K-Means using scikit-learn

Use the scikit k-Means implementation to build the cluster of the data frame.

Preparations

Create the same data frame as above so that it is fresh.

In [4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import copy
import sklearn as sk
from sklearn.cluster import KMeans
# to check the time of execution, import function time
import time
# check versions of libraries
print('pandas version is: {}'.format(pd.__version__))
print('numpy version is: {}'.format(np.__version__))
print('sklearn version is: {}'.format(sk.__version__))
```

```
pandas version is: 1.0.1
numpy version is: 1.18.1
sklearn version is: 0.22.1
```

In [5]:
```
# Dataset
df = pd.DataFrame({
    'x': [1, 2, 4, 5, 6, 8, 3, 7],
    'y': [1, 1, 3, 4, 7, 8, 3, 7]
})

# Check that the definition of dataset is OK
print ("*** data frame ***")
print ("First column = No.")
print (df)

```

```
*** data frame ***
First column = No.
   x  y
0  1  1
1  2  1
2  4  3
3  5  4
4  6  7
5  8  8
6  3  3
7  7  7
```

K-Means training

Invoke the imported k-Means constructor with the number of clusters (here 3). Then train the model with the dataset.

In [6]:
```
# invoke constuctor
kmeans = KMeans(n_clusters=3)

# Fitting K-Means modell
print(kmeans.fit(df))
```

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

K-Means prediction

Use the model to calculate a prediction for the same data frame. Each datapoint will be labeled for the chosen cluster.

In [9]:
```
# create label for each datapoint in dataframe
labels = kmeans.predict(df)

# save centroids of each cluster
centroids = kmeans.cluster_centers_
```

Display the result

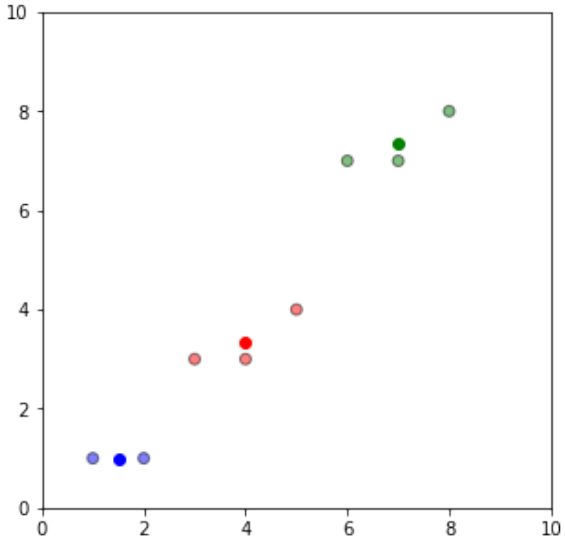
Display the positions of the centroids and the data frame. The color depends of the assigned label for each datapoint.

In [10]:
```
# Display result
fig = plt.figure(figsize=(5, 5))

# set color for each datapoint
colmap = {1: 'b', 2: 'g', 3: 'r'}
colors = list( map(lambda x: colmap[x+1], labels))

# draw each datapoint
plt.scatter(df['x'], df['y'],color=colors, alpha=0.5, edgecolor='k')

# draw each centroid
for idx, centroid in enumerate(centroids):
    plt.scatter(*centroid, color=colmap[idx+1])
plt.xlim(0, 10)
plt.ylim(0, 10)
plt.show()
```



In [15]:

```
1
2 # print current date and time
3 print("date & time:",time.strftime("%d.%m.%Y %H:%M:%S"))
4 print ("*** End of Homework-H3.4_k-Means_Clustering ***")
5
```

```
date & time: 24.05.2023 22:31:59
*** End of Homework-H3.4_k-Means_Clustering ***
```