ASSIGNMENT 3

TITLE: DESCRIPTIVE STATISTICS- MEASURES OF CENTRAL TENDENCY AND VARIABILITY

PROBLEM STATEMENT: -

Perform the following operations on any open-source dataset (e.g., data.csv)

1. Provide summary statistics (mean, median, minimum, maximum, standard deviation) for a dataset (age, income etc.) with numeric variables grouped by one of the qualitative (categorical) variable. For example, if your categorical variable is age groups and quantitative variable is income, then provide summary statistics of income grouped by the age groups. Create a list that contains a numeric value for each response to the categorical variable.
2. Write a Python program to display some basic statistical details like percentile, mean, standard deviation etc. of the species of 'Iris-setosa', 'Iris-versicolor' and 'Iris- versicolor' of iris.csv dataset.

Provide the codes with outputs and explain everything that you do in this step.

OBJECTIVE: To analyze and demonstrate knowledge of statistical data analysis techniques for decision-making

PREREQUISITE: -

1. Basic of Python Programming
2. Concept of statistics mean, median, minimum, maximum, standard deviation

THEORY:

The data are summarized in some, but not all ways. We chose descriptives that are either most often reported, or most often covered in introductory courses. These are as follows:

- **Central Tendency**
  - Mean
  - Median
  - Mode
- **Dispersion**
  - Standard deviation (Std. deviation)
  - Minimum
  - Maximum

- **Central Tendency**

The Mean, Median and Mode are the three measures of central tendency. Mean is the arithmetic average of a data set. This is found by adding the numbers in a data set and dividing by the number of observations in

the data set. The median is the middle number in a data set when the numbers are listed in either ascending or descending order. The mode is the value that occurs the most often in a data set and the range is the difference between the highest and lowest values in a data set.

The Mean

$$\overline{x} = \frac{\sum x}{N}$$

Here,
∑ represents the summation
X represents observations
N represents the number of observations .

In the case where the data is presented in a tabular form, the following formula is used to compute the mean

Mean = ∑f x / ∑f

Where ∑f = N

The Median

If the total number of observations (n) is an odd number, then the formula is given below:

$$Median = \left(\frac{n+1}{2}\right)^{th} observation$$

If the total number of the observations (n) is an even number, then the formula is given below:

$$Median = \frac{\left(\frac{n}{2}\right)^{th} observation + \left(\frac{n}{2}+1\right)^{th} observation}{2}$$

Consider the case where the data is continuous and presented in the form of a frequency distribution, the median formula is as follows.

Find the median class, the total count of observations ∑f.

The median class consists of the class in which (n / 2) is present.

$$\text{Median } = 1 + \left[\frac{\frac{n}{2} - c}{f}\right] \times h$$

Here

l = lesser limit belonging to the median class

c = cumulative frequency value of the class before the median class

f = frequency possessed by the median class

h = size of the class

The Mode

The mode is the most frequently occurring observation or value.

Consider the case where the data is continuous, and the value of mode can be computed using the following steps.

a] Determine the modal class that is the class possessing the maximum frequency.

b] Calculate the mode using the below formula.

$$\text{Mode} = 1 + \left[\frac{f_m - f_1}{2f_m - f_1 - f_2}\right] \times h$$

l = lesser limit of modal class

fm = frequency possessed by the modal class

f1 = frequency possessed by the class before the modal class

f2 = frequency possessed by the class after the modal class

h = width of the class

Standard deviation

$$s = \sqrt{\frac{\Sigma (X - \bar{x})^2}{n - 1}}$$

- s = sample standard deviation
- Σ = sum of...
- X = each value
- $\bar{x}$ = sample mean
- n = number of values in the sample

Steps for calculating the standard deviation.

The standard deviation is usually calculated automatically by whichever software you use for your statistical analysis. But you can also calculate it by hand to better understand how the formula works.

There are six main steps for finding the standard deviation by hand. We'll use a small data set of 6 scores to walk through the steps.

Data set					
46	69	32	60	52	41

3. Step 1: Find the mean.

To find the mean, add up all the scores, then divide them by the number of scores.

Mean ($\bar{x}$)
$\bar{x}$ = (46 + 69 + 32 + 60 + 52 + 41) ÷ 6 = **50**

4. Step 2: Find each score's deviation from the mean

Subtract the mean from each score to get the deviations from the mean.

Since $\bar{x}$ = 50, here we take away 50 from each score.

Score	Deviation from the mean
46	46 – 50 = **-4**
69	69 – 50 = **19**
32	32 – 50 = **-18**
60	60 – 50 = **10**
52	52 – 50 = **2**
41	41 – 50 = **-9**

5. Step 3: Square each deviation from the mean.

Multiply each deviation from the mean by itself. This will result in positive numbers.

Squared deviations from the mean
$(-4)^2 = 4 \times 4 =$ **16**
$19^2 = 19 \times 19 =$ **361**
$(-18)^2 = -18 \times -18 =$ **324**
$10^2 = 10 \times 10 =$ **100**

$2^2 = 2 \times 2 =$ **4**
$(-9)^2 = -9 \times -9 =$ **81**

6. Step 4: Find the sum of squares.

Add up all of the squared deviations. This is called the sum of squares.

Sum of squares
16 + 361 + 324 + 100 + 4 + 81 = **886**

7. Step 5: Find the variance.

Divide the sum of the squares by $n - 1$ (for a sample) or N (for a population) – this is the variance.

Since we're working with a sample size of 6, we will use $n - 1$, where $n = 6$.

Variance
886 ÷ (6 – 1) = 886 ÷ 5 = **177.2**

8. Step 6: Find the square root of the variance.

To find the standard deviation, we take the square root of the variance.

Standard deviation
√177.2 = **13.31**

From learning that $SD = 13.31$, we can say that each score deviates from the mean by 13.31 points on average.

Consider HR dataset it contains fields like Age, Monthly Income, Attrition, Business Travel and so on so following are steps to find statistics (mean, median, minimum, maximum, standard deviation)

```
#Mean of monthly income and age
print("The mean of monthly income is :",df.loc[:,"MonthlyIncome"].mean())
print("The mean of age is :",df.loc[:,"Age"].mean())

#Mode of monthly income and age
print("The median of monthly income is :",df.loc[:,"MonthlyIncome"].median())
```

```
print("The median of age is :",df.loc[:,"Age"].median())

#Median of monthly income and age
print("The mode of monthly income is :",df.loc[:,"MonthlyIncome"].mode())
print("The mode of age is :",df.loc[:,"Age"].mode())

#Standard deviation of monthly income and age
print("The standard deviation of monthly income is :",df.loc[:,"MonthlyIncome"].std())
print("The standard deviation of age is :",df.loc[:,"Age"].std())

#Storing age and monthly income in array and then finding maximum and minimum values
array1 = np.array(df['MonthlyIncome'])
array2=np.array(df["Age"])
print("Income",array1)
print("Age array",array2)
print("Maximum income among the employees is :",max(array1))
print("Minimum income among the employees is :",min(array1))
print("Maximum age among the employees is :",max(array2))
print("Minimum age among the employees is :",min(array2))

# Replacing the categorical values by numeric values
df.head()
df["BusinessTravel"].replace({"Travel_Rarely":1, "Travel_Frequently":0}, inplace=True)
df["Attrition"].replace({ "Yes":1, "No":0}, inplace=True)
df.head()
```

##Use of describe()

To display basic statistical details like percentile, mean, standard deviation etc. for Iris-Vigginica use describe.

```
Iris-setosa
       Sepal_Length  Sepal_Width  Petal_Length  Petal_Width
count      50.00000    50.000000     50.000000     50.00000
mean        5.00600     3.418000      1.464000      0.24400
std         0.35249     0.381024      0.173511      0.10721
min         4.30000     2.300000      1.000000      0.10000
25%         4.80000     3.125000      1.400000      0.20000
50%         5.00000     3.400000      1.500000      0.20000
75%         5.20000     3.675000      1.575000      0.30000
max         5.80000     4.400000      1.900000      0.60000
Iris-versicolor
       Sepal_Length  Sepal_Width  Petal_Length  Petal_Width
count     50.000000    50.000000     50.000000    50.000000
mean       5.936000     2.770000      4.260000     1.326000
std        0.516171     0.313798      0.469911     0.197753
min        4.900000     2.000000      3.000000     1.000000
25%        5.600000     2.525000      4.000000     1.200000
50%        5.900000     2.800000      4.350000     1.300000
75%        6.300000     3.000000      4.600000     1.500000
max        7.000000     3.400000      5.100000     1.800000
Iris-virginica
       Sepal_Length  Sepal_Width  Petal_Length  Petal_Width
count      50.00000    50.000000     50.000000     50.00000
mean        6.58800     2.974000      5.552000      2.02600
std         0.63588     0.322497      0.551895      0.27465
min         4.90000     2.200000      4.500000      1.40000
25%         6.22500     2.800000      5.100000      1.80000
50%         6.50000     3.000000      5.550000      2.00000
75%         6.90000     3.175000      5.875000      2.30000
max         7.90000     3.800000      6.900000      2.50000
```

Conclusion: In this way we have explored the functions of the python library for calculating Data statistics (mean, median, minimum, maximum, standard deviation).

Assignment Question:

1. How to find Mean Mode and Median.
2. How to find standard deviation.
3. Write a Python program to display some basic statistical details like percentile, mean, standard deviation etc.
4. What is use of Describe () Command?

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

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

In [3]:
```
df.head(10)
```

Out[3]:

	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
5	6	Female	22	17	76
6	7	Female	35	18	6
7	8	Female	23	18	94
8	9	Male	64	19	3
9	10	Female	30	19	72

In [4]:
```
df.columns
```

Out[4]:
```
Index(['CustomerID', 'Genre', 'Age', 'Annual Income (k$)',
       'Spending Score (1-100)'],
      dtype='object')
```

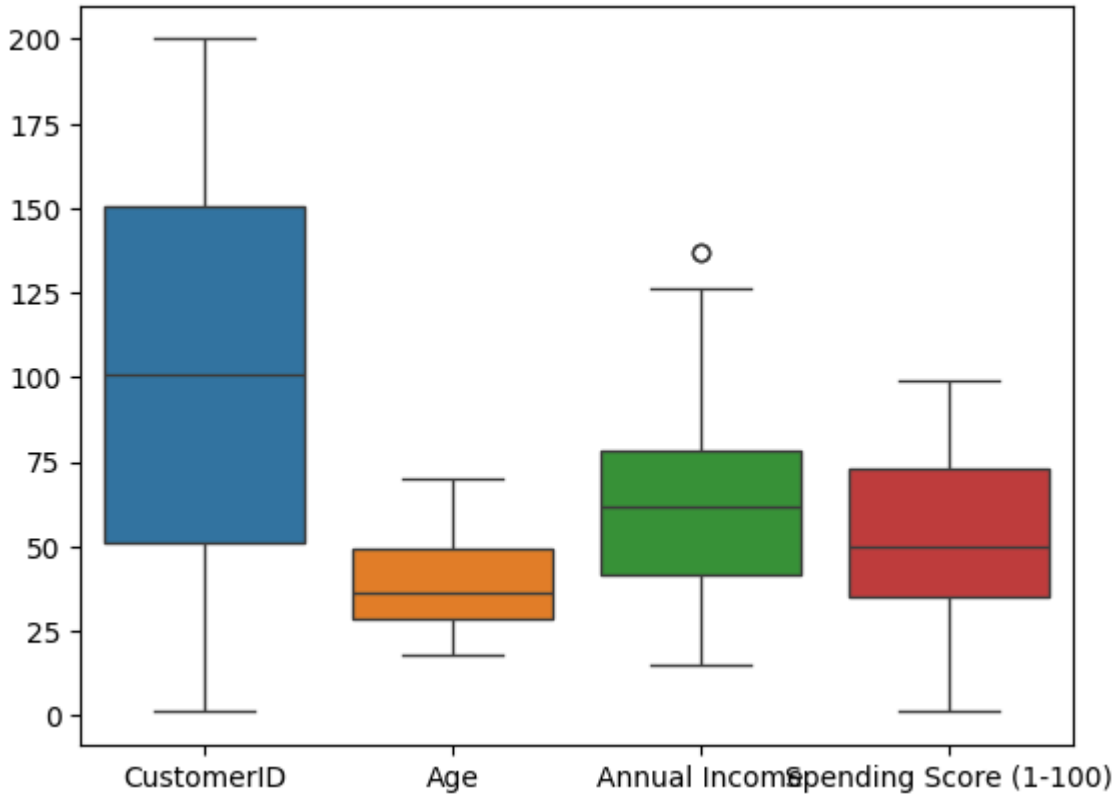
In [5]:
```
df = df.rename(columns={

    "Annual Income (k$)": "Annual Income"
})
```

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
CustomerID              0
Genre                   0
Age                     0
Annual Income           0
Spending Score (1-100)  0
dtype: int64
```

In [7]:
```
sns.boxplot(df)
```

Out[7]:
```
<Axes: >
```



In [8]:
```
bins = [0, 24, 54, float('inf')]
labels = ['Youth', 'Adult', 'Old']
df['Age Group'] = pd.cut(df['Age'], bins=bins, labels=labels, right=False)
```

In [9]:
```
summary_stats = df.groupby('Age Group')[['Annual Income', 'Spending Score (1-100
```

```
C:\Users\91705\AppData\Local\Temp\ipykernel_15640\563290720.py:1: FutureWarning:
The default of observed=False is deprecated and will be changed to True in a futu
re version of pandas. Pass observed=False to retain current behavior or observed=
True to adopt the future default and silence this warning.
  summary_stats = df.groupby('Age Group')[['Annual Income', 'Spending Score (1-10
0)']].agg(['mean', 'median', 'min', 'max', 'std'])
```

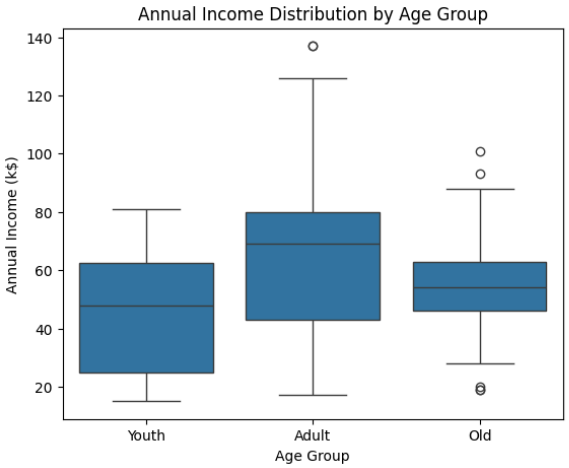
In [10]:
```
summary_stats
```

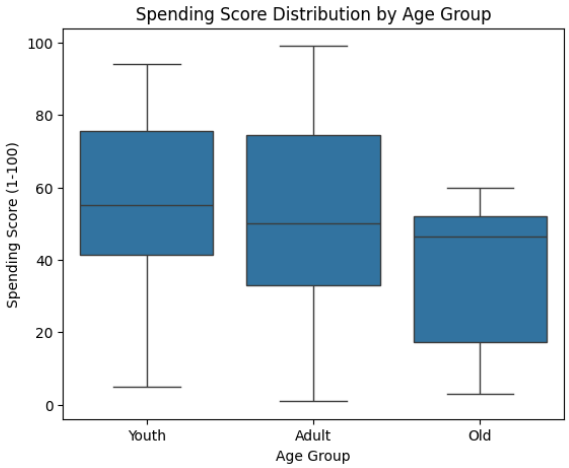
Out[10]:

	Annual Income					Spending Score (1-				
	mean	median	min	max	std	mean	median	min	max	
Age Group										
Youth	45.354839	48.0	15	81	21.143239	54.290323	55.0	5	94	25.001
Adult	65.577778	69.0	17	137	27.258975	52.170370	50.0	1	99	26.867
Old	54.500000	54.0	19	101	19.448845	38.647059	46.5	3	60	18.780

In [11]:
```
fig, axes = plt.subplots(1, 2, figsize=(14, 5))

sns.boxplot(x="Age Group", y="Annual Income", data=df, ax=axes[0])
axes[0].set_title("Annual Income Distribution by Age Group")
```

```
axes[0].set_ylabel("Annual Income (k$)")


sns.boxplot(x="Age Group", y="Spending Score (1-100)", data=df, ax=axes[1],
            )
axes[1].set_title("Spending Score Distribution by Age Group")
axes[1].set_ylabel("Spending Score (1-100)")

plt.show()
```





Iris Statistical Details

In [12]:
```
iris=pd.read_csv("Iris.csv")
```

In [13]:
```
iris.head()
```

Out[13]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [14]:
```
iris.columns
```

Out[14]:
```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

In [21]:
```
iris['Species'].value_counts()
```

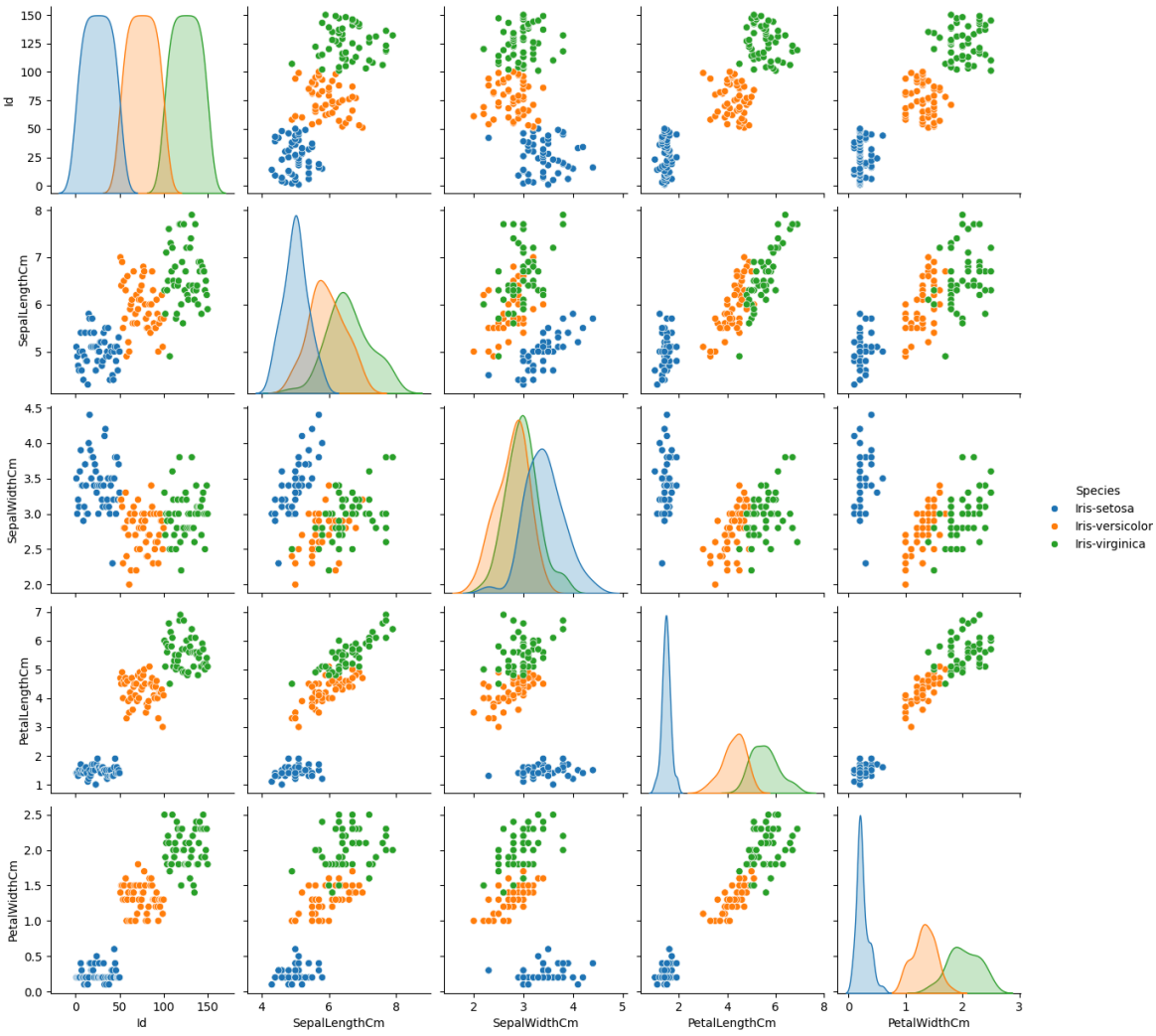
Out[21]:
```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: count, dtype: int64
```

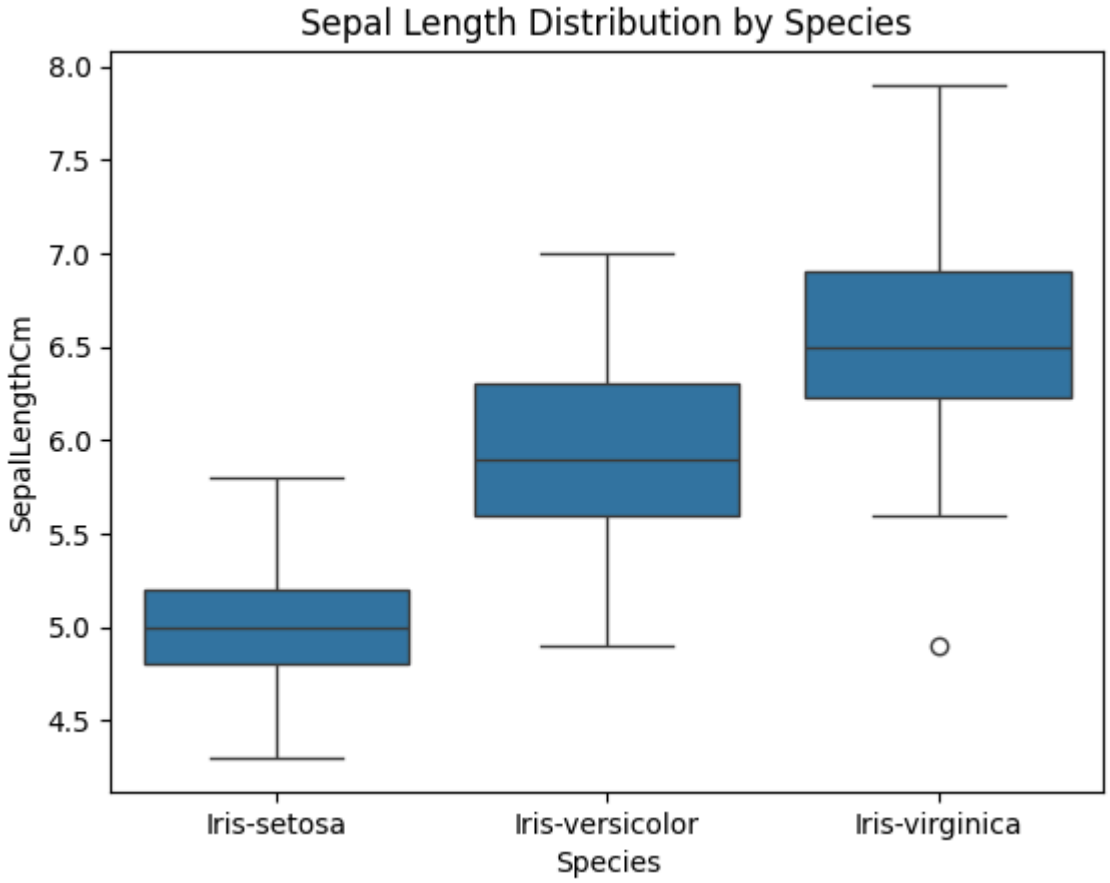
In [22]:
```
species_list=['Iris-setosa','Iris-versicolor','Iris-virginica']
```

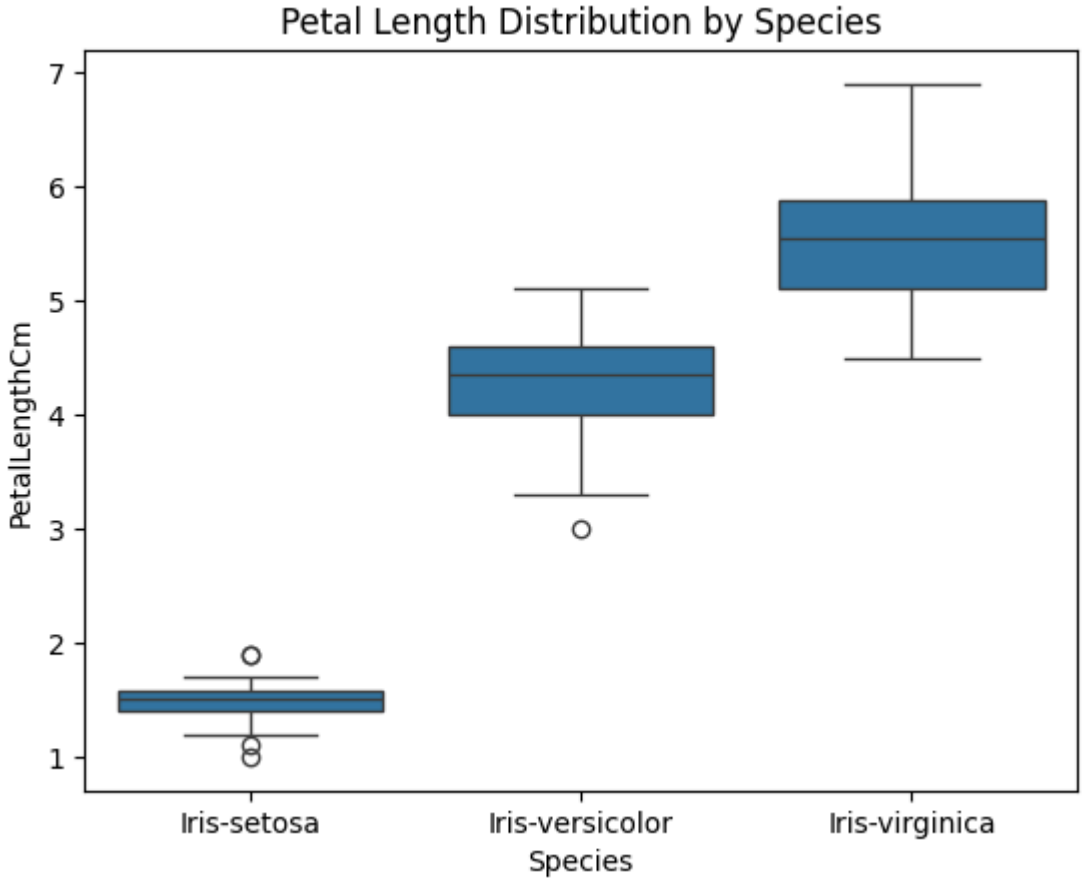
In [25]:
```
for species in species_list:
    print(f"\nStatistics for {species}:")
    species_data = iris[iris['Species'] == species].drop(columns=['Species'])
    print(species_data.describe())
```

```
Statistics for Iris-setosa:
             Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count  50.00000       50.00000     50.000000      50.000000      50.00000
mean   25.50000        5.00600      3.418000       1.464000       0.24400
std    14.57738        0.35249      0.381024       0.173511       0.10721
min     1.00000        4.30000      2.300000       1.000000       0.10000
25%    13.25000        4.80000      3.125000       1.400000       0.20000
50%    25.50000        5.00000      3.400000       1.500000       0.20000
75%    37.75000        5.20000      3.675000       1.575000       0.30000
max    50.00000        5.80000      4.400000       1.900000       0.60000

Statistics for Iris-versicolor:
              Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count   50.00000       50.000000     50.000000      50.000000     50.000000
mean    75.50000        5.936000      2.770000       4.260000      1.326000
std     14.57738        0.516171      0.313798       0.469911      0.197753
min     51.00000        4.900000      2.000000       3.000000      1.000000
25%     63.25000        5.600000      2.525000       4.000000      1.200000
50%     75.50000        5.900000      2.800000       4.350000      1.300000
75%     87.75000        6.300000      3.000000       4.600000      1.500000
max    100.00000        7.000000      3.400000       5.100000      1.800000

Statistics for Iris-virginica:
              Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count   50.00000       50.00000     50.000000      50.000000      50.00000
mean   125.50000        6.58800      2.974000       5.552000       2.02600
std     14.57738        0.63588      0.322497       0.551895       0.27465
min    101.00000        4.90000      2.200000       4.500000       1.40000
25%    113.25000        6.22500      2.800000       5.100000       1.80000
50%    125.50000        6.50000      3.000000       5.550000       2.00000
75%    137.75000        6.90000      3.175000       5.875000       2.30000
max    150.00000        7.90000      3.800000       6.900000       2.50000
```

In [26]:
```
sns.pairplot(iris, hue='Species', diag_kind='kde')
plt.show()

sns.boxplot(x='Species', y='SepalLengthCm', data=iris)
plt.title("Sepal Length Distribution by Species")
plt.show()

sns.boxplot(x='Species', y='PetalLengthCm', data=iris)
plt.title("Petal Length Distribution by Species")
plt.show()
```


Id
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
Iris-setosa
Iris-versicolor
Iris-virginica


Sepal Length Distribution by Species
SepalLengthCm
Iris-setosa
Iris-versicolor
Iris-virginica
Species


Petal Length Distribution by Species
PetalLengthCm
7
6
5
4
3
2
1
Iris-setosa
Iris-versicolor
Iris-virginica
Species