In [ ]:
```
#importing necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
#loading the dataset
df = pd.read_csv("D:/Analytics/Datasets/Customer Data/Customers.csv")
```

In [ ]:
```
#extracting first-five rows
df.head()
```

Out[ ]:

	CustomerID	Gender	Age	Annual Income ($)	Spending Score (1-100)	Profession	Work Experience	Family Size
0	1	Male	19	15000	39	Healthcare	1	4
1	2	Male	21	35000	81	Engineer	3	3
2	3	Female	20	86000	6	Engineer	1	1
3	4	Female	23	59000	77	Lawyer	0	2
4	5	Female	31	38000	40	Entertainment	2	6

In [ ]:
```
#extracting last-five rows
df.tail()
```

Out[ ]:

	CustomerID	Gender	Age	Annual Income ($)	Spending Score (1-100)	Profession	Work Experience	Family Size
1995	1996	Female	71	184387	40	Artist	8	7
1996	1997	Female	91	73158	32	Doctor	7	7
1997	1998	Male	87	90961	14	Healthcare	9	2
1998	1999	Male	77	182109	4	Executive	7	2
1999	2000	Male	90	110610	52	Entertainment	5	2

In [ ]:
```
#determining the shape
df.shape
```

Out[ ]:
```
(2000, 8)
```

In [ ]:
```
#determining the size
df.size
```

Out[ ]:
```
16000
```

In [ ]:
```
#checking the null values
df.isnull().sum()
```

Out[ ]:
```
CustomerID                0
Gender                    0
Age                       0
Annual Income ($)         0
Spending Score (1-100)    0
Profession               35
Work Experience           0
Family Size               0
dtype: int64
```

In [ ]:
```
#determining mode of 'Profession' column
df["Profession"].mode()
```

Out[ ]:
```
0    Artist
Name: Profession, dtype: object
```

In [ ]:
```
#replacing null values with mode
df["Profession"].fillna("Artist", inplace=True)
```

In [ ]:
```
# checking the duplicates
df.duplicated().value_counts()
```

Out[ ]:
```
False    2000
dtype: int64
```

In [ ]:
```
#checking the information
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 8 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   CustomerID              2000 non-null   int64
 1   Gender                  2000 non-null   object
 2   Age                     2000 non-null   int64
 3   Annual Income ($)       2000 non-null   int64
 4   Spending Score (1-100)  2000 non-null   int64
 5   Profession              2000 non-null   object
 6   Work Experience         2000 non-null   int64
 7   Family Size             2000 non-null   int64
dtypes: int64(6), object(2)
memory usage: 125.1+ KB
```

In [ ]:
```
#extracting statistical summary
df.describe()
```

Out[ ]:

	CustomerID	Age	Annual Income ($)	Spending Score (1-100)	Work Experience	Family Size
count	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
mean	1000.500000	48.960000	110731.821500	50.962500	4.102500	3.768500
std	577.494589	28.429747	45739.536688	27.934661	3.922204	1.970749
min	1.000000	0.000000	0.000000	0.000000	0.000000	1.000000
25%	500.750000	25.000000	74572.000000	28.000000	1.000000	2.000000
50%	1000.500000	48.000000	110045.000000	50.000000	3.000000	4.000000
75%	1500.250000	73.000000	149092.750000	75.000000	7.000000	5.000000
max	2000.000000	99.000000	189974.000000	100.000000	17.000000	9.000000

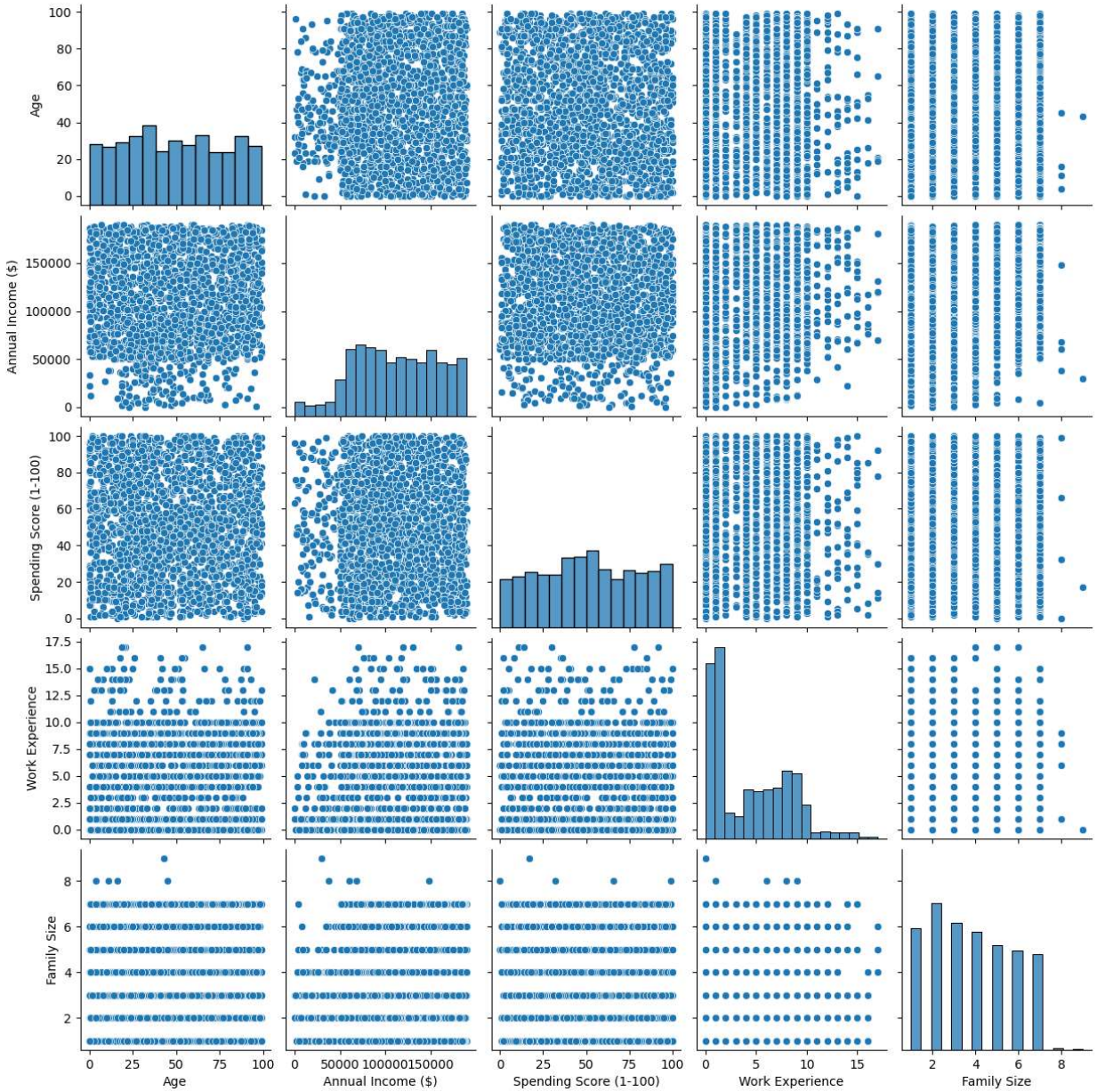
In [ ]:
```
ProfessionList = ['Artist','Healthcare','Entertainment','Engineer','Doctor','Executive
```

In [ ]:
```
x = 1
for i in ProfessionList:
    df_profession = df[df.Profession==i]
    average_salary = df_profession['Spending Score (1-100)'].quantile(0.5)
    print(x, 'Spending score median of', i, 'is', round(average_salary, 2))
    x += 1
```

```
1 Spending score median of Artist is 52.0
2 Spending score median of Healthcare is 51.0
3 Spending score median of Entertainment is 53.0
4 Spending score median of Engineer is 47.0
5 Spending score median of Doctor is 50.0
6 Spending score median of Executive is 49.0
7 Spending score median of Lawyer is 49.0
8 Spending score median of Marketing is 46.0
9 Spending score median of Homemaker is 45.5
```

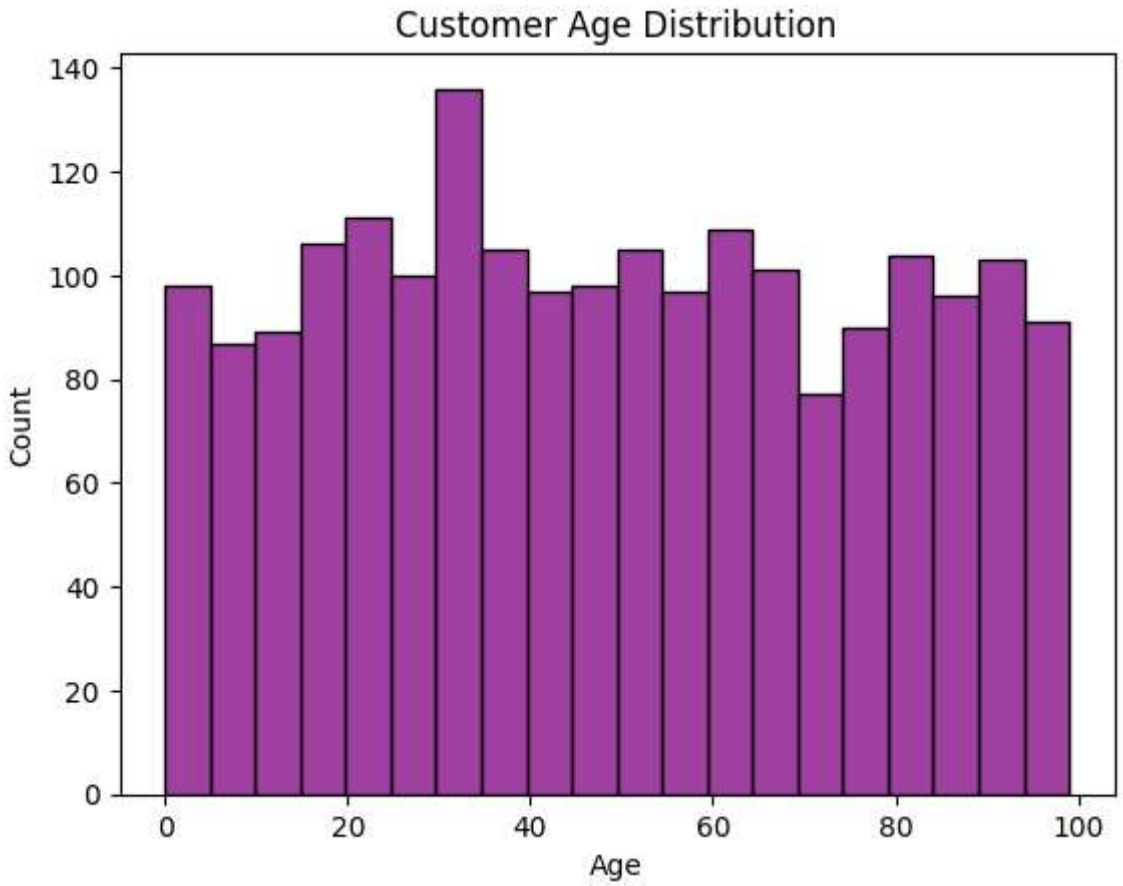
In [ ]:
```
#creating the pairplot
sns.pairplot(df.drop("CustomerID", axis=1))
```

Out[ ]:
```
<seaborn.axisgrid.PairGrid at 0x24c62b5bfa0>
```

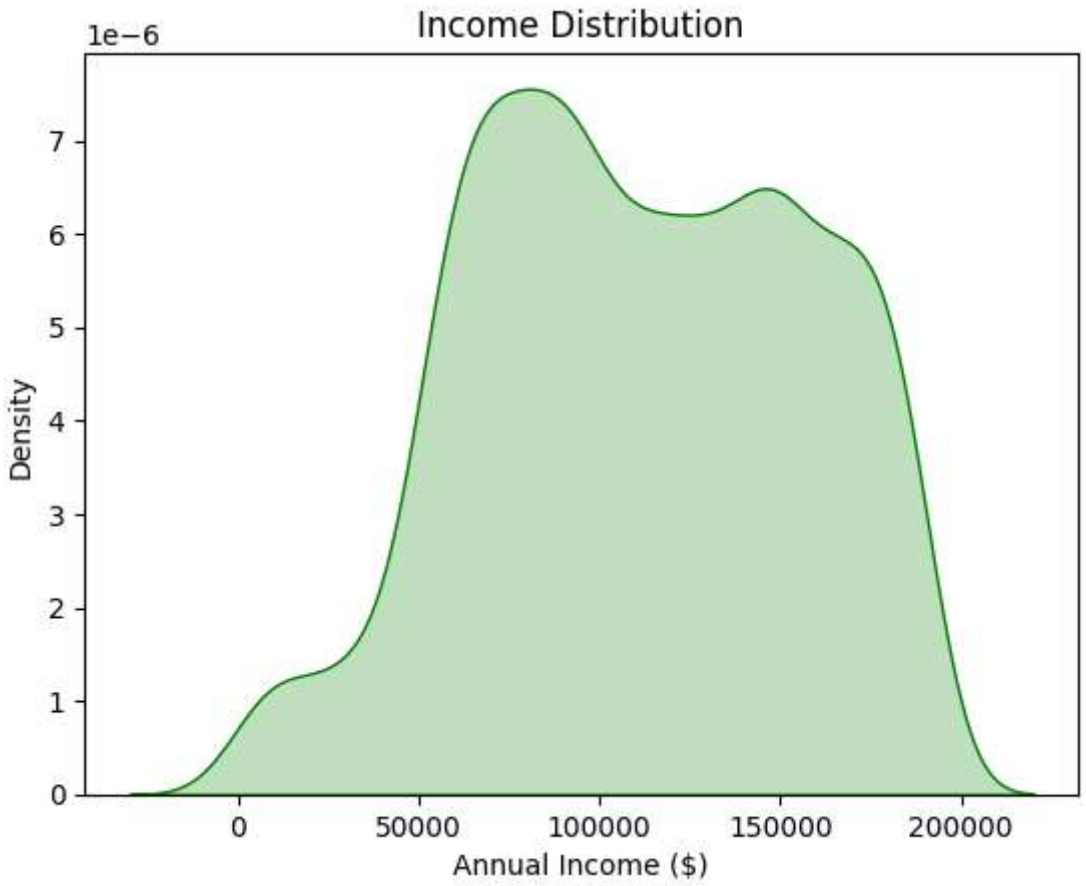


In [ ]:
```
# segment customers by gender
sns.countplot(x='Gender', data=df)
plt.title('Customer Gender Distribution')
plt.show()
```

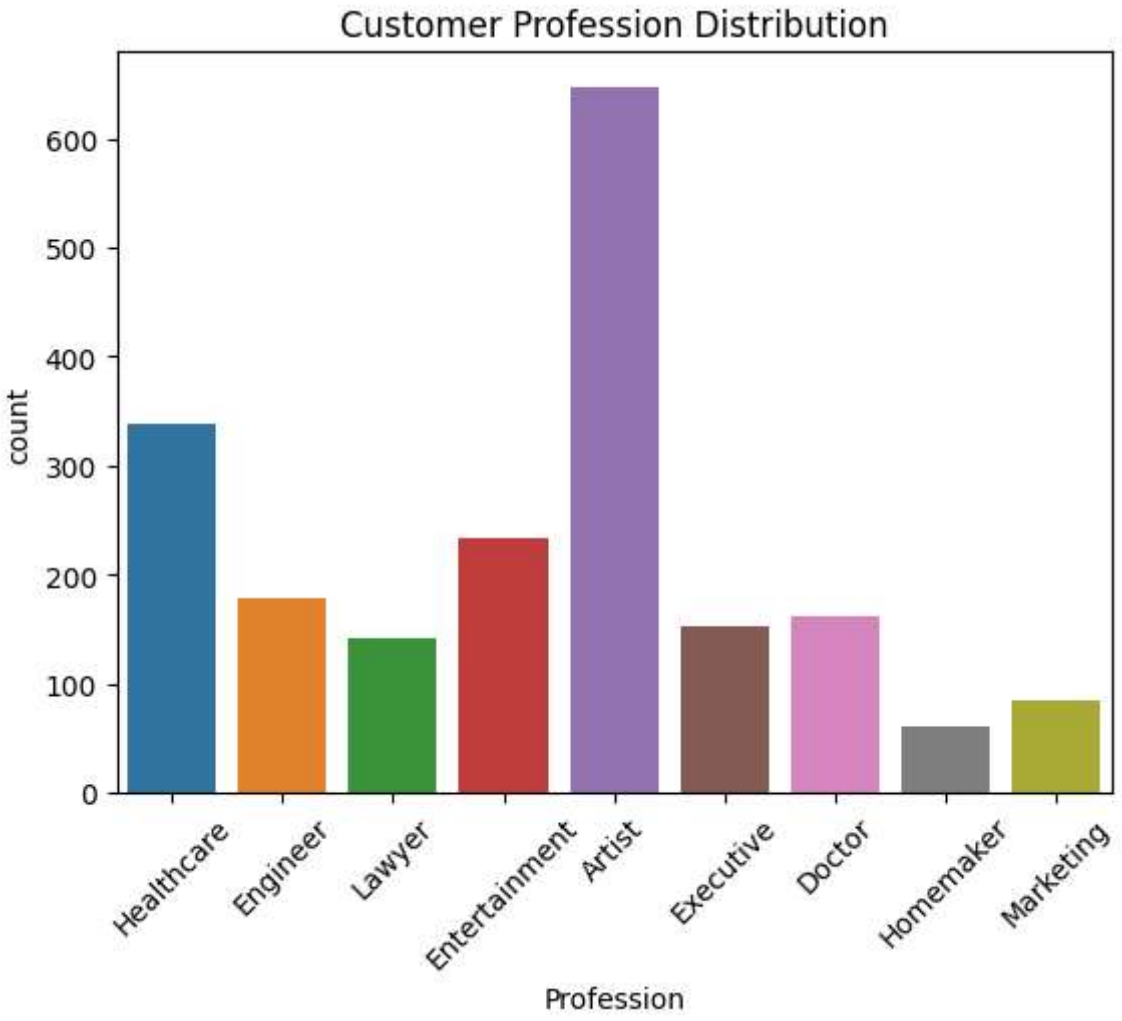
In [ ]:
```
# segment customers by age
sns.histplot(x='Age', data=df, color='purple', bins=20)
plt.title('Customer Age Distribution')
plt.show()
```



In [ ]:
```
# segment by income
sns.kdeplot(x='Annual Income ($)', data=df, color="green", fill=True)
plt.title('Income Distribution')
plt.show()
```

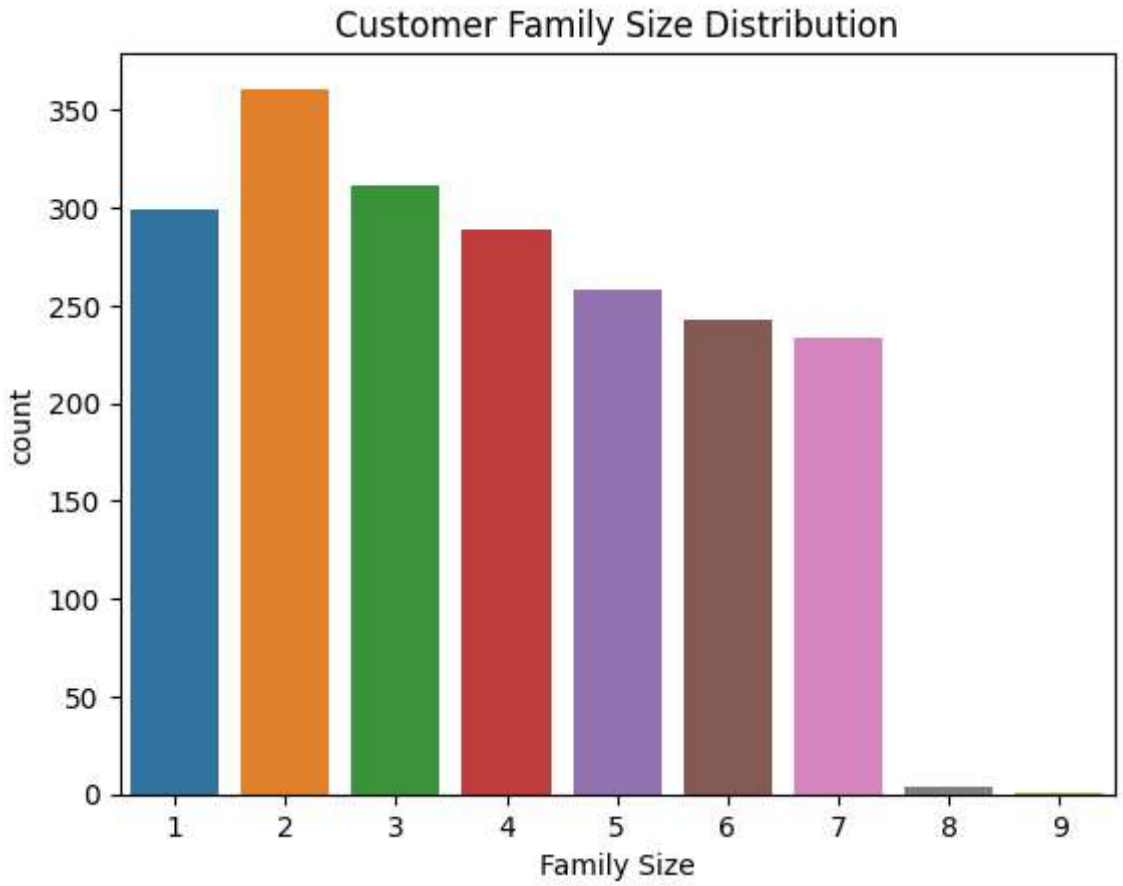


In [ ]:
```
# segment customers by profession
sns.countplot(x='Profession', data=df)
plt.xticks(rotation=45)
plt.title('Customer Profession Distribution')
plt.show()
```



In [ ]:
```
# segment customers by work experience
sns.kdeplot(x='Work Experience', data=df, color='red', fill=True)
plt.title('Work Experience Distribution')
plt.show()
```

In [ ]:
```
# segment customers by family size
sns.countplot(x='Family Size', data=df)
plt.title('Customer Family Size Distribution')
plt.show()
```

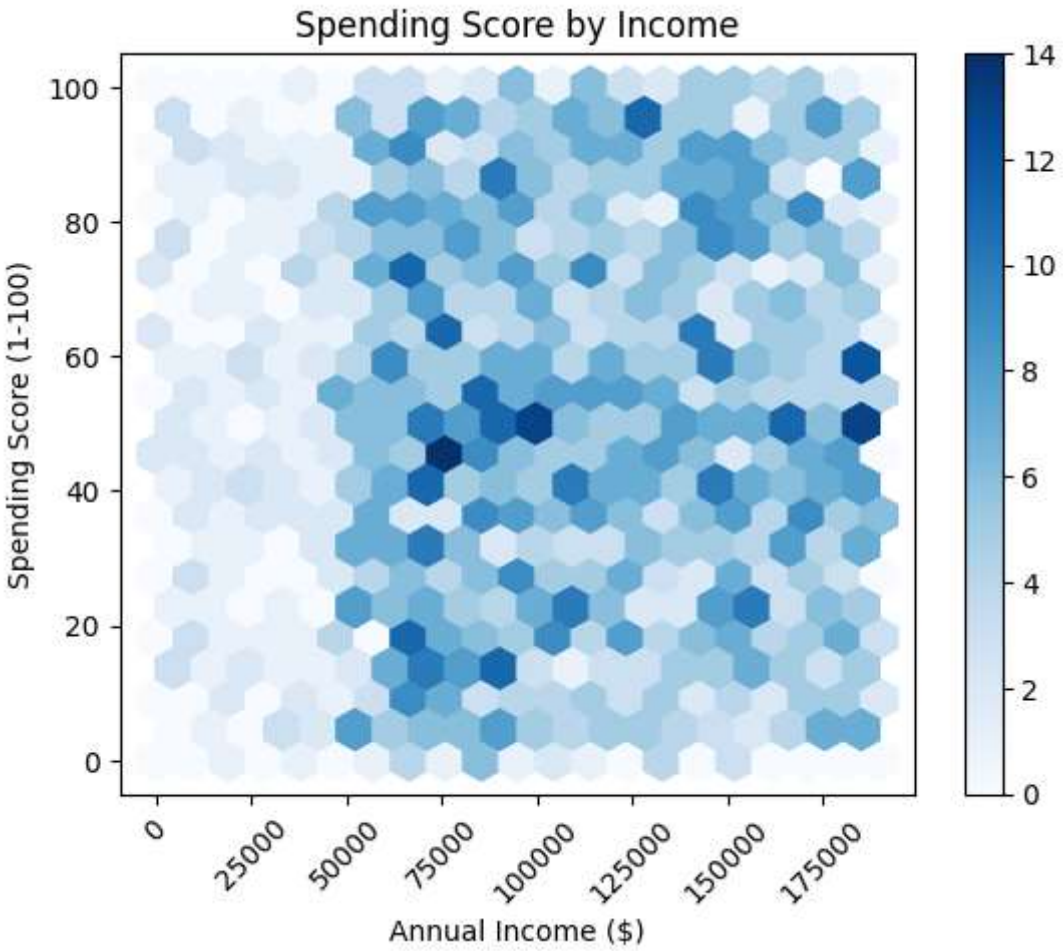


In [ ]:
```
# spending score by gender
sns.violinplot(x='Gender', y='Spending Score (1-100)', data=df)
plt.title('Spending Score by Gender')
plt.show()
```

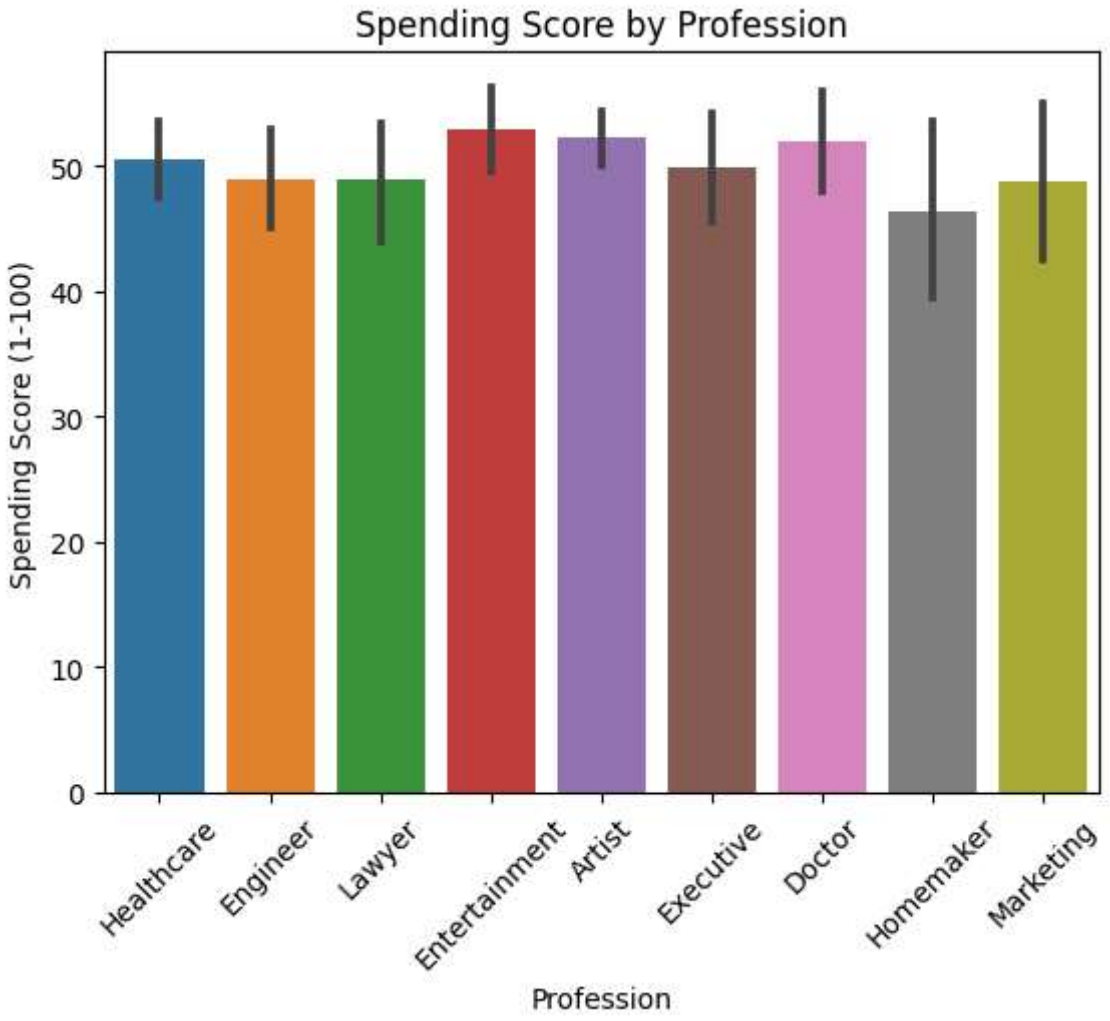
In [ ]:
```
# spending behavior by age
sns.lineplot(x='Age', y='Spending Score (1-100)', color="orange", data=df)
plt.title('Spending Score by Age')
plt.show()
```

In [ ]:
```
# analyze spending behavior by age and gender
sns.lineplot(x='Age', y='Spending Score (1-100)', hue='Gender', data=df)
plt.title('Spending Score by Age and Gender')
plt.show()
```

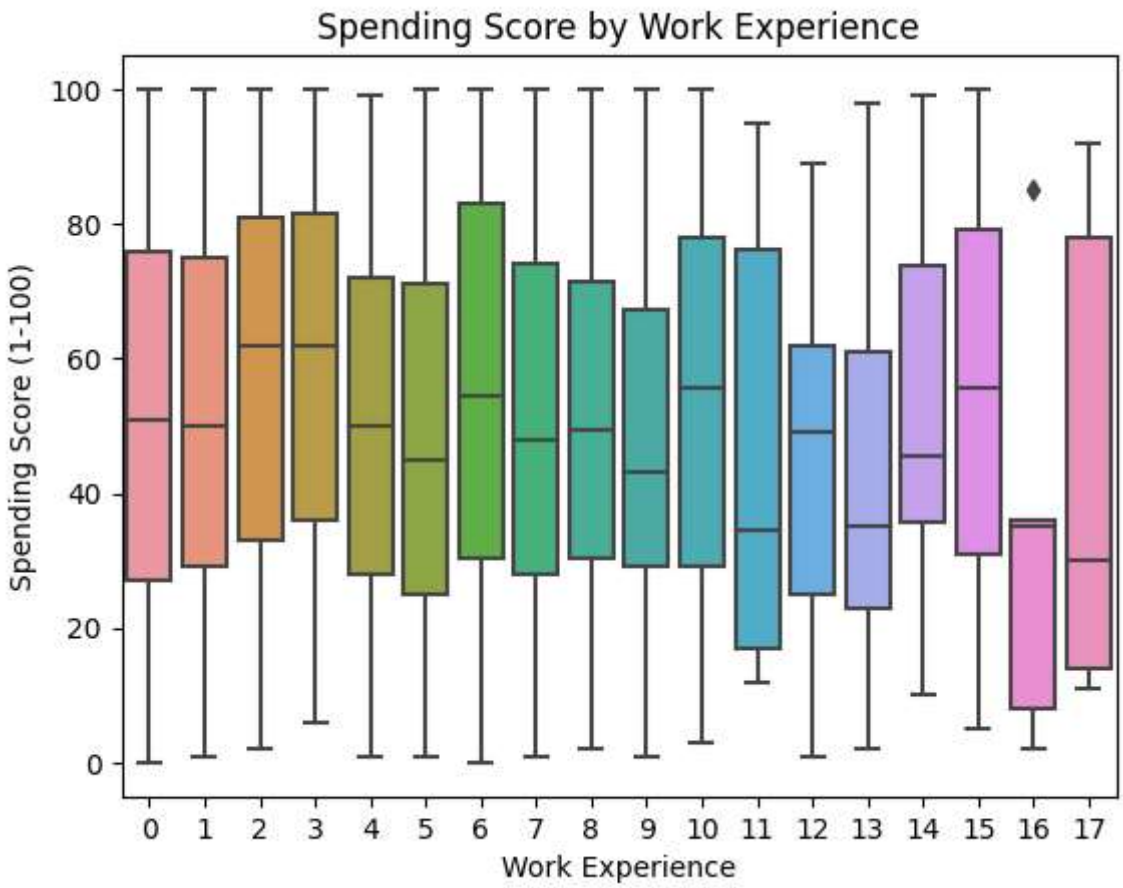
In [ ]:
```
# spending behavior by income
plt.hexbin(x='Annual Income ($)', y='Spending Score (1-100)', data=df,gridsize=20, cma
plt.xlabel('Annual Income ($)')
plt.xticks(rotation=45)
plt.ylabel('Spending Score (1-100)')
plt.title('Spending Score by Income')
plt.colorbar()
plt.show()
```



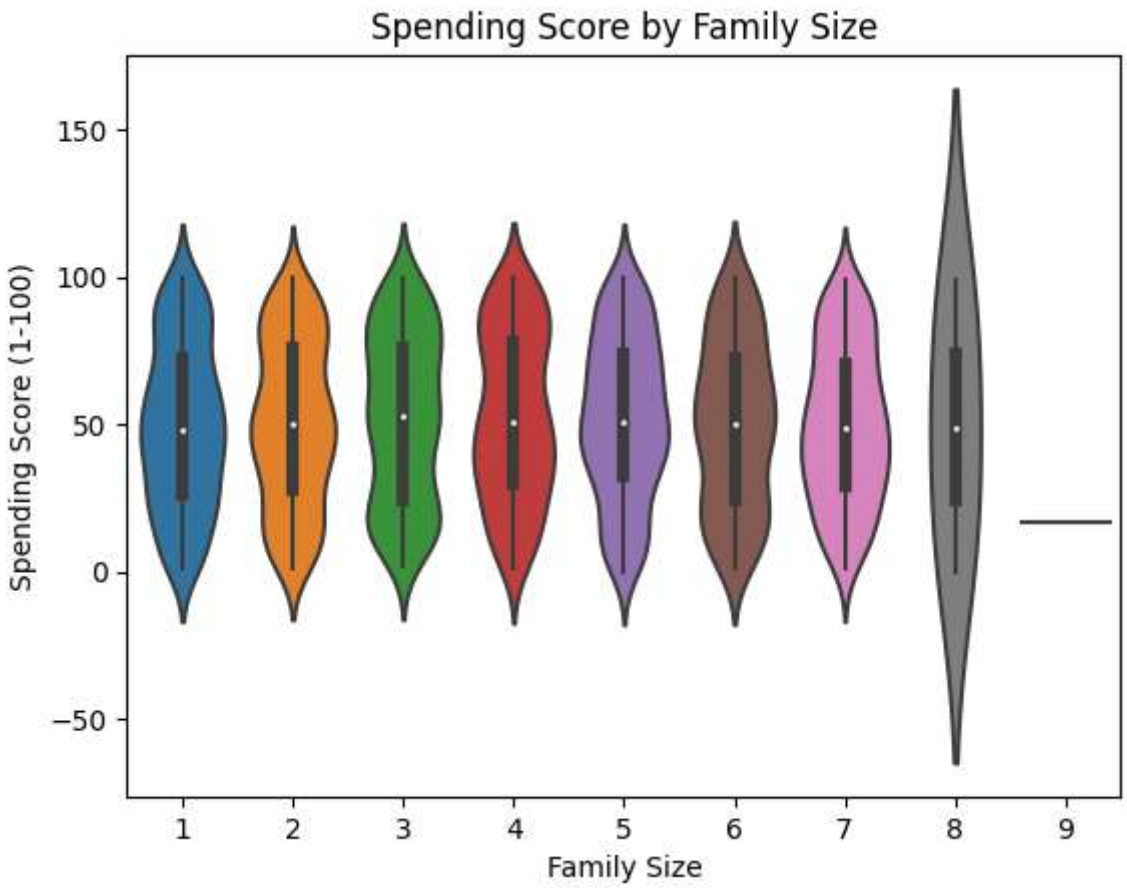
In [ ]:
```
# spending behavior by profession
sns.barplot(x='Profession', y='Spending Score (1-100)', data=df)
plt.xticks(rotation=45)
plt.title('Spending Score by Profession')
plt.show()
```



In [ ]:
```
# spending behavior by work experience
sns.boxplot(x='Work Experience', y='Spending Score (1-100)', data=df)
plt.title('Spending Score by Work Experience')
plt.show()
```



In [ ]:
```
# spending behavior by family size
sns.violinplot(x='Family Size', y='Spending Score (1-100)', data=df)
plt.title('Spending Score by Family Size')
plt.show()
```


Spending Score by Family Size
Spending Score (1-100)
150
100
50
0
−50
1
2
3
4
5
6
7
8
9
Family Size

In [ ]: