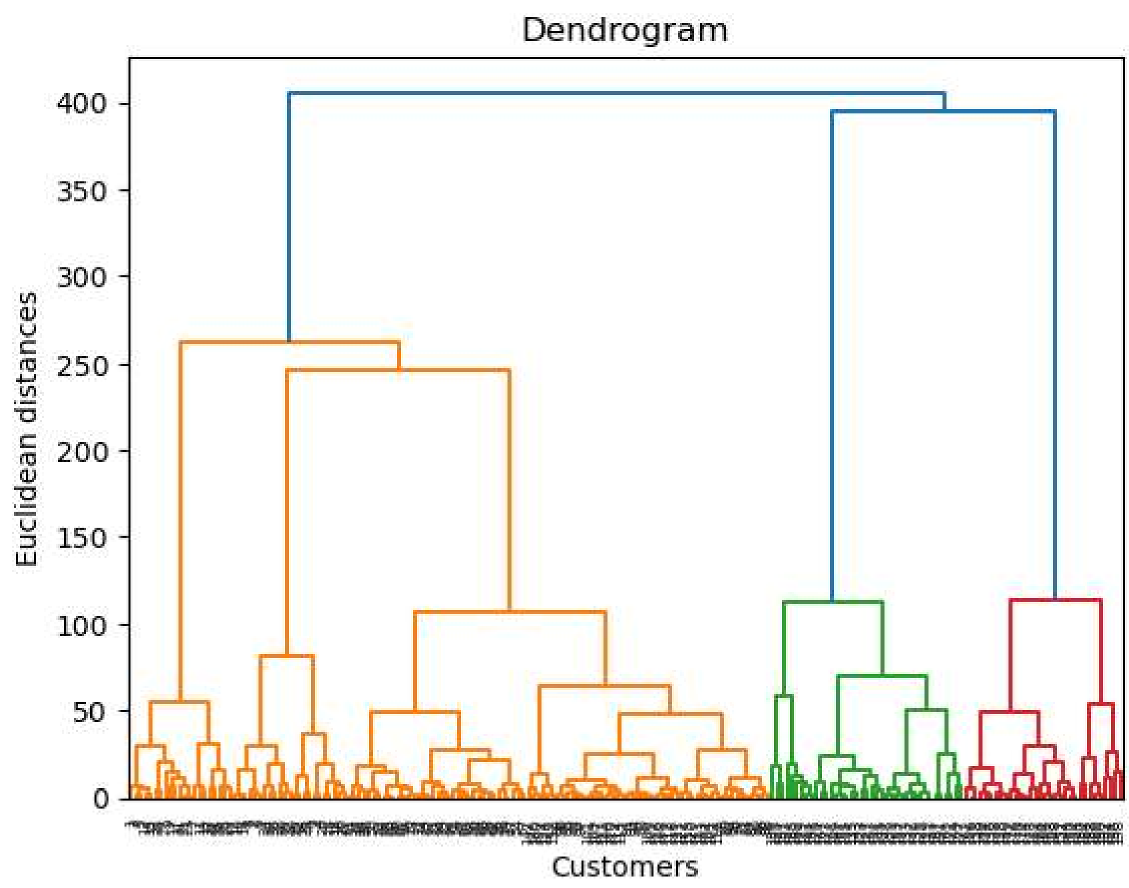
Hierachical Clustering(Using Dendogram) ,Agglomerative Clustering

In [1]:
```
# Importing Libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [2]:
```
dataset = pd.read_csv(r'C:\Users\HP\Downloads\Machine Learning\7 July, Hierarchical_clustering,K_means_cluster
x = dataset.iloc[:, [3,4]].values
```

Using the Dendogram to findout the optimal number of clusters

In [3]:
```
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()
```

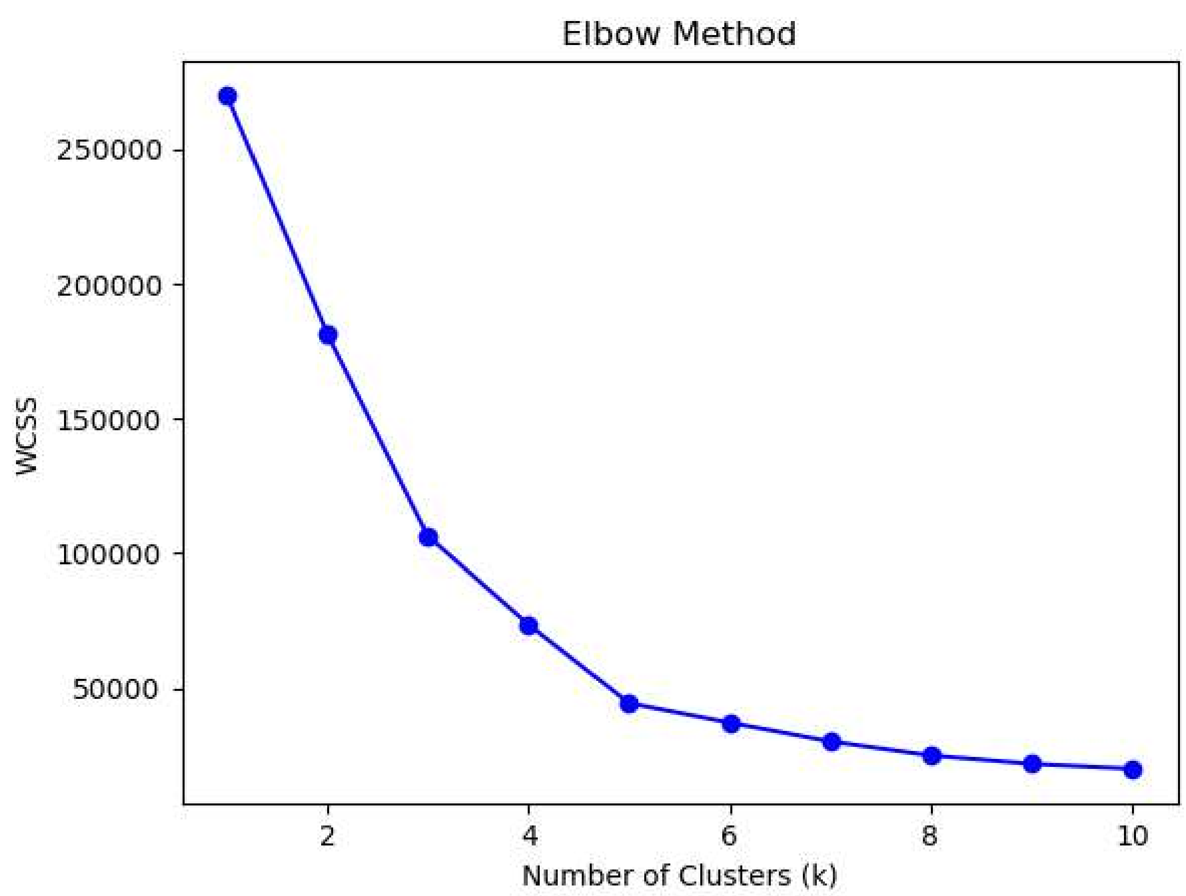


In [6]:
```
# Use of elbow method--- To determine cut level or trushold, we will implement elbow method
#Elbow Method:-1- To calculate a measure of dissimilarity(e.g. , within-cluster sum of squares,
               #(WCSS) is  using for different cut levels.
             # 2-Plot the dissimilarity measure against the number of clusters.
              #3-Look for a point where the rate of decrease in dissimilarity slows down significantly(forming
             # 4-This point can be considered as a potential cut level.
```

In [7]:
```
import warnings
# Ignore all Warnings:
warnings.filterwarnings("ignore")
from sklearn.cluster import KMeans
# Assuming that you have stored data in 'x'
# x should be a 2D array or matrix with shape (n_samples, n_features)
# Initialize an empty list to store the WCSS values for different numbers of clusters
wcss = []
# Define the range of cluster numbers from for try
k_values = range (1,11) # Try cluster numbers from 1 to 10

# Calculate WCSS for each cluster number
for k in k_values:
    kmeans = KMeans(n_clusters = k, random_state = 42)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_) # Inertia is the WCSS value

# To Plot the WCSS values against the number of clusters:
plt.title('Elbow Method')
plt.plot(k_values,wcss, 'bo-')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('WCSS')
plt.show()
```

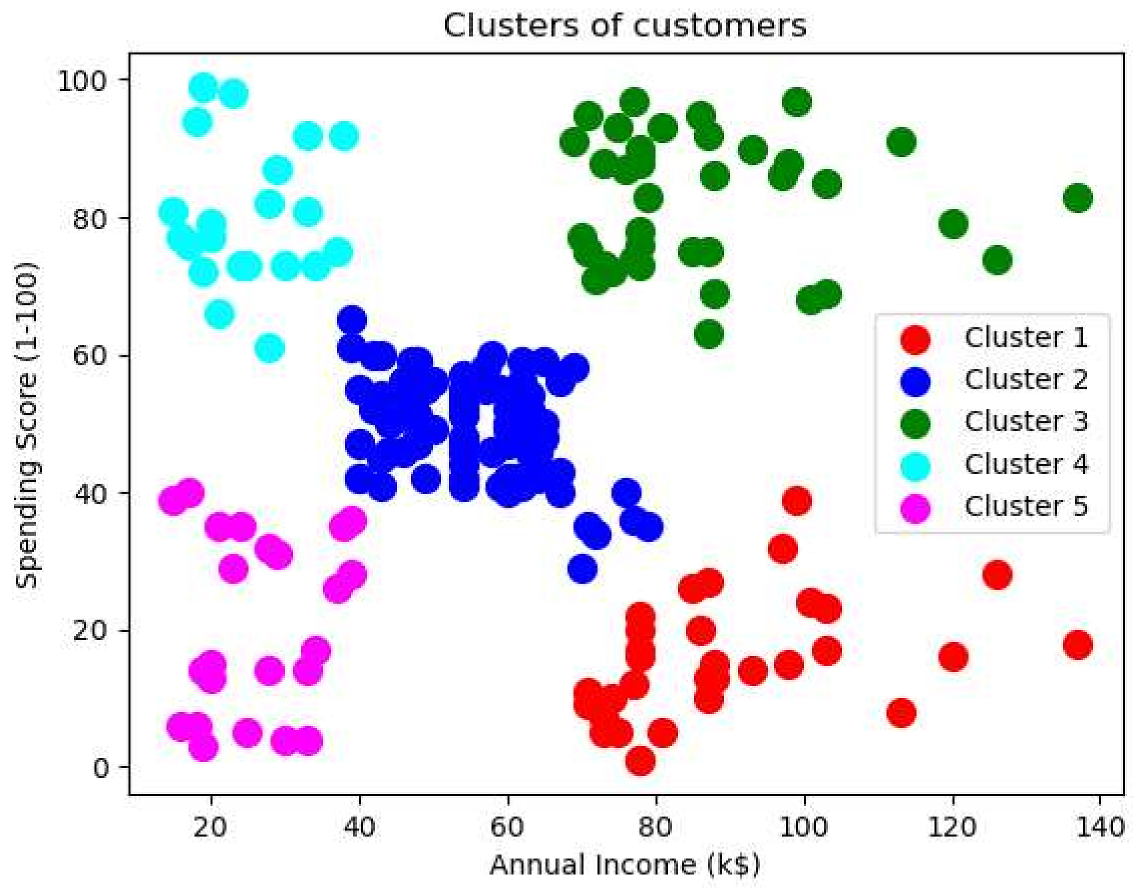


In [8]:
```
# Note: In this plot we are assuming that cluster number is 5
```

Hierarchical Clustering model and Agglomerative Clustering for Training data

In [11]:
```
# Using aggluramative clustering for Training data:
from sklearn.cluster import AgglomerativeClustering
hc = AgglomerativeClustering(n_clusters = 5, affinity = 'euclidean', linkage = 'ward')
y_hc = hc.fit_predict(x)
```

In [12]:
```
# Visualizing the clusters:
plt.scatter(x[y_hc == 0, 0], x[y_hc == 0,  1],  s = 100, c = 'red',  label = 'Cluster 1 ')
plt.scatter(x[y_hc == 1, 0], x[y_hc == 1,  1],  s = 100, c = 'blue',  label = 'Cluster 2')
plt.scatter(x[y_hc == 2, 0], x[y_hc == 2,  1],  s = 100, c = 'green', label = 'Cluster 3')
plt.scatter(x[y_hc == 3, 0], x[y_hc == 3,  1],  s = 100, c = 'cyan',   label = 'Cluster 4')
plt.scatter(x[y_hc == 4, 0], x[y_hc == 4,  1],  s = 100, c = 'magenta', label = 'Cluster 5')
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```



In [13]:
```
dataset.head()
```

Out[13]:

	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

In [14]:
```
dataset = pd.read_csv(r'C:\Users\HP\Downloads\Machine Learning\7 July, Hierarchical_clustering,K_means_cluster
x = dataset.iloc[:, [3,4]].values
```

In [15]:
```
dataset['agguluramative'] = y_hc
dataset.head()
```

Out[15]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)	agguluramative
0	1	Male	19	15	39	4
1	2	Male	21	15	81	3
2	3	Female	20	16	6	4
3	4	Female	23	16	77	3
4	5	Female	31	17	40	4

In [ ]:

In [ ]: