333333333338, 0.980000
00000000009, 0.96666666666666656, 0.96666666666666656, 0.97333333333
333338, 0.95999999999999996, 0.96666666666666656, 0.9599999999999999
6, 0.96666666666666656, 0.95333333333333337, 0.95333333333333337, 0.
95333333333333337]
```

In [45]:
```
print('Length of list', len(knn_store))
print('Max of list', max(knn_store))
```

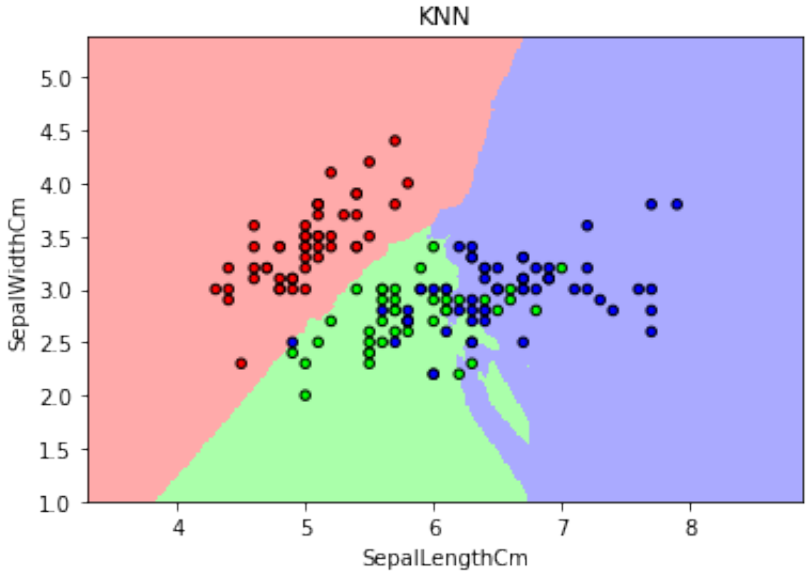
```
Length of list 30
Max of list 0.98
```

In [56]:
```
import matplotlib.pyplot as plt
%matplotlib inline

# plot the value of K for KNN (x-axis) versus the cross-validated accu
racy (y-axis)
plt.plot(knn_opt, knn_store)
plt.xlabel('Value of K for KNN')
plt.ylabel('Cross-validated accuracy')
plt.show()

#Finding the Misclassification Error MSE
MSE = [1 - x for x in knn_store]
MSE

#Plotting the values of K against MSE
plt.plot(knn_opt, MSE)
plt.xlabel('Number of Neighbors K')
plt.ylabel('Misclassification Error')
plt.show()
```


0.980
0.975
0.970
0.965
0.960
0.955
Cross-validated accuracy
0
5
10
15
20
25
30
Value of K for KNN


0.045
0.040
0.035
0.030
0.025
0.020
Misclassification Error
0
5
10
15
20
25
30
Number of Neighbors K

In [54]:
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn import neighbors, datasets

n_neighbors = 20

iris = datasets.load_iris()

x = iris.data[:, :2]
y = iris.target

#Defining Step Size
h = .02

#Creating color maps
cmap_light = ListedColormap(['#FFAAAA', '#AAFFAA', '#AAAAFF'])
cmap_bold = ListedColormap(['#FF0000', '#00FF00', '#0000FF'])

for weights in ['uniform']:
    # Creating an instance of Neighbours Classifier and fitting the da
ta.
    clf = neighbors.KNeighborsClassifier(n_neighbors, weights=weights)
    clf.fit(x, y)

    # Plotting the decision boundary.
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                         np.arange(y_min, y_max, h))
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])

    # Putting the result into a color plot
    Z = Z.reshape(xx.shape)
    plt.figure()
    plt.pcolormesh(xx, yy, Z, cmap=cmap_light)

    # Plotting the training points
    plt.scatter(X[:, 0], X[:, 1], c=y, cmap=cmap_bold,
                edgecolor='k', s=20)
    plt.xlim(xx.min(), xx.max())
    plt.ylim(yy.min(), yy.max())
    plt.title("KNN")
    plt.xlabel("SepalLengthCm")
    plt.ylabel("SepalWidthCm")

plt.show()
```



In [23]:
```
#Conclusion

#On performing the Cross Validation method with CV=10 we see the follo
wing results-
#Percentage of Confusion Matrix Accuracy is 97.36%
#Length of the values of KNN classifiers is 30 and the maximum value o
f accuracy is 0.98 within this range.
#Through the graph, after implying 10-fold cross validation, we see th
at the optimal value for K is 20.
#We confirm this after calculating the MSE error which comes out to be
the least for K=20.
#We also show the visual graphical representation between SepalLenghtC
m and SepalWidthCM.
```