In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from mpl_toolkits.mplot3d import Axes3D
%matplotlib inline
```

In [2]:
```
data=pd.read_csv("C:/sameer/Mall_Customers.csv - Mall_Customers.csv (1).csv")
```

In [3]:
```
data.shape
```

Out[3]:
```
(200, 5)
```

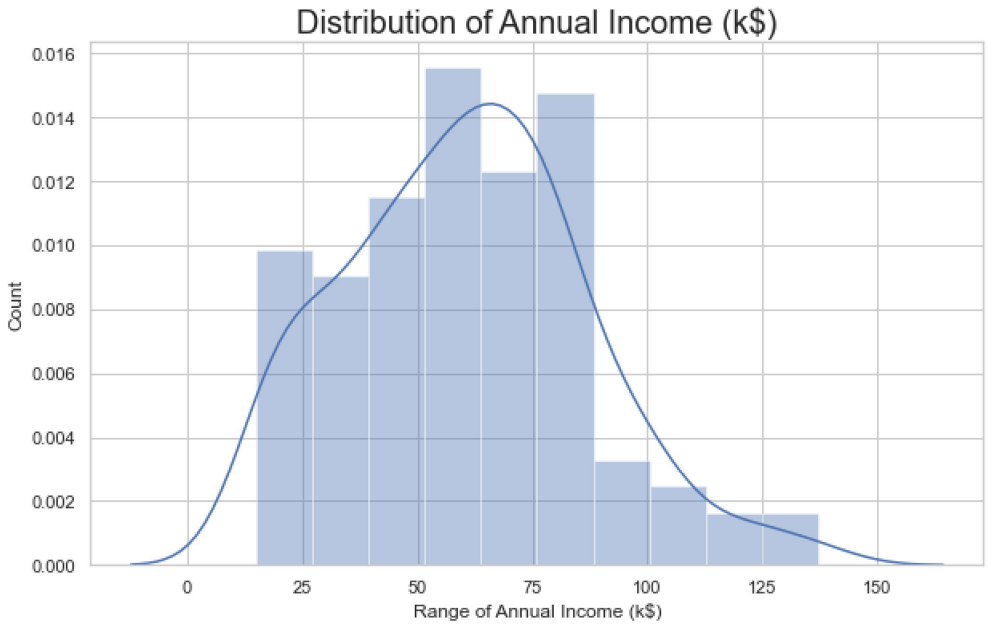
In [4]:
```
data.head()
```

Out[4]:

	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

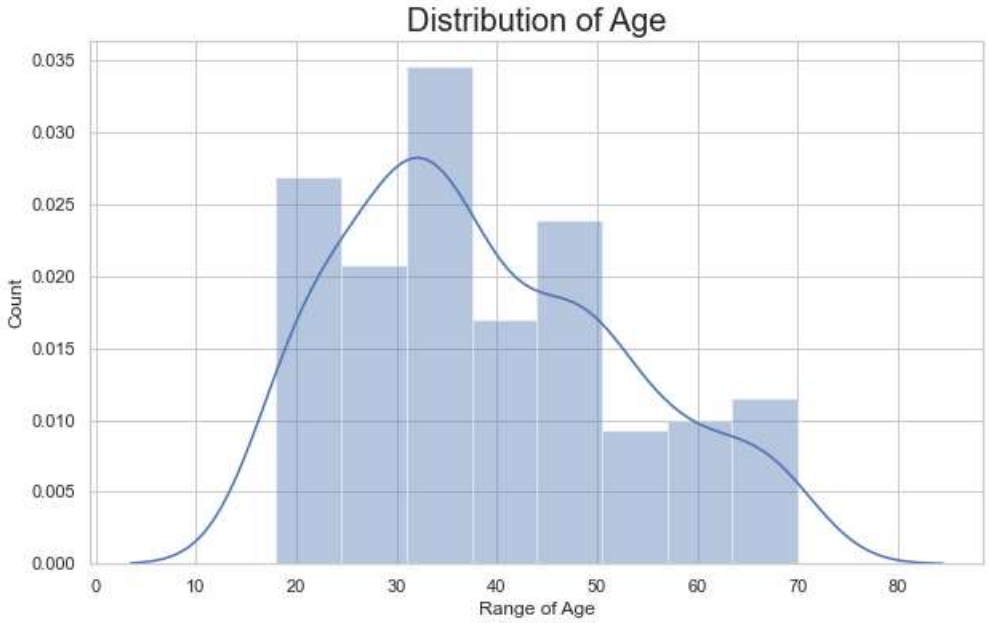
In [5]:
```
#Distribution of Annnual Income
plt.figure(figsize=(10, 6))
sns.set(style = 'whitegrid')
sns.distplot(data['Annual Income (k$)'])
plt.title('Distribution of Annual Income (k$)', fontsize = 20)
plt.xlabel('Range of Annual Income (k$)')
plt.ylabel('Count')
```

Out[5]:
```
Text(0, 0.5, 'Count')
```



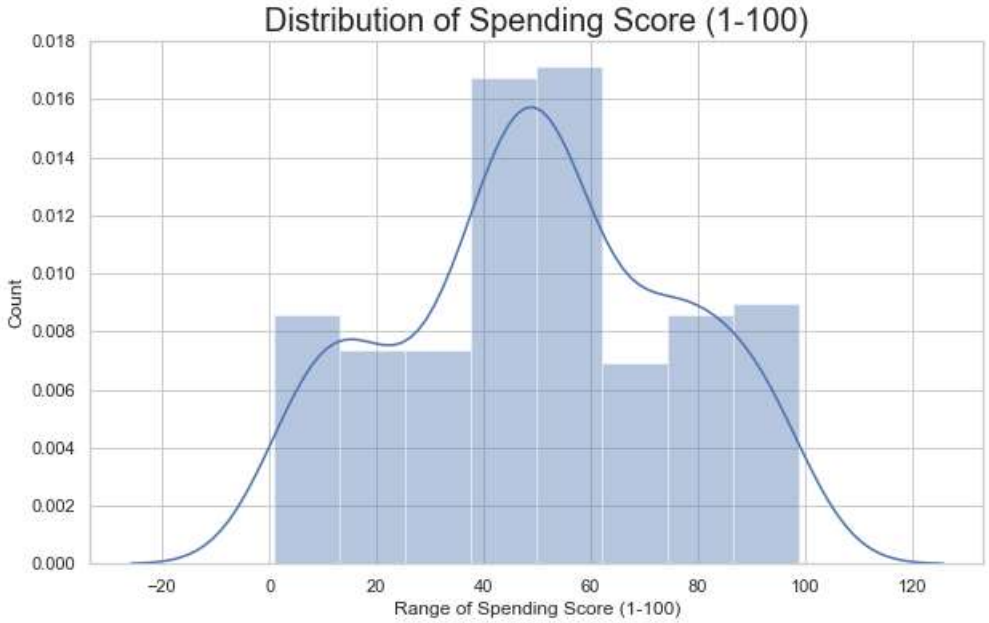
In [6]:
```
#Distribution of age
plt.figure(figsize=(10, 6))
sns.set(style = 'whitegrid')
sns.distplot(data['Age'])
plt.title('Distribution of Age', fontsize = 20)
plt.xlabel('Range of Age')
plt.ylabel('Count')
```

Out[6]: Text(0, 0.5, 'Count')

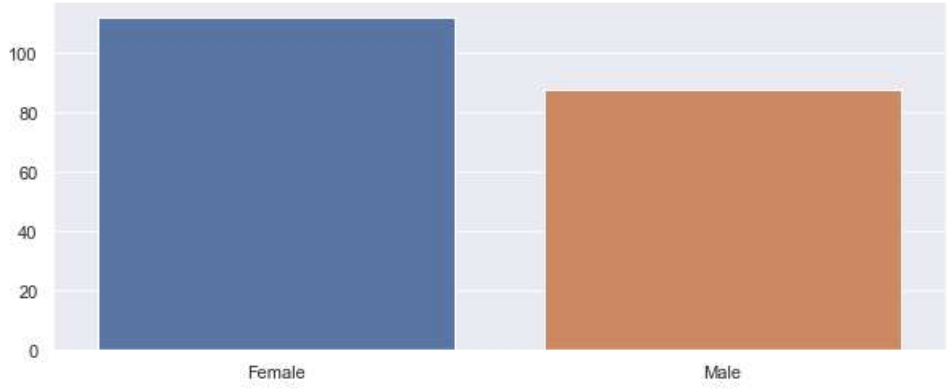


In [7]:
```
#Distribution of spending score
plt.figure(figsize=(10, 6))
sns.set(style = 'whitegrid')
sns.distplot(data['Spending Score (1-100)'])
plt.title('Distribution of Spending Score (1-100)', fontsize = 20)
plt.xlabel('Range of Spending Score (1-100)')
plt.ylabel('Count')
```

Out[7]: Text(0, 0.5, 'Count')



In [8]:
```
#Gender Analysis:
genders = data.Genre.value_counts()
sns.set_style("darkgrid")
plt.figure(figsize=(10,4))
sns.barplot(x=genders.index, y=genders.values)
plt.show()
```



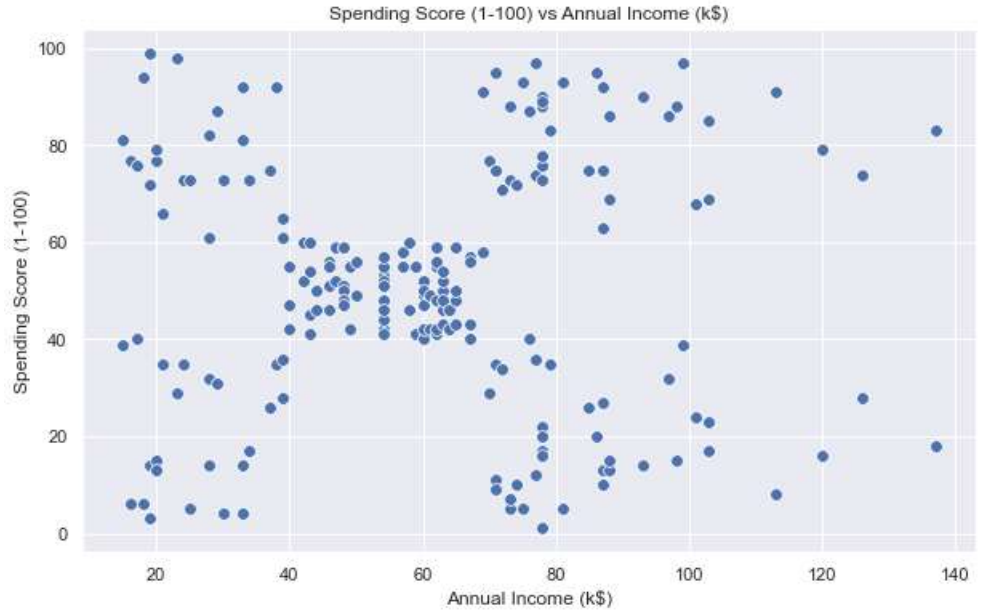
In [9]:
```
#We take just the Annual Income and Spending score
df1=data[["CustomerID","Genre","Age","Annual Income (k$)","Spending Score (1-100)"]]
X=df1[["Annual Income (k$)","Spending Score (1-100)"]]
```

In [10]:
```
X.head()
```

Out[10]:

	Annual Income (k$)	Spending Score (1-100)
0	15	39
1	15	81
2	16	6
3	16	77
4	17	40

In [11]:
```
#Scatterplot of the input data
plt.figure(figsize=(10,6))
sns.scatterplot(x = 'Annual Income (k$)',y = 'Spending Score (1-100)', data = X ,s = 60 )
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.title('Spending Score (1-100) vs Annual Income (k$)')
plt.show()
```

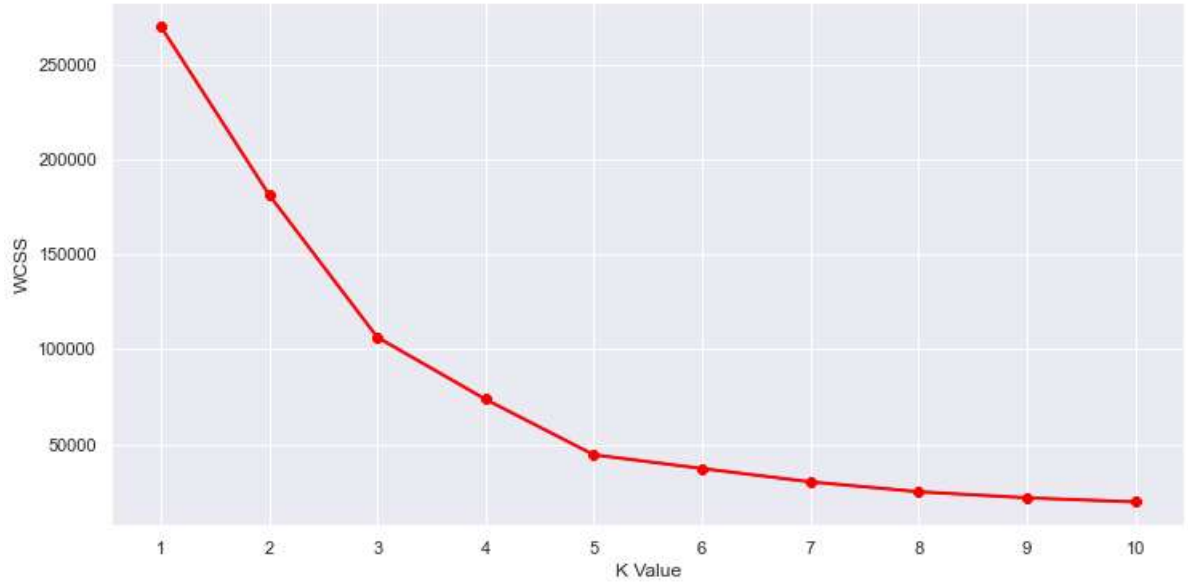


In [12]:
```
#Importing KMeans from sklearn
from sklearn.cluster import KMeans
```

In [13]:
```
wcss=[]
for i in range(1,11):
    km=KMeans(n_clusters=i)
    km.fit(X)
    wcss.append(km.inertia_)
```

```
C:\Users\samir\Anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1037: UserWarning: KMeans is known to have a memory leak on Windows wi
th MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  "KMeans is known to have a memory leak on Windows "
```

In [14]:
```
#The elbow curve
plt.figure(figsize=(12,6))
plt.plot(range(1,11),wcss)
plt.plot(range(1,11),wcss, linewidth=2, color="red", marker ="8")
plt.xlabel("K Value")
plt.xticks(np.arange(1,11,1))
plt.ylabel("WCSS")
plt.show()
```



In [15]:
```
#Taking 5 clusters
km1=KMeans(n_clusters=5)
```

In [16]:
```
#Fitting the input data
km1.fit(X)
```

Out[16]:
```
KMeans(n_clusters=5)
```

In [17]:
```
#predicting the labels of the input data
y=km1.predict(X)
```

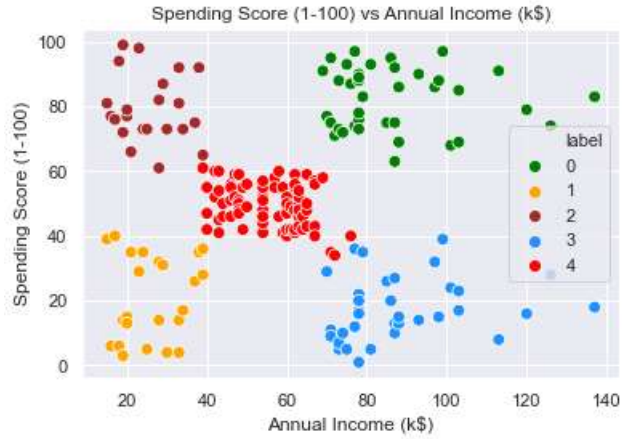
In [18]:
```
#adding the labels to a column named label
df1["label"] = y
```

In [19]:
```
#The new dataframe with the clustering done
df1.head()
```

Out[19]:

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

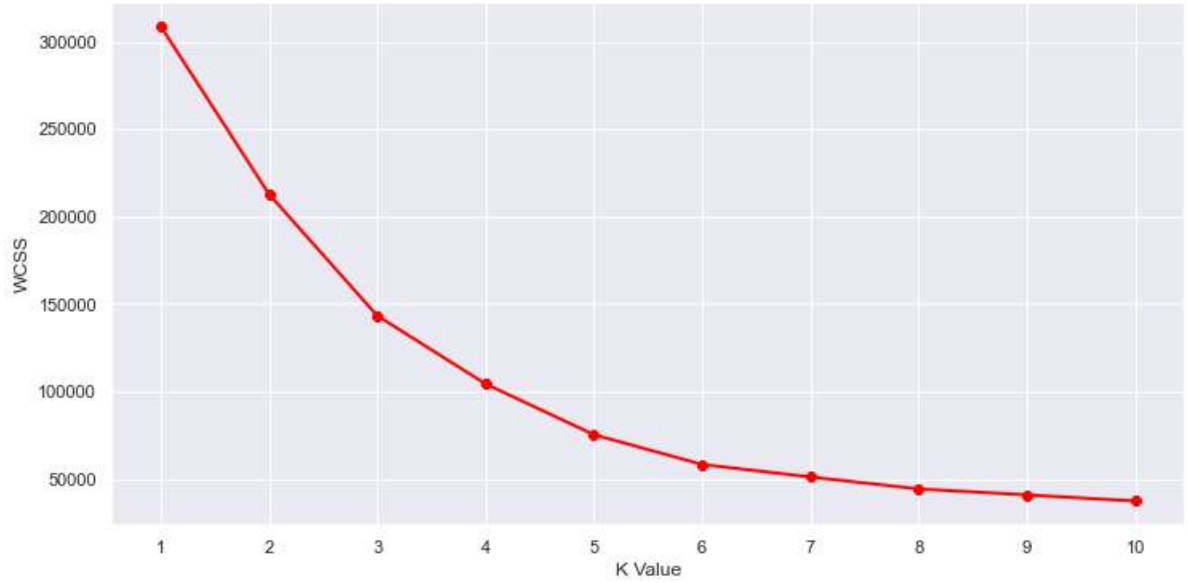
In [20]:
```
#Scatterplot of the clustersplt.figure(figsize=(10,6))
sns.scatterplot(x = 'Annual Income (k$)',y = 'Spending Score (1-100)',hue="label",
palette=['green','orange','brown','dodgerblue','red'], legend='full',data = df1 ,s = 60 )
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.title('Spending Score (1-100) vs Annual Income (k$)')
plt.show()
```



In [21]:
```
#Taking the features
df2=data
X2=df2[["Age","Annual Income (k$)","Spending Score (1-100)"]]
```

In [22]:
```
#Now we calculate the Within Cluster Sum of Squared Errors (WSS) for different values of k.
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.xticks(np.arange(1,11,1))
plt.ylabel("WCSS")
plt.show()
```

```
C:\Users\samir\Anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1037: UserWarning: KMeans is known to have a memory leak on Windows wi
th MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  "KMeans is known to have a memory leak on Windows "
```



In [23]:
```
#Here can assume that K=5 will be a good value.
#We choose the k for which WSS starts to diminish
km2 = KMeans(n_clusters=5)
y2 = km.fit_predict(X2)
df2["label"] = y2
```

In [24]:
```
#The data with labels
df2.head()
```

Out[24]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)	label
0	1	Male	19	15	39	9
1	2	Male	21	15	81	4
2	3	Female	20	16	6	5
3	4	Female	23	16	77	4
4	5	Female	31	17	40	9

In [25]:
```
#Now, if we want to know the customer IDs, we can do that too.
cust1=df2[df2["label"]==1]
print('Number of customer in 1st group=', len(cust1))
print('They are -', cust1["CustomerID"].values)
print("---------------------------------------------")
cust2=df2[df2["label"]==2]
print('Number of customer in 2nd group=', len(cust2))
print('They are -', cust2["CustomerID"].values)
print("---------------------------------------------")
cust3=df2[df2["label"]==0]
print('Number of customer in 3rd group=', len(cust3))
print('They are -', cust3["CustomerID"].values)
print("---------------------------------------------")
cust4=df2[df2["label"]==3]
print('Number of customer in 4th group=', len(cust4))
print('They are -', cust4["CustomerID"].values)
print("---------------------------------------------")
cust5=df2[df2["label"]==4]
print('Number of customer in 5th group=', len(cust5))
print('They are -', cust5["CustomerID"].values)
print("---------------------------------------------")
```

```
Number of customer in 1st group= 29
They are - [124 126 128 130 132 134 136 138 140 142 144 146 148 150 152 154 156 158
 160 162 164 166 168 170 172 174 176 178 180]
---------------------------------------------
Number of customer in 2nd group= 22
They are - [129 131 135 137 139 141 145 149 151 153 155 157 159 163 165 167 169 171
 173 175 177 179]
---------------------------------------------
Number of customer in 3rd group= 27
They are - [ 41  47  51  54  55  57  58  60  61  63  64  65  68  71  73  74  75  81
  83  87  91 103 107 109 110 111 117]
---------------------------------------------
Number of customer in 4th group= 29
They are - [ 48  52  53  59  62  66  69  70  76  79  85  88  89  92  95  96  98 100
 101 104 106 112 114 115 116 121 125 133 143]
---------------------------------------------
Number of customer in 5th group= 23
They are - [ 2  4  6  8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46]
---------------------------------------------
```

In [ ]: