USE CLUSTERING TECHNIQUE FOR ANY CUSTOMER DATASET USING MACHINE LEARNING

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

In [2]:
```
data = pd.read_csv("Mall_Customers.csv")
data.head()
```

Out[2]:

	CustomerID	Gender	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 [3]:
```
data.columns
```

Out[3]:
```
Index(['CustomerID', 'Gender', 'Age', 'Annual Income (k$)',
       'Spending Score (1-100)'],
      dtype='object')
```

In [4]:
```
data.info()
```

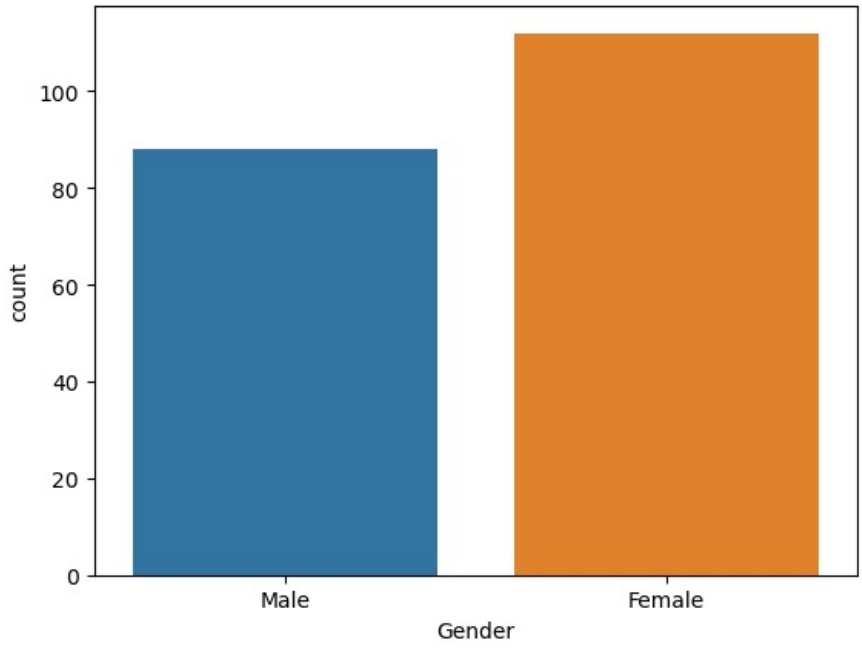
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   CustomerID              200 non-null    int64
 1   Gender                  200 non-null    object
 2   Age                     200 non-null    int64
 3   Annual Income (k$)      200 non-null    int64
 4   Spending Score (1-100)  200 non-null    int64
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
```

In [5]:
```
data.isnull().sum()
```

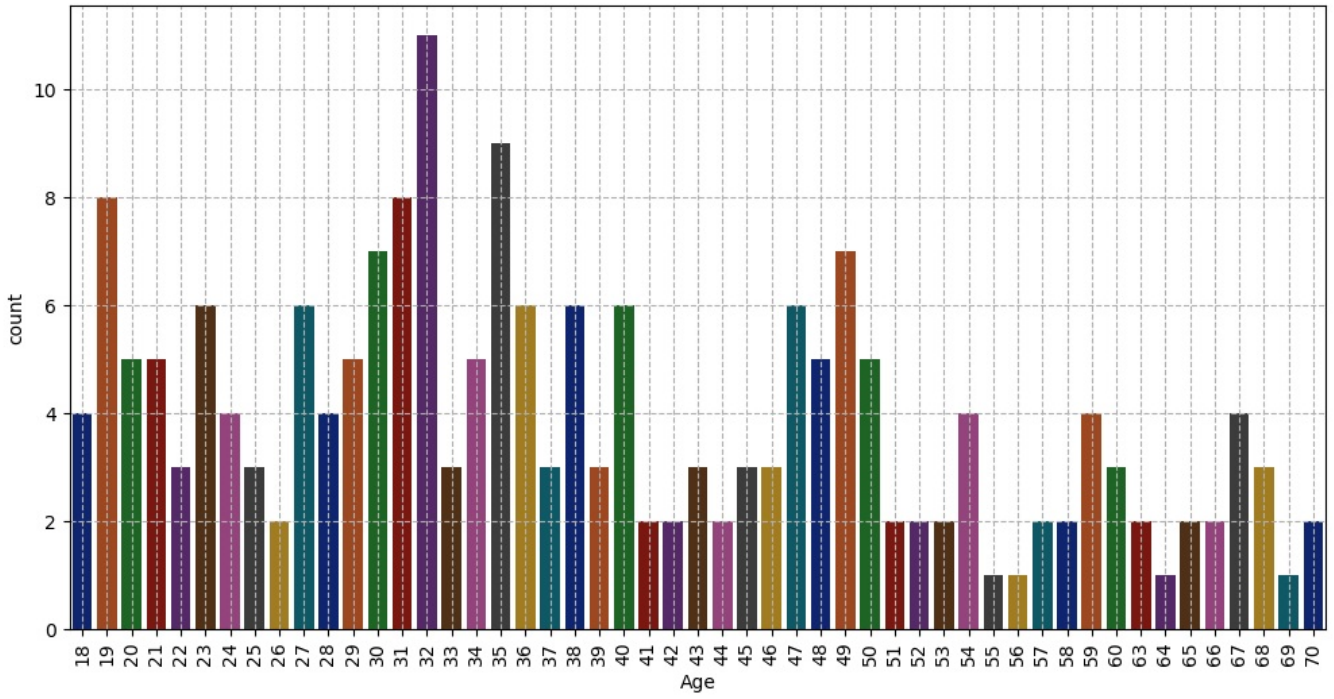
Out[5]:
```
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

In [6]:
```
sns.countplot(x='Gender' , data=data)
```

Out[6]:
```
<Axes: xlabel='Gender', ylabel='count'>
```



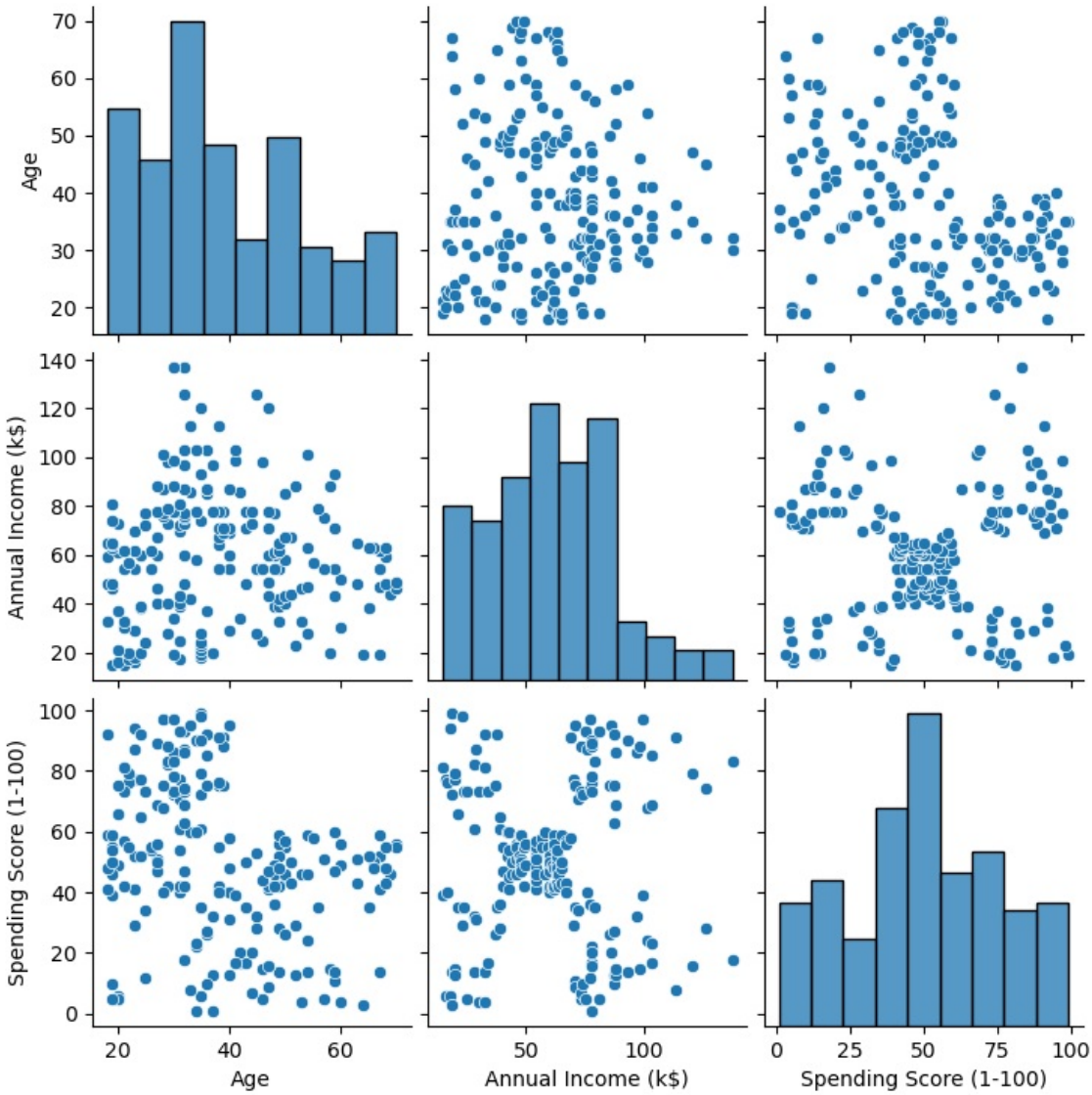
In [7]:
```
plt.figure(figsize=(12,6))
sns.countplot(x='Age', data=data, palette='dark')
plt.grid(linestyle='--')
plt.xticks(rotation=90)
plt.show()
```



In [8]:
```
data =data.drop('CustomerID', axis=1)
```

In [9]:
```
plt.figure(figsize=(10,7))
sns.pairplot(data)
plt.show()
```

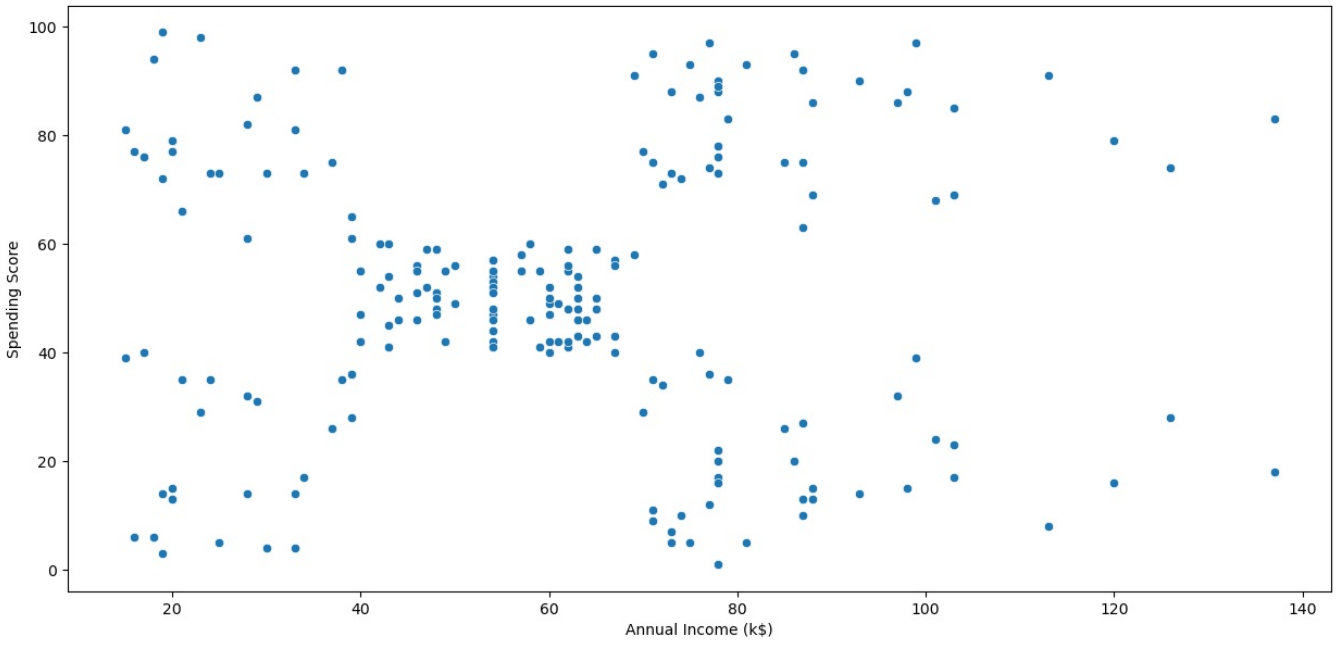
```
<Figure size 1000x700 with 0 Axes>
```



In [10]:
```
x = data.drop(columns=['Gender', 'Age'], axis=1).values
```

visualize the data points

In [11]:
```
plt.figure(figsize=(15,7))
sns.scatterplot(x=x[:,0], y=x[:, 1])
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score')
plt.show()
```

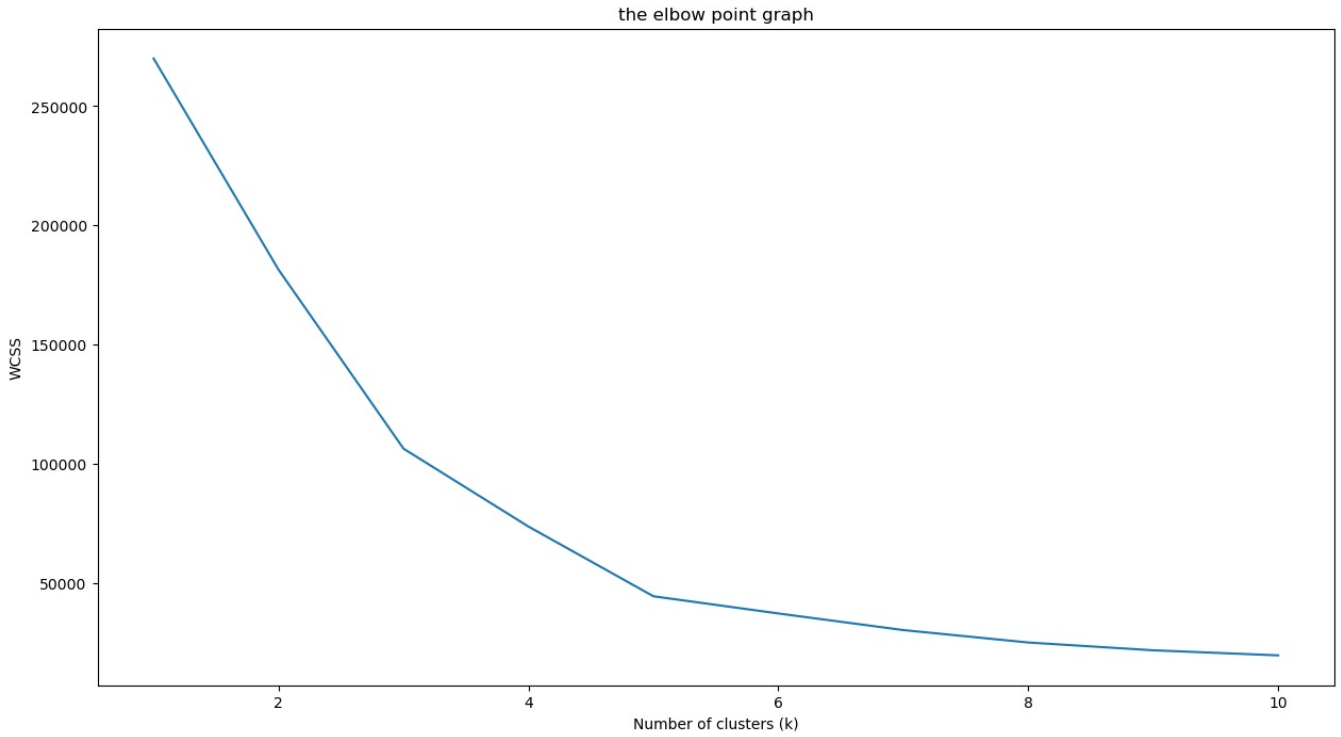


find the k value using the elbow method

In [12]:
```
wcss = []
for I in range(1,11):
    kmeans = KMeans(n_clusters=I, init='k-means++', random_state=2, n_init=10)
    kmeans.fit(x)

    wcss.append(kmeans.inertia_)

plt.figure(figsize=(15,8))
plt.plot(range(1,11), wcss)
plt.title('the elbow point graph')
plt.xlabel('Number of clusters (k)')
plt.ylabel('WCSS')
plt.show()
```



training the k-means algorithm on the training dataset

In [13]:
```
kmeans = KMeans(n_clusters=5, init='k-means++', random_state=0)
y = kmeans.fit_predict(x)
```

```
C:\Users\Admin\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1416: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

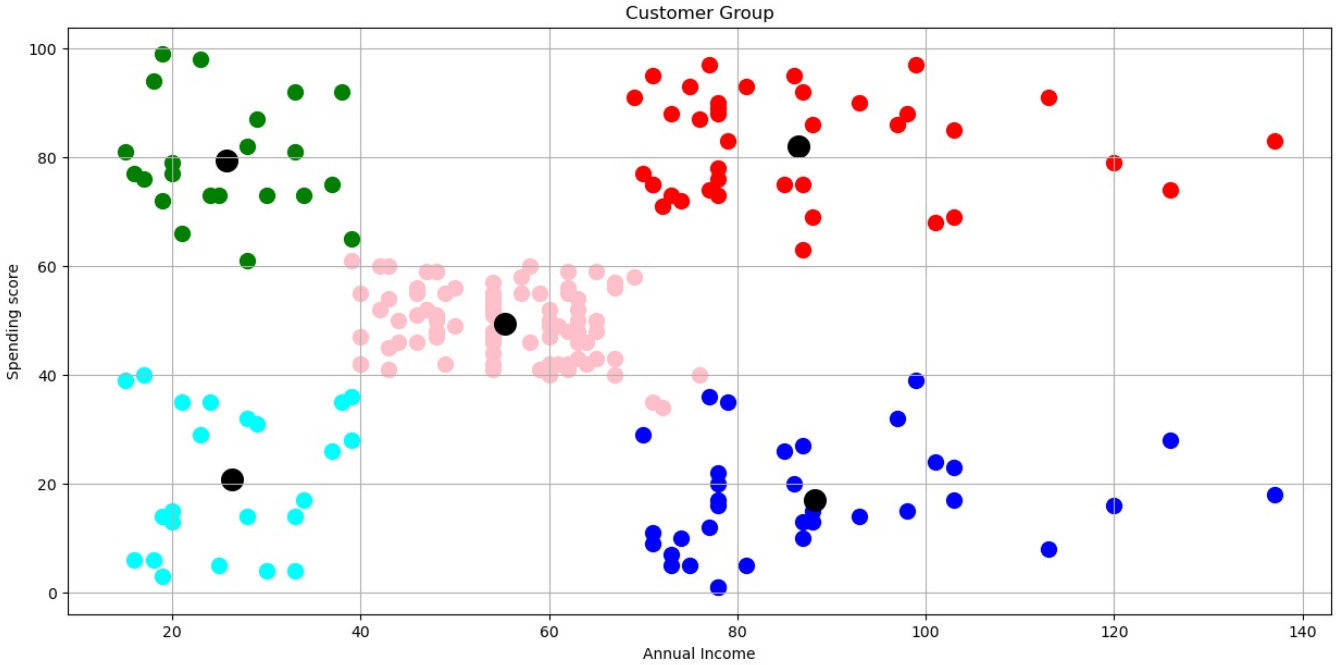
In [14]:
```
kmeans.cluster_centers_
```

Out[14]:
```
array([[55.2962963 , 49.51851852],
       [86.53846154, 82.12820513],
       [88.2       , 17.11428571],
       [26.30434783, 20.91304348],
       [25.72727273, 79.36363636]])
```

visualize the clusters formed

In [15]:
```
plt.figure(figsize=(15,7))
plt.scatter(x[y==0,0], x[y==0,1], s=100, c='pink', label='Cluster 1')
plt.scatter(x[y==1,0], x[y==1,1], s=100, c='red', label='Cluster 2')
plt.scatter(x[y==2,0], x[y==2,1], s=100, c='blue', label='Cluster 3')
plt.scatter(x[y==3,0], x[y==3,1], s=100, c='cyan', label='Cluster 4')
plt.scatter(x[y==4,0], x[y==4,1], s=100, c='green', label='Cluster 5')

plt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1], s=200, c='black')
plt.title('Customer Group')
plt.xlabel('Annual Income')
plt.ylabel('Spending score')
plt.grid()
plt.show()
```



In [ ]:

In [ ]: