In [4]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
df=pd.read_csv(r'D:\Data Science Project\Mall_Customers.csv')
df
```

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
...	...	...	...	...	...
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 [3]:

```
df.isnull().sum()
```

Out[3]:

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

In [5]:

```
df.shape
```

Out[5]:

```
(200, 5)
```

In [6]:

```
df.describe()
```

Out[6]:

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000	200.000000
mean	100.500000	38.850000	60.560000	50.200000
std	57.879185	13.969007	26.264721	25.823522
min	1.000000	18.000000	15.000000	1.000000
25%	50.750000	28.750000	41.500000	34.750000
50%	100.500000	36.000000	61.500000	50.000000
75%	150.250000	49.000000	78.000000	73.000000
max	200.000000	70.000000	137.000000	99.000000

In [7]:

```
df.drop(["CustomerID"],axis=1,inplace=True)
```

In [8]:

```
df.head()
```

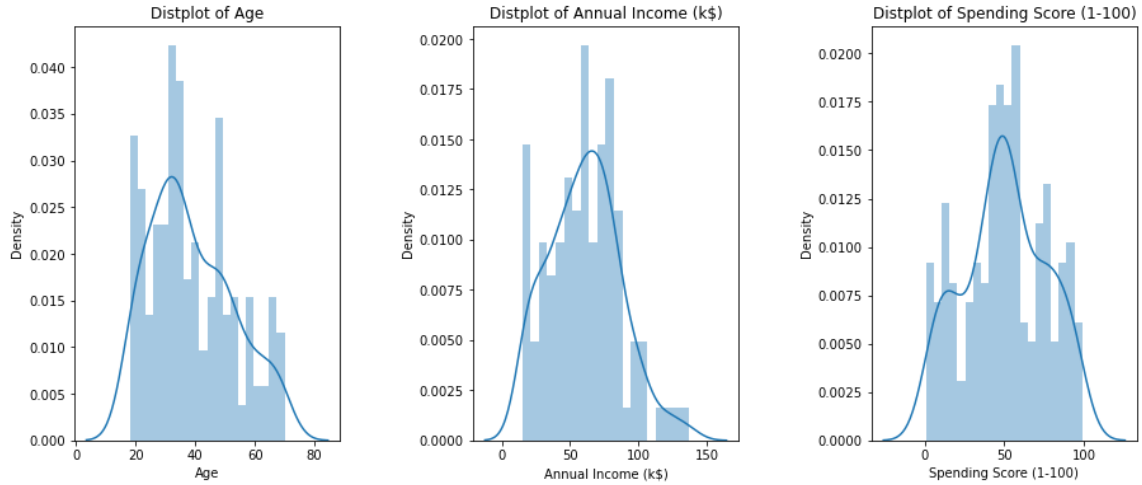
Out[8]:

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

In [11]:

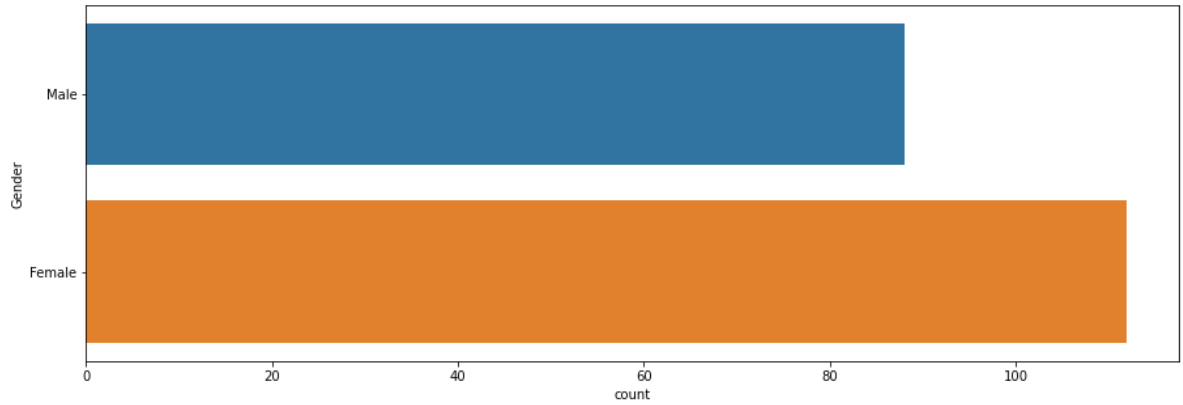
```
plt.figure(1,figsize=(15,6))
n=0
for x in ['Age','Annual Income (k$)','Spending Score (1-100)']:
    n+=1
    plt.subplot(1,3,n)
    plt.subplots_adjust(hspace=0.5,wspace=0.5)
    sns.distplot(df[x],bins=20)
    plt.title("Distplot of {}".format(x))
plt.show()
```

```
in a future version. Please adapt your code to use either `displot` (a fig
ure-level function with similar flexibility) or `histplot` (an axes-level
function for histograms).
  warnings.warn(msg, FutureWarning)
```



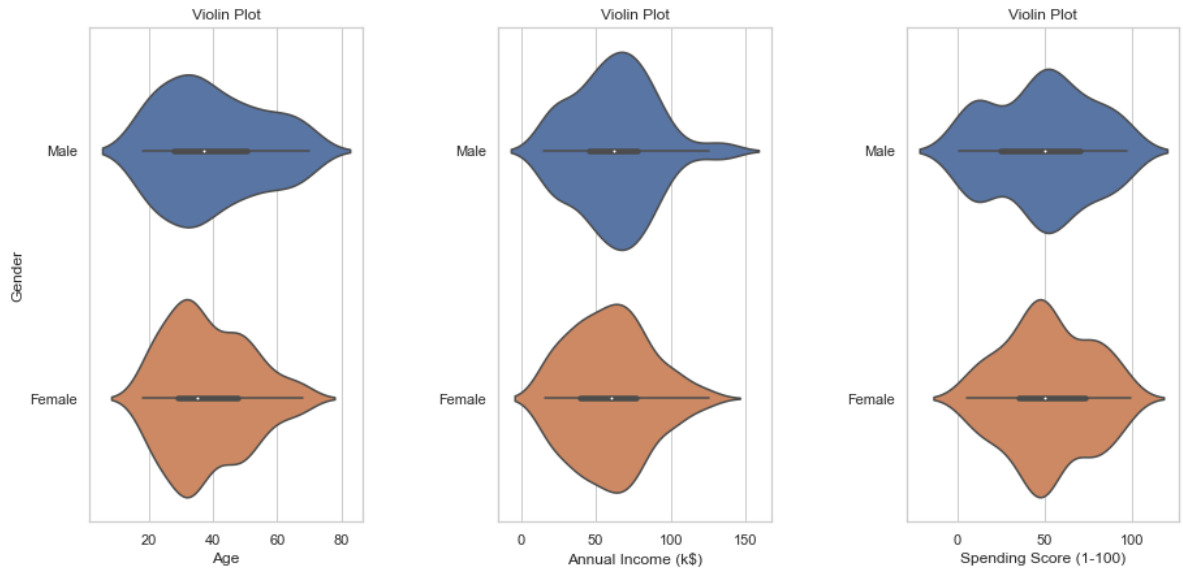
In [12]:

```
plt.figure(figsize=(15,5))
sns.countplot(y='Gender',data=df)
plt.show()
```



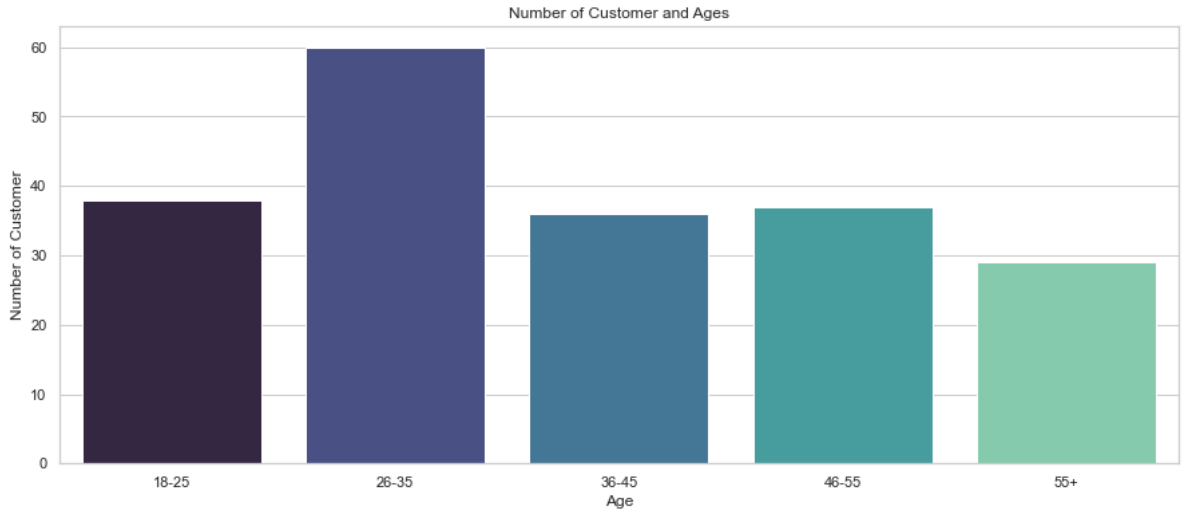
In [14]:

```
plt.figure(1,figsize=(15,7))
n=0
for cols in ['Age','Annual Income (k$)','Spending Score (1-100)']:
    n+=1
    plt.subplot(1,3,n)
    sns.set(style="whitegrid")
    plt.subplots_adjust(hspace=0.5,wspace=0.5)
    sns.violinplot(x=cols,y='Gender',data=df)
    plt.ylabel('Gender' if n==1 else '')
    plt.title('Violin Plot')
plt.show()
```



In [15]:

```
age_18_25 = df.Age[(df.Age>=18) & (df.Age<=25)]
age_26_35 = df.Age[(df.Age>=26) & (df.Age<=35)]
age_36_45 = df.Age[(df.Age>=36) & (df.Age<=45)]
age_46_55 = df.Age[(df.Age>=46) & (df.Age<=55)]
age_55above = df.Age[df.Age>=56]
agex = ["18-25","26-35","36-45","46-55","55+"]
agey = [len(age_18_25.values),len(age_26_35.values),len(age_36_45.values),len(age_46_55.val
plt.figure(figsize=(15,6))
sns.barplot(x=agex,y=agey,palette="mako")
plt.title("Number of Customer and Ages")
plt.xlabel("Age")
plt.ylabel("Number of Customer")
plt.show()
```

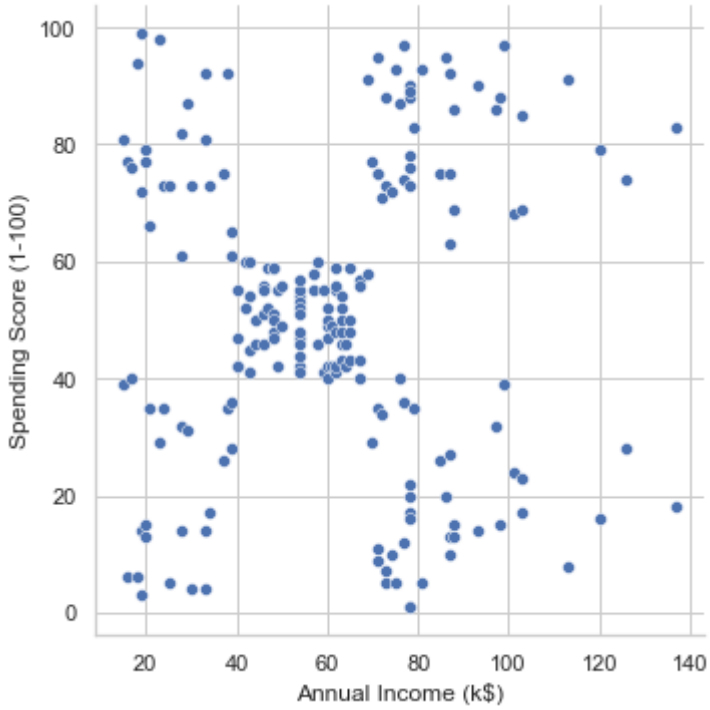


In [18]:

```
sns.relplot(x="Annual Income (k$)",y="Spending Score (1-100)",data=df)
```

Out[18]:

```
<seaborn.axisgrid.FacetGrid at 0x20614c8f3a0>
```

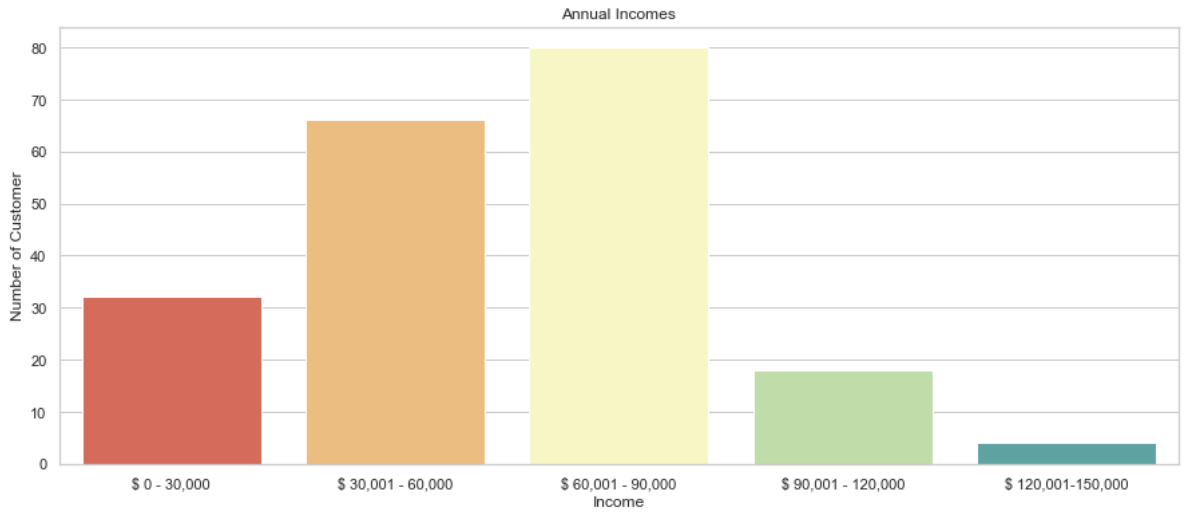


In [20]:

```
ss_1_20 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"]>=1) & (df["Spending Sc
ss_21_40 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"]>=21) & (df["Spending
ss_41_60 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"]>=41) & (df["Spending
ss_61_80 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"]>=61) & (df["Spending
ss_81_100 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"]>=81) & (df["Spending

ssx = ["1-20","21-40","41-60","61-80","81-100"]
ssy = [len(ss_1_20.values),len(ss_21_40.values),len(ss_41_60.values),len(ss_61_80.values),l

plt.figure(figsize=(15,6))
sns.barplot(x=ssx,y=ssy,palette="rocket")
plt.title("Spending Scores")
plt.xlabel("Score")
plt.ylabel("Number of Customer Having the Score")
plt.show()
```

In [22]:

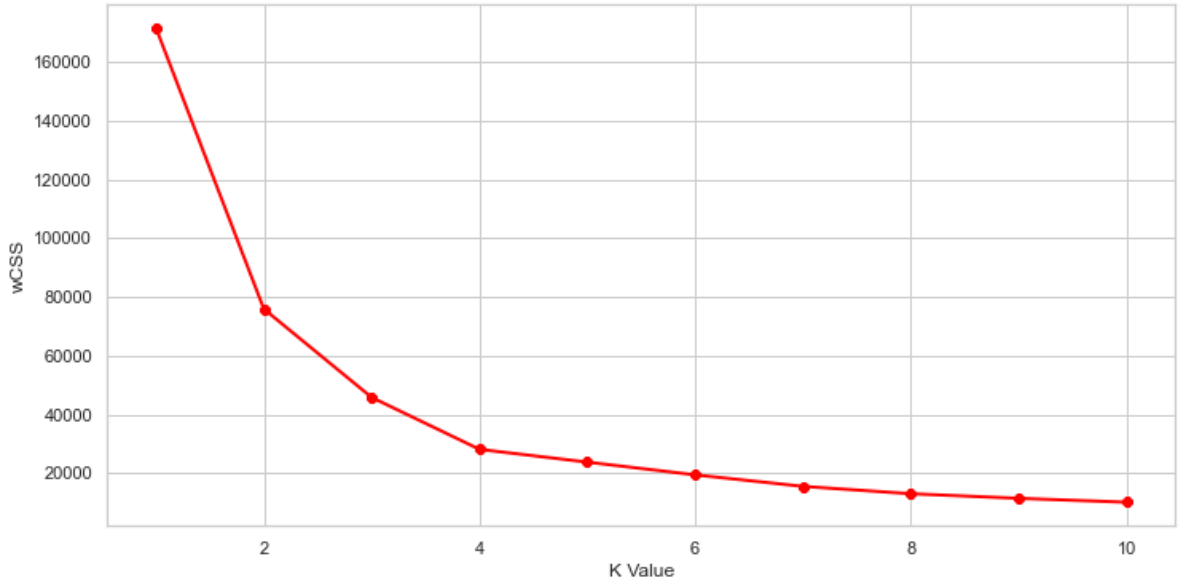
```
ai0_30 = df["Annual Income (k$)"][(df["Annual Income (k$)"]>=0) & (df["Annual Income (k$)"]
ai31_60 = df["Annual Income (k$)"][(df["Annual Income (k$)"]>=31) & (df["Annual Income (k$)
ai61_90 = df["Annual Income (k$)"][(df["Annual Income (k$)"]>=61) & (df["Annual Income (k$)
ai91_120 = df["Annual Income (k$)"][(df["Annual Income (k$)"]>=91) & (df["Annual Income (k$
ai121_150 = df["Annual Income (k$)"][(df["Annual Income (k$)"]>=121) & (df["Annual Income (

aix = ["$ 0 - 30,000","$ 30,001 - 60,000","$ 60,001 - 90,000","$ 90,001 - 120,000","$ 120,0
aiy = [len(ai0_30.values),len(ai31_60.values), len(ai61_90.values),len(ai91_120.values),len

plt.figure(figsize=(15,6))
sns.barplot(x=aix,y=aiy,palette="Spectral")
plt.title("Annual Incomes")
plt.xlabel("Income")
plt.ylabel("Number of Customer")
plt.show()
```



In [24]:

```
X1=df.loc[:,["Age","Spending Score (1-100)"]].values
from sklearn.cluster import KMeans
wcss=[]
for k in range(1,11):
    kmeans = KMeans(n_clusters=k,init="k-means++")
    kmeans.fit(X1)
    wcss.append(kmeans.inertia_)
plt.figure(figsize=(12,6))
plt.plot(range(1,11),wcss,linewidth=2,color="red",marker="8")
plt.xlabel("K Value")
plt.ylabel("wCSS")
plt.show()
```

```
C:\Users\Ruchitaa\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:88
1: UserWarning: KMeans is known to have a memory leak on Windows with MKL, w
hen there are less chunks than available threads. You can avoid it by settin
g the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```



In [27]:

```
kmeans = KMeans(n_clusters=4)
label = kmeans.fit_predict(X1)
print(label)
```

```
[0 1 3 1 0 1 3 1 3 1 3 1 3 1 3 1 0 0 3 1 0 1 3 1 3 1 3 0 3 1 3 1 3 1 3 1 3
 1 3 1 2 1 2 0 3 0 2 0 0 0 2 0 0 2 2 2 2 2 0 2 2 0 2 2 2 0 2 2 0 0 2 2 2 2
 2 0 2 0 0 2 2 0 2 2 0 2 2 0 0 2 2 0 2 0 0 0 2 0 2 0 0 2 2 0 2 0 2 2 2 2 2
 0 0 0 0 0 2 2 2 2 0 0 0 1 0 1 2 1 3 1 3 1 0 1 3 1 3 1 3 1 3 1 0 1 3 1 2 1
 3 1 3 1 3 1 3 1 3 1 3 1 2 1 3 1 3 1 3 1 3 0 3 1 3 1 3 1 3 1 3 1 3 1 3 1 0
 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1]
```

In [26]:

```
print(kmeans.cluster_centers_)
```

```
[[43.29166667 15.02083333]
 [30.1754386  82.35087719]
 [55.70833333 48.22916667]
 [27.61702128 49.14893617]]
```

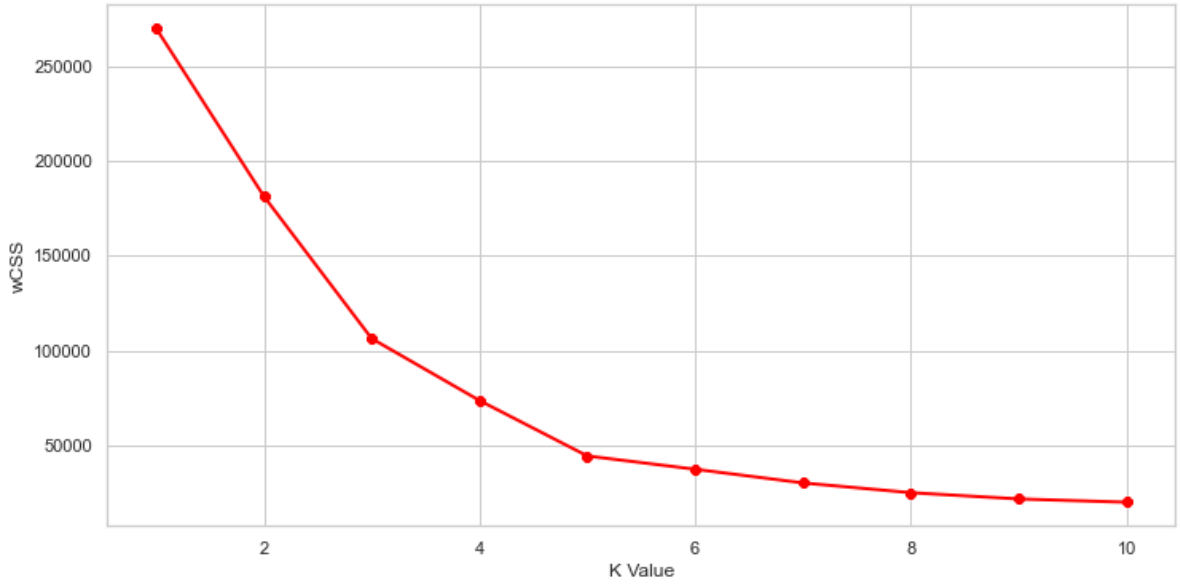
In [28]:

```
plt.scatter(X1[:,0],X1[:,1],c=kmeans.labels_,cmap='rainbow')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],color='black')
plt.title('Clusters of Customers')
plt.xlabel('Age')
plt.ylabel('Spending Score(1-100)')
plt.show()
```

In [30]:

```
X2=df.loc[:,["Annual Income (k$)","Spending Score (1-100)"]].values
from sklearn.cluster import KMeans
wcss=[]
for k in range(1,11):
    kmeans = KMeans(n_clusters=k,init="k-means++")
    kmeans.fit(X2)
    wcss.append(kmeans.inertia_)
plt.figure(figsize=(12,6))
plt.plot(range(1,11),wcss,linewidth=2,color="red",marker="8")
plt.xlabel("K Value")
plt.ylabel("wCSS")
plt.show()
```

```
C:\Users\Ruchitaa\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:88
1: UserWarning: KMeans is known to have a memory leak on Windows with MKL, w
hen there are less chunks than available threads. You can avoid it by settin
g the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```



In [33]:

```
kmeans = KMeans(n_clusters=5)
label = kmeans.fit_predict(X2)
print(label)
```

```
[3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3
 0 3 0 3 0 3 4 3 0 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
 4 4 4 4 4 4 4 4 4 4 4 4 2 1 2 4 2 1 2 1 2 4 2 1 2 1 2 1 2 1 2 4 2 1 2 1 2
 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1
 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2]
```

In [34]:

```
print(kmeans.cluster_centers_)
```

```
[[25.72727273 79.36363636]
 [88.2        17.11428571]
 [86.53846154 82.12820513]
 [26.30434783 20.91304348]
 [55.2962963  49.51851852]]
```

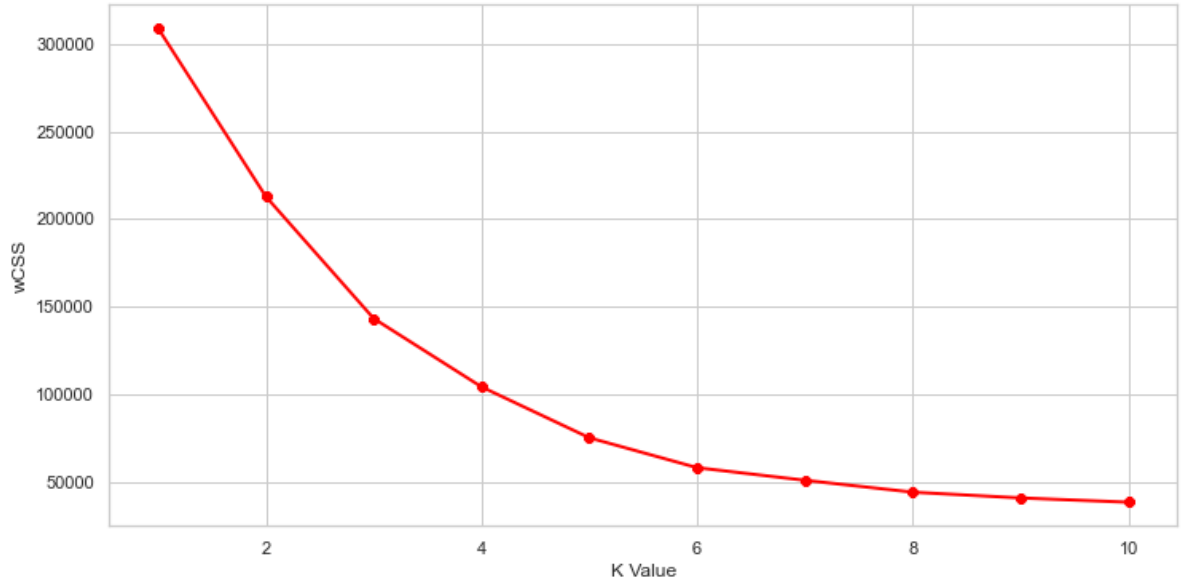
In [35]:

```
plt.scatter(X2[:,0],X1[:,1],c=kmeans.labels_,cmap='rainbow')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],color='black')
plt.title('Clusters of Customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score(1-100)')
plt.show()
```

In [36]:

```
X3=df.iloc[:,1:]
wcss=[]
for k in range(1,11):
    kmeans = KMeans(n_clusters=k,init="k-means++")
    kmeans.fit(X3)
    wcss.append(kmeans.inertia_)
plt.figure(figsize=(12,6))
plt.plot(range(1,11),wcss,linewidth=2,color="red",marker="8")
plt.xlabel("K Value")
plt.ylabel("wCSS")
plt.show()
```

```
C:\Users\Ruchitaa\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:88
1: UserWarning: KMeans is known to have a memory leak on Windows with MKL, w
hen there are less chunks than available threads. You can avoid it by settin
g the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```



In [37]:

```
kmeans = KMeans(n_clusters=5)
label = kmeans.fit_predict(X3)
print(label)
```

```
[3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3
 2 3 2 3 2 3 2 3 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 1 4 1 0 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 0 1 4 1 4 1
 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4
 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1]
```

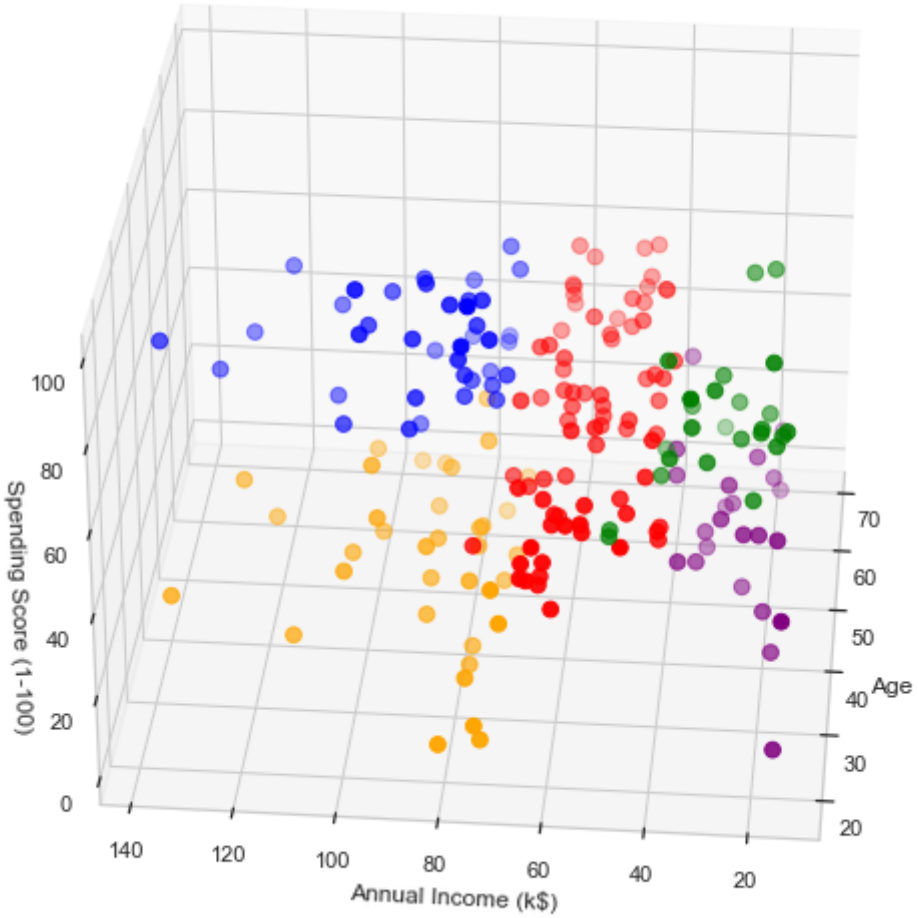
In [38]:

```
print(kmeans.cluster_centers_)
```

```
[[43.08860759 55.29113924 49.56962025]
 [32.69230769 86.53846154 82.12820513]
 [25.52173913 26.30434783 78.56521739]
 [45.2173913  26.30434783 20.91304348]
 [40.66666667 87.75       17.58333333]]
```

In [42]:

```
clusters = kmeans.fit_predict(X3)
df["label"]=clusters
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(figsize=(20,10))
ax=fig.add_subplot(111,projection='3d')
ax.scatter(df.Age[df.label==0],df["Annual Income (k$)"][df.label==0],df["Spending Score (1-
ax.scatter(df.Age[df.label==1],df["Annual Income (k$)"][df.label==1],df["Spending Score (1-
ax.scatter(df.Age[df.label==2],df["Annual Income (k$)"][df.label==2],df["Spending Score (1-
ax.scatter(df.Age[df.label==3],df["Annual Income (k$)"][df.label==3],df["Spending Score (1-
ax.scatter(df.Age[df.label==4],df["Annual Income (k$)"][df.label==4],df["Spending Score (1-
ax.view_init(30,185)
plt.xlabel('Age')
plt.ylabel('Annual Income (k$)')
ax.set_zlabel('Spending Score (1-100)')
plt.show()
```



In [ ]: