In [1]:
```
import numpy as np
```

In [2]:
```
import matplotlib.pyplot as plt
```

In [3]:
```
import pandas as pd
```

In [4]:
```
dataset=pd.read_csv("G:\College\BE\Data Mining\Assignments\Mall_Customers.csv")
```

In [5]:
```
x=dataset.iloc[:,[3,4]].values
```

In [6]:
```
dataset
```

Out[6]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
5	6	Female	22	17	76
6	7	Female	35	18	6
7	8	Female	23	18	94
8	9	Male	64	19	3
9	10	Female	30	19	72
10	11	Male	67	19	14
11	12	Female	35	19	99
12	13	Female	58	20	15
13	14	Female	24	20	77
14	15	Male	37	20	13
15	16	Male	22	20	79
16	17	Female	35	21	35
17	18	Male	20	21	66
18	19	Male	52	23	29
19	20	Female	35	23	98
20	21	Male	35	24	35
21	22	Male	25	24	73
22	23	Female	46	25	5
23	24	Male	31	25	73
24	25	Female	54	28	14
25	26	Male	29	28	82
26	27	Female	45	28	32
27	28	Male	35	28	61
28	29	Female	40	29	31
29	30	Female	23	29	87
...	...	...	...	...	...
170	171	Male	40	87	13
171	172	Male	28	87	75
172	173	Male	36	87	10
173	174	Male	36	87	92

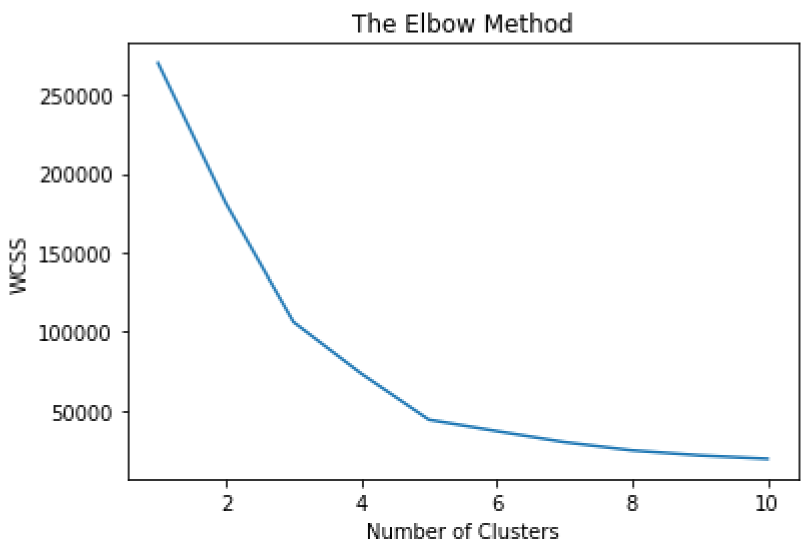
	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
174	175	Female	52	88	13
175	176	Female	30	88	86
176	177	Male	58	88	15
177	178	Male	27	88	69
178	179	Male	59	93	14
179	180	Male	35	93	90
180	181	Female	37	97	32
181	182	Female	32	97	86
182	183	Male	46	98	15
183	184	Female	29	98	88
184	185	Female	41	99	39
185	186	Male	30	99	97
186	187	Female	54	101	24
187	188	Male	28	101	68
188	189	Female	41	103	17
189	190	Female	36	103	85
190	191	Female	34	103	23
191	192	Female	32	103	69
192	193	Male	33	113	8
193	194	Female	38	113	91
194	195	Female	47	120	16
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

200 rows × 5 columns

In [7]:
```
# Using the elbow method to find the optimal number of clusters
from sklearn.cluster import KMeans
```

In [8]:
```
wcss=[]
```

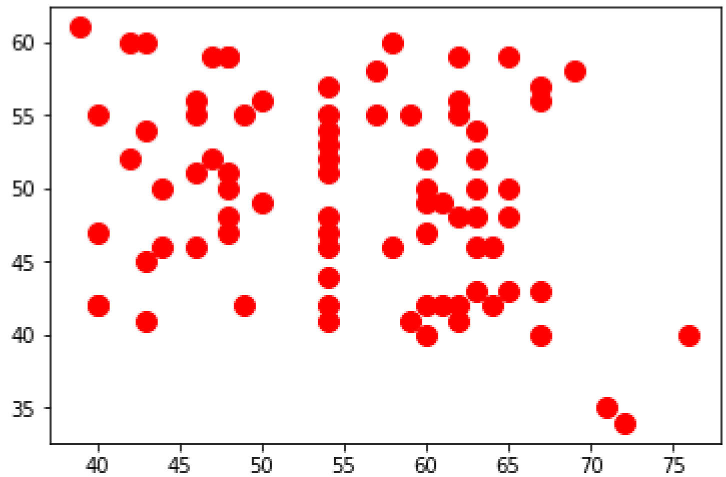
In [9]:
```
for i in range (1,11):
    kmeans= KMeans(n_clusters=i, init='k-means++', random_state= 42)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,11),wcss)
plt.title("The Elbow Method")
plt.xlabel("Number of Clusters")
plt.ylabel("WCSS")
plt.show()
```



In [10]:
```
# Fitting K-Means to the dataset
kmeans=KMeans(n_clusters=5, init='k-means++', random_state= 42)
y_kmeans= kmeans.fit_predict(x)
```

In [11]:
```
plt.scatter ( x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'red', label
```

Out[11]:
```
<matplotlib.collections.PathCollection at 0x2731dbabdd8>
```



In [12]:
```
# Visualising the clusters
plt.scatter( x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'red', label
plt.scatter( x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'blue', labe
plt.scatter( x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green', lab
plt.scatter( x[y_kmeans == 3, 0], x[y_kmeans == 3, 1], s = 100, c = 'cyan', labe
plt.scatter( x[y_kmeans == 4, 0], x[y_kmeans == 4, 1], s = 100, c = 'magenta', l
plt.scatter( kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 3
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```

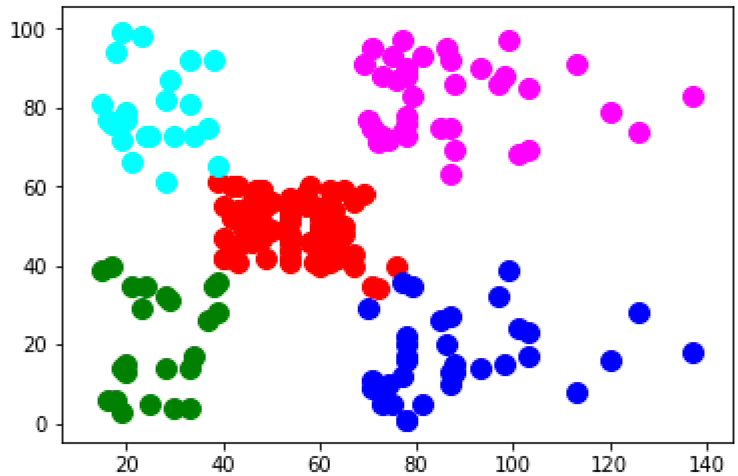
```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-12-bafefef84330> in <module>
      5 plt.scatter( x[y_kmeans == 3, 0], x[y_kmeans == 3, 1], s = 100, c = 'cy
an', label = "Cluster" )
      6 plt.scatter( x[y_kmeans == 4, 0], x[y_kmeans == 4, 1], s = 100, c = 'ma
genta', label = "Cluster" )
----> 7 plt.scatter( kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:,
1], s = 300, c = 5 )
      8 plt.title('Clusters of customers')
      9 plt.xlabel('Annual Income (k$)')

~\Anaconda3\lib\site-packages\matplotlib\pyplot.py in scatter(x, y, s, c, marke
r, cmap, norm, vmin, vmax, alpha, linewidths, verts, edgecolors, data, **kwarg
s)
   2860         vmin=vmin, vmax=vmax, alpha=alpha, linewidths=linewidths,
   2861         verts=verts, edgecolors=edgecolors, **({"data": data} if data
-> 2862         is not None else {}), **kwargs)
   2863     sci(__ret)
   2864     return __ret

~\Anaconda3\lib\site-packages\matplotlib\__init__.py in inner(ax, data, *args,
 **kwargs)
   1808                         "the Matplotlib list!)" % (label_namer, func.__
name__),
   1809                         RuntimeWarning, stacklevel=2)
-> 1810             return func(ax, *args, **kwargs)
   1811
   1812         inner.__doc__ = _add_data_doc(inner.__doc__,

~\Anaconda3\lib\site-packages\matplotlib\axes\_axes.py in scatter(self, x, y,
 s, c, marker, cmap, norm, vmin, vmax, alpha, linewidths, verts, edgecolors, **
kwargs)
   4209             try:  # First, does 'c' look suitable for value-mapping?
   4210                 c_array = np.asanyarray(c, dtype=float)
-> 4211                 n_elem = c_array.shape[0]
   4212                 if c_array.shape in xy_shape:
   4213                     c = np.ma.ravel(c_array)

IndexError: tuple index out of range
```



In [ ]:
```
plt.scatter( x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'red', label
plt.scatter( x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'blue', label
plt.scatter( x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green', lab
plt.scatter( x[y_kmeans == 3, 0], x[y_kmeans == 3, 1], s = 100, c = 'cyan', label
plt.scatter( x[y_kmeans == 4, 0], x[y_kmeans == 4, 1], s = 100, c = 'magenta', la
```

In [ ]:
```
# Visualising the clusters
plt.scatter( x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'red', label
plt.scatter( x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'blue', label
plt.scatter( x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green', lab
plt.scatter( x[y_kmeans == 3, 0], x[y_kmeans == 3, 1], s = 100, c = 'cyan', label
plt.scatter( x[y_kmeans == 4, 0], x[y_kmeans == 4, 1], s = 100, c = 'magenta', la
plt.scatter( kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 3
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```

In [ ]:
```
# Using the dendrogram to find the optimal number of clusters
import scipy.cluster.hierarchy as sch
dendrogram = sch.dendrogram(sch.linkage(x, method = 'ward'))
plt.title('Dendrogram')
plt.xlabel('Customers')
plt.ylabel('Euclidean distances')
plt.show()
```

In [ ]:
```
# Fitting Hierarchical Clustering to the dataset
from sklearn.cluster import AgglomerativeClustering
hc = AgglomerativeClustering(n_clusters = 5, affinity = 'euclidean', linkage = '
y_hc = hc.fit_predict(x)
```

In [ ]:

```
# Visualising the clusters
plt.scatter(x[y_hc == 0, 0], x[y_hc == 0, 1], s = 100, c = 'red', label = 'Clust
plt.scatter(x[y_hc == 1, 0], x[y_hc == 1, 1], s = 100, c = 'blue', label = 'Clus
plt.scatter(x[y_hc == 2, 0], x[y_hc == 2, 1], s = 100, c = 'green', label = 'Clu
plt.scatter(x[y_hc == 3, 0], x[y_hc == 3, 1], s = 100, c = 'cyan', label = 'Clus
plt.scatter(x[y_hc == 4, 0], x[y_hc == 4, 1], s = 100, c = 'magenta', label = 'C
#plt.scatter(km.cluster_centers_[:,0], km.cluster_centers_[:, 1], s = 200, c = '
plt.style.use('fivethirtyeight')
plt.title('Hierarchial Clustering', fontsize = 15)
plt.xlabel('Annual Income')
plt.ylabel('Spending Score')
plt.legend()
plt.grid()
plt.show()
```

In [ ]: