```
# Name:Atharva Kangralkar
# Class:CS(AIML) - A
# Roll: 54
# PRN:12311493
# LINK:- https://colab.research.google.com/drive/1eksVrNHhmyMTobQyfJgck_qwK2Zrr5e3?usp=sharing
```

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

```
df = pd.read_csv("/content/Customers.csv")
```

```
#Q1) Show first 5 lines of Customers dataset.
df.head()
```

	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

```
#Q2) Find out detail information Customers dataset.
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              1965 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
```

```
#Q3) Show the preview of numerical columns in Customers dataset.
df.describe()
```

	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

```
#Q4) Find out shape of Customers dataset.
df.shape
```

```
(2000, 8)
```

```
#Q5) Find out mean of Age column in Customers dataset.
df["Age"].mean()
```

```
48.96
```

```
#Q6) Find out mean of Annual Income column in Customers dataset.
df["Annual Income ($)"].mean()
```

```
110731.8215
```

```
# Q7) Find out median of Age column in Customers dataset.
df["Age"].median()
```

```
48.0
```

```
# Q8) Find out median of Annual Income column in Customers dataset.
df["Annual Income ($)"].median()
```

```
110045.0
```

```
len(df["Age"].unique())
```

```
100
```

```
df["Age"].unique()
```

```
array([19, 21, 20, 23, 31, 22, 35, 64, 30, 67, 58, 24, 37, 52, 25, 46, 54,
       29, 45, 40, 60, 53, 18, 49, 42, 36, 65, 48, 50, 27, 33, 59, 47, 51,
       69, 70, 63, 43, 68, 32, 26, 57, 38, 55, 34, 66, 39, 44, 28, 56, 41,
       16, 76, 62, 80,  1,  0, 86, 79, 83, 95, 93, 78, 15,  6, 84,  4, 91,
       14, 92, 77, 89, 12,  7, 94, 96, 74, 85, 73,  9, 10, 11, 17, 90, 61,
       13, 72,  5, 75, 99, 88, 82,  8, 87,  3, 97, 81, 98,  2, 71])
```

```
# Q9) Find summary statistics of income grouped by the age groups. (groupby function)
df["Age Group"] = pd.cut(df["Age"], bins=10)  # Automatically creates 5 age groups
print(df.groupby("Age Group")["Annual Income ($)"].describe())
```

```
                 count           mean           std      min       25%  \
Age Group
(-0.099, 9.9]  185.0  119410.983784  41426.998555  12000.0  84000.00
(9.9, 19.8]    195.0  115841.379487  47308.971630   2000.0  78451.00
(19.8, 29.7]   211.0  103448.559242  47674.365796      0.0  67223.50
(29.7, 39.6]   241.0  102080.908714  47720.394249      0.0  66000.00
(39.6, 49.5]   195.0  104140.492308  49760.890582   2000.0  68833.00
(49.5, 59.4]   202.0  107900.287129  44234.408613   4000.0  74101.00
(59.4, 69.3]   210.0  113624.961905  47545.317324   7000.0  77544.25
(69.3, 79.2]   167.0  110549.832335  43579.341673   3000.0  76497.50
(79.2, 89.1]   200.0  119412.955000  39193.272793   7000.0  93279.25
(89.1, 99.0]   194.0  113636.536082  43453.084933   1000.0  75766.25

                     50%        75%       max
Age Group
(-0.099, 9.9]  123804.0  151298.00  189709.0
(9.9, 19.8]    116822.0  156413.50  189689.0
(19.8, 29.7]    99950.0  145981.00  189650.0
(29.7, 39.6]    96794.0  144485.00  189630.0
(39.6, 49.5]    98000.0  147020.00  187898.0
(49.5, 59.4]   104533.5  145249.00  189672.0
```

```
(59.4, 69.3]   116548.5  152871.75  189945.0
(69.3, 79.2]   106681.0  147867.50  187141.0
(79.2, 89.1]   122020.0  150055.75  188719.0
(89.1, 99.0]   110019.0  152188.75  189974.0
<ipython-input-24-2bf85db7b61a>:3: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass
  print(df.groupby("Age Group")["Annual Income ($)"].describe())
```

```
# Q10) Find summary statistics of income grouped by the Profession groups. ( groupby function)
df.groupby("Profession")["Annual Income ($)"].describe()
```

Profession	count	mean	std	min	25%	50%	75%	max
Artist	612.0	108776.580065	45430.821575	0.0	73122.75	105211.0	146370.50	189709.0
Doctor	161.0	111573.217391	48261.233502	0.0	73892.00	111871.0	154821.00	189672.0
Engineer	179.0	111161.240223	46503.822115	7000.0	76260.50	112766.0	151230.00	189974.0
Entertainment	234.0	110650.333333	45001.884572	1000.0	73497.50	109446.0	148479.50	186882.0
Executive	153.0	113770.130719	45434.149328	4000.0	77873.00	112957.0	150782.00	189630.0
Healthcare	339.0	112574.041298	45426.143104	3000.0	76475.50	111717.0	151050.50	189689.0
Homemaker	60.0	108758.616667	40393.442633	29000.0	78213.25	100387.0	135993.75	188696.0
Lawyer	142.0	110995.838028	47793.706749	3000.0	74277.25	113338.5	150881.25	189650.0
Marketing	85.0	107994.211765	48772.573140	5000.0	65483.00	120899.0	145704.00	186069.0

```
# Q11) Find out the mode of Age column in Customers dataset.
df["Age"].mode()
```

	Age
0	31

```
# Q12) Find out count value based on Age groups.
df["Age Group"].value_counts()
```

Age Group	count
(29.7, 39.6]	241
(19.8, 29.7]	211
(59.4, 69.3]	210
(49.5, 59.4]	202
(79.2, 89.1]	200
(9.9, 19.8]	195
(39.6, 49.5]	195
(89.1, 99.0]	194
(-0.099, 9.9]	185
(69.3, 79.2]	167

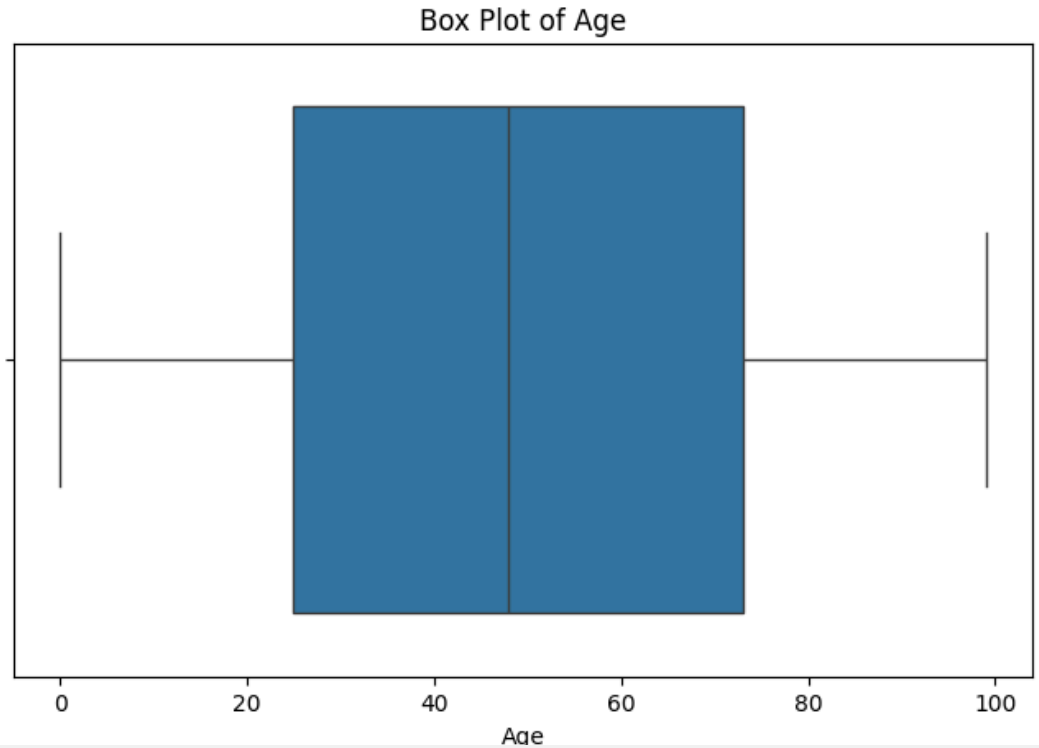
```
# Q13) Find out the variance of Age column in Customers dataset.
df["Age"].var()
```

```
808.2505252626332
```

```
# Q14) Find out the standard deviation of Age column in Customers dataset.
df["Age"].std()
```

```
28.429747189565955
```

```
# Q15) Draw box plot. Use Age column.
plt.figure(figsize=(8,5))
sns.boxplot(x=df["Age"])
plt.title("Box Plot of Age")
plt.show()
```




```
# Q16) Use Range function to find out range of Age and Annual Income column.
age_range = df["Age"].max() - df["Age"].min()
income_range = df["Annual Income ($)"].max() - df["Annual Income ($)"].min()
print(f"Age Range: {age_range}")
print(f"Annual Income Range: {income_range}")
```

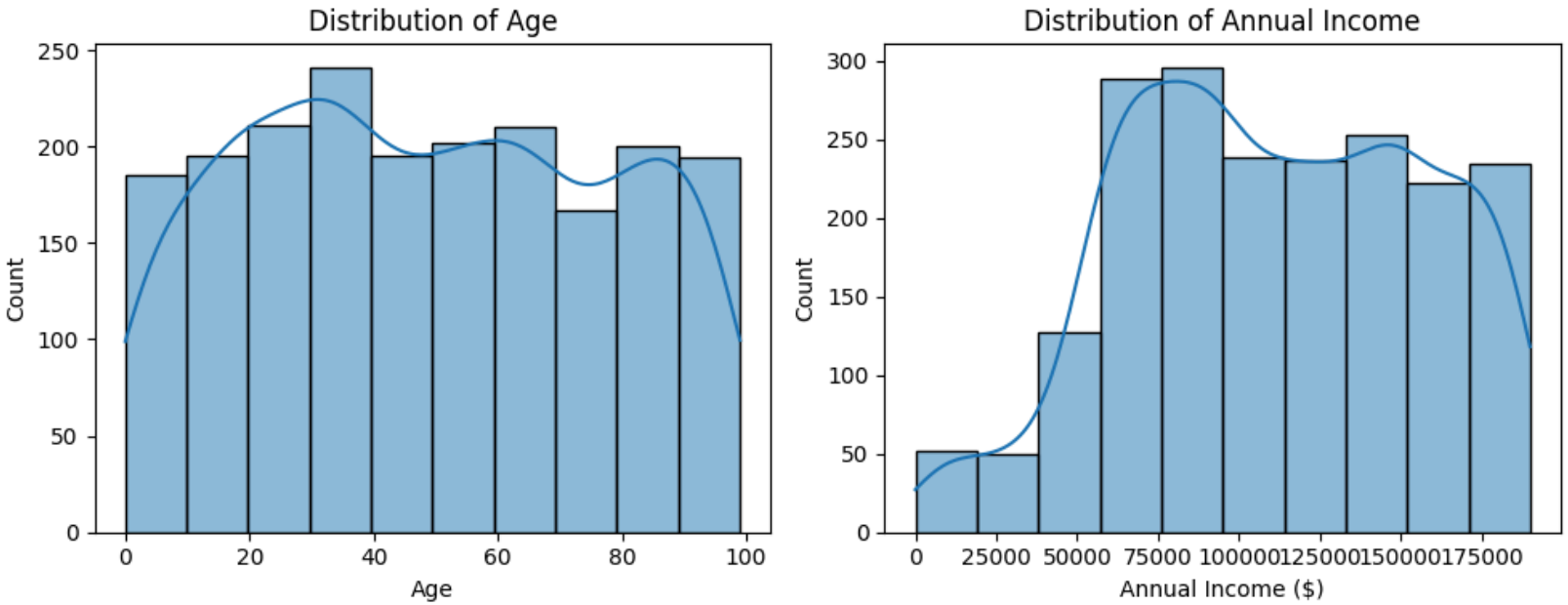
```
Age Range: 99
Annual Income Range: 189974
```

```
# Q17) Use aggerate function and find out statistics for Age and Annual Income column.
print(df[["Age", "Annual Income ($)"]].agg(["mean", "median", "std", "var", "min", "max"]))
```

```
               Age  Annual Income ($)
mean     48.960000       1.107318e+05
median   48.000000       1.100450e+05
std      28.429747       4.573954e+04
var     808.250525       2.092105e+09
min       0.000000       0.000000e+00
```

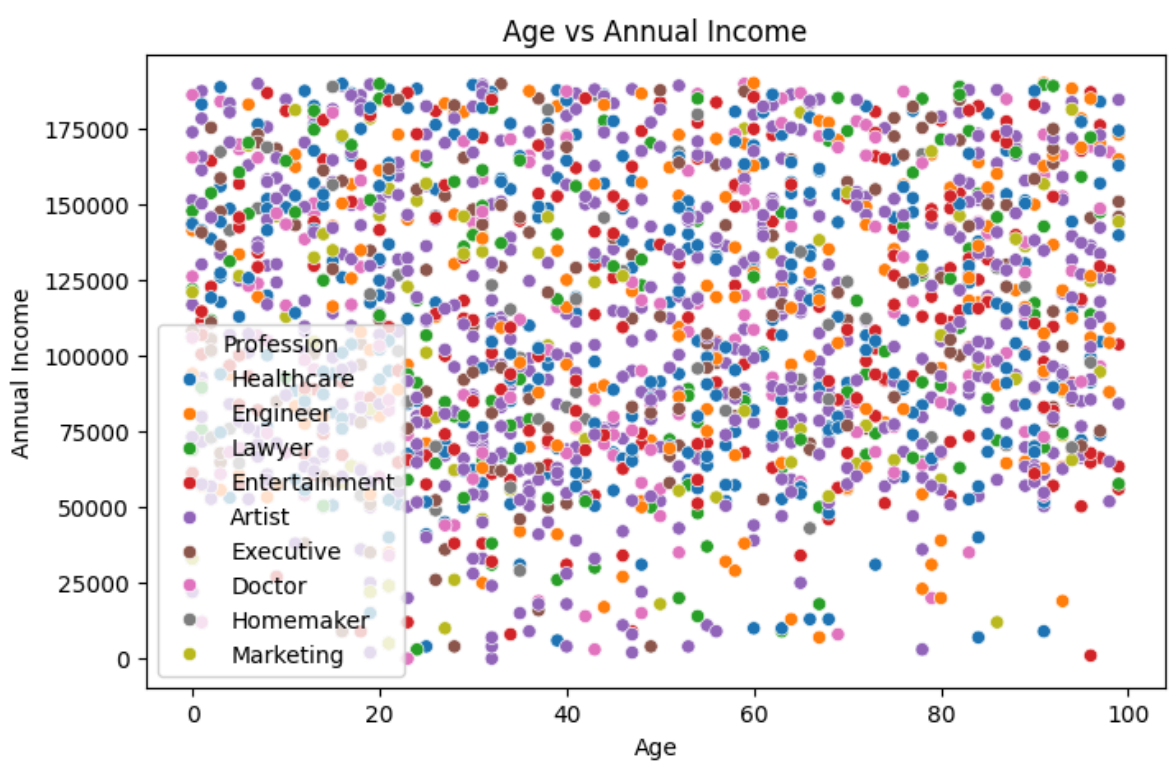
```
max      99.000000       1.899740e+05
```

```
# Q18) Perform Univariate and Bivariate analysis on Customers dataset.
plt.figure(figsize=(10,4))

plt.subplot(1,2,1)
sns.histplot(df["Age"], bins=10, kde=True)
plt.title("Distribution of Age")

plt.subplot(1,2,2)
sns.histplot(df["Annual Income ($)"], bins=10, kde=True)
plt.title("Distribution of Annual Income")

plt.tight_layout()
plt.show()
```




```
plt.figure(figsize=(8,5))
sns.scatterplot(x=df["Age"], y=df["Annual Income ($)"], hue=df["Profession"])
plt.title("Age vs Annual Income")
plt.xlabel("Age")
plt.ylabel("Annual Income")
plt.show()
```

```
print(df[["Age", "Annual Income ($)"]].corr())
```




```
                     Age  Annual Income ($)
Age                1.000000           0.021378
```

Start coding or generate with AI.