In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import datasets
import warnings
from sklearn.cluster import KMeans
```

In [2]:
```
iris = datasets.load_iris()
iris_df = pd.DataFrame(iris.data, columns = iris.feature_names)
iris_df
```

Out[2]:

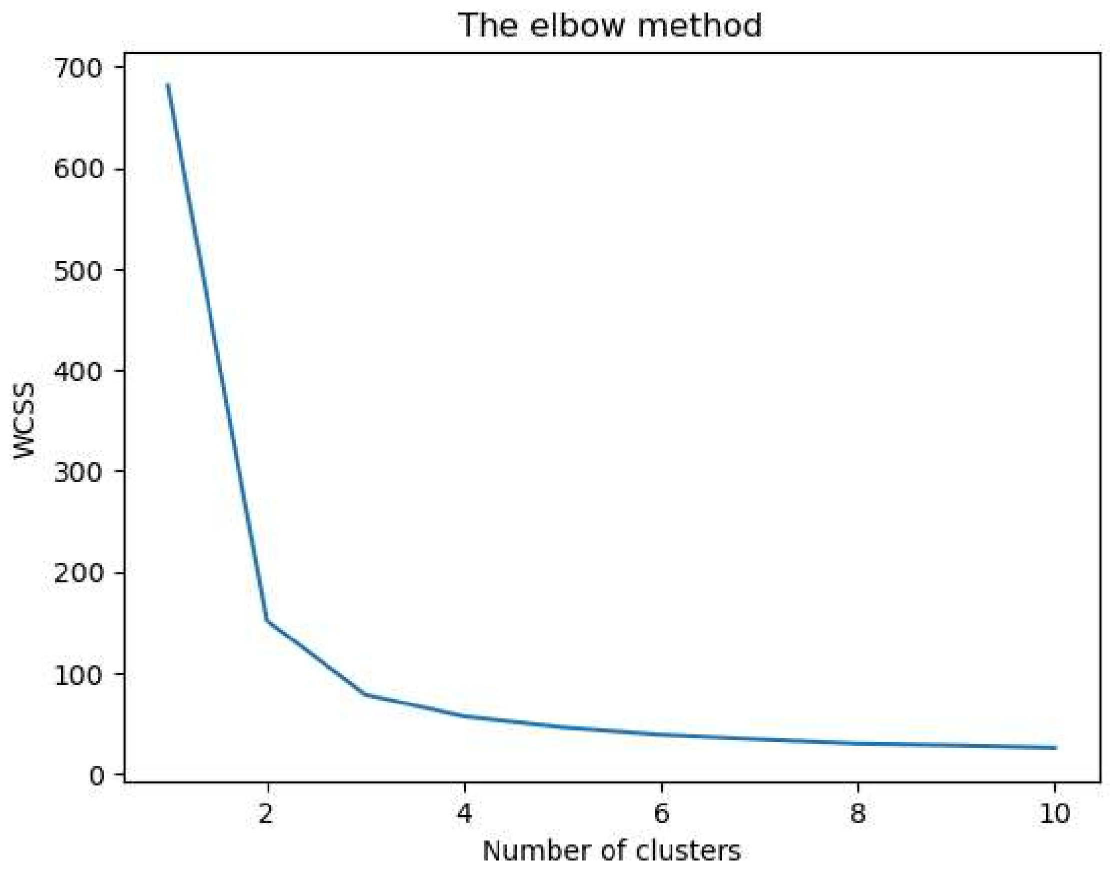
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

In [3]:
```
x = iris_df.iloc[:, [0, 1, 2, 3]].values
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter = 300, n_
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)
plt.title('The elbow method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```

```
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```



In [4]:

```
kmeans = KMeans(n_clusters = 3, init = 'k-means++', max_iter = 300, n_init
y_kmeans = kmeans.fit_predict(x)
```

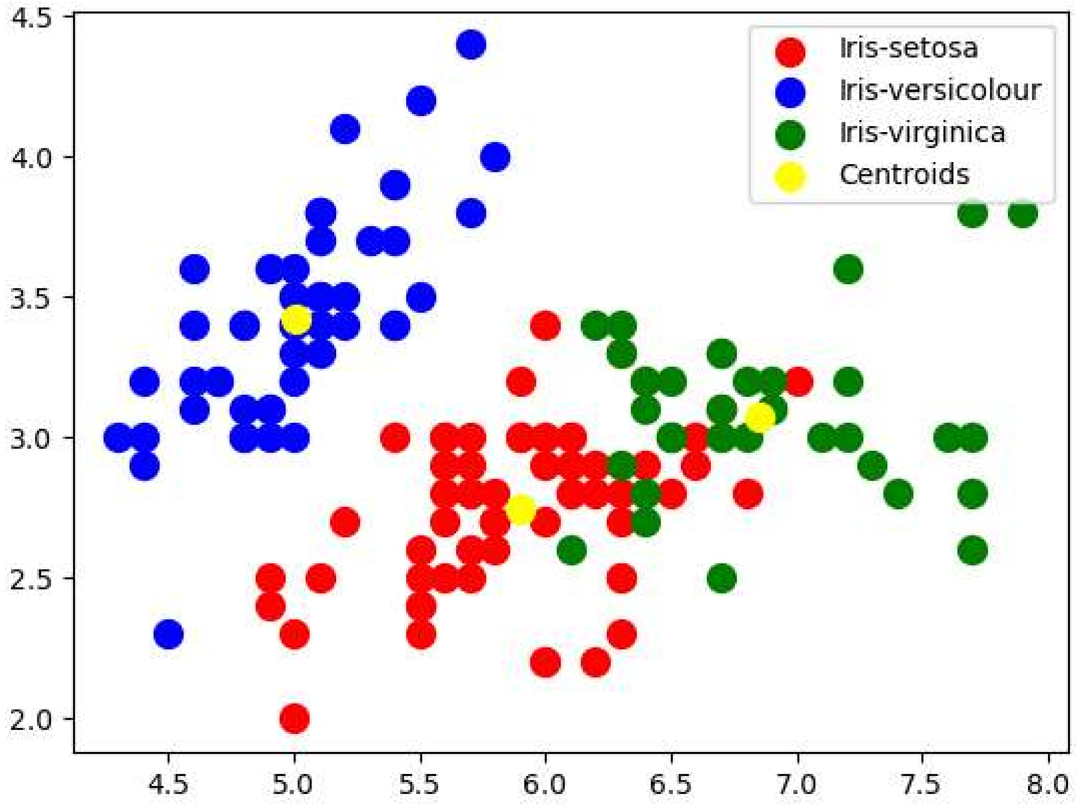
```
C:\Users\prane\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:14
36: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

In [5]:
```
plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'red',
plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'blue',
plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green'

plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:,1], s

plt.legend()
```

Out[5]:
```
<matplotlib.legend.Legend at 0x201e4e072d0>
```



In [ ]: