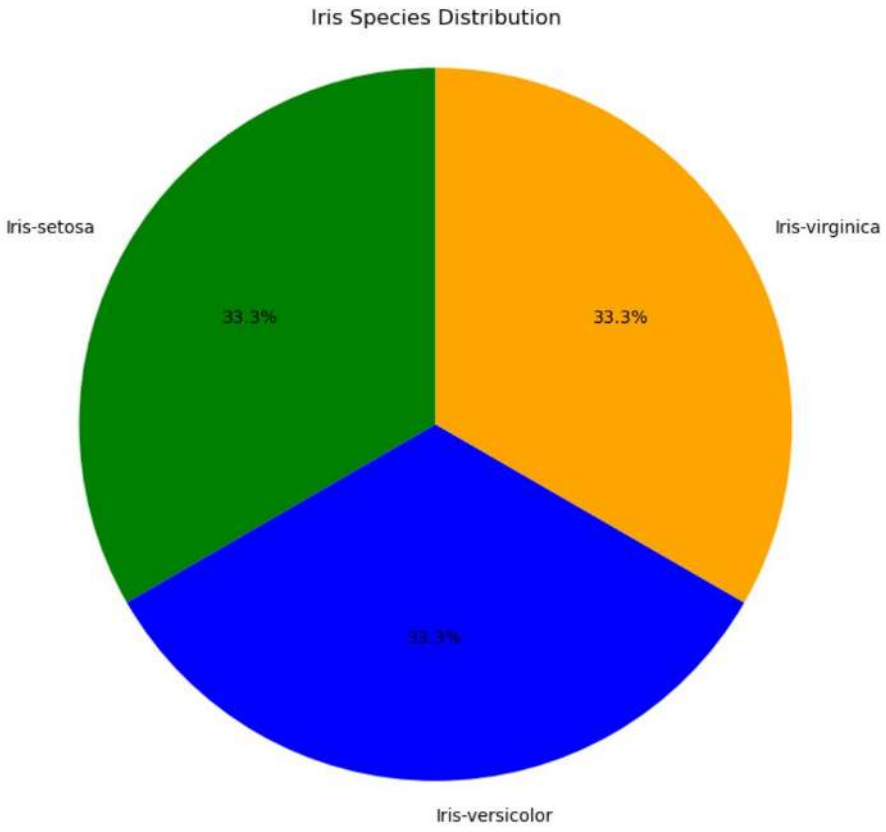
SLIP 1

Q.2 A) Write a Python program to create a Pie plot to get the frequency of the three species of the Iris data (Use iris.csv)

In [8]:

```python
from sklearn.datasets import load_iris
import pandas as pd
import matplotlib.pyplot as plt

# Load the Iris data from the CSV file
iris_data = pd.read_csv('Iris.csv')
# Get the frequency of each species
species_counts = iris_data['Species'].value_counts()

# # Create a Pie plot
plt.figure(figsize=(8, 8))
plt.pie(species_counts, labels=species_counts.index, autopct='%1.1f%%', startangle=90, colors=['green', 'blue', 'orange'])
plt.title('Iris Species Distribution')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()

```



B) Write a Python program to view basic statistical details of the data.(Use wineequality-red.csv)

In [9]:
```
import pandas as pd

# Load the wine data from the CSV file
wine_data = pd.read_csv('winequality-red.csv')

# Display basic statistical details of the data
statistical_details = wine_data.describe()
print(statistical_details)

```

```
       fixed acidity  volatile acidity  citric acid  residual sugar  \
count      10.000000            10.000    10.000000       10.000000
mean        7.950000             0.631     0.104000        2.330000
std         1.162612             0.161     0.194548        1.376025
min         7.300000             0.280     0.000000        1.200000
25%         7.400000             0.585     0.000000        1.825000
50%         7.650000             0.655     0.010000        1.900000
75%         7.800000             0.700     0.055000        2.225000
max        11.200000             0.880     0.560000        6.100000

       chlorides  free sulfur dioxide  total sulfur dioxide    density  \
count  10.000000            10.000000             10.000000  10.000000
mean    0.077000            14.800000             48.900000   0.997080
std     0.010198             4.467164             25.066356   0.001038
min     0.065000             9.000000             18.000000   0.994600
25%     0.071500            11.500000             34.000000   0.996800
50%     0.075000            15.000000             47.000000   0.997400
75%     0.076000            16.500000             59.750000   0.997800
max     0.098000            25.000000            102.000000   0.998000

              pH  sulphates    alcohol    quality
count  10.000000  10.000000  10.000000  10.000000
mean    3.355000   0.589000   9.700000   5.500000
std     0.127997   0.100161   0.359011   0.849837
min     3.160000   0.460000   9.400000   5.000000
25%     3.270000   0.560000   9.400000   5.000000
50%     3.355000   0.565000   9.650000   5.000000
75%     3.480000   0.632500   9.800000   5.750000
max     3.510000   0.800000  10.500000   7.000000
```

SLIP 2

Q.2 A) Write a Python program for Handling Missing Value. Replace missing value of salary, age column with mean of that column.(Use Data.csv file).

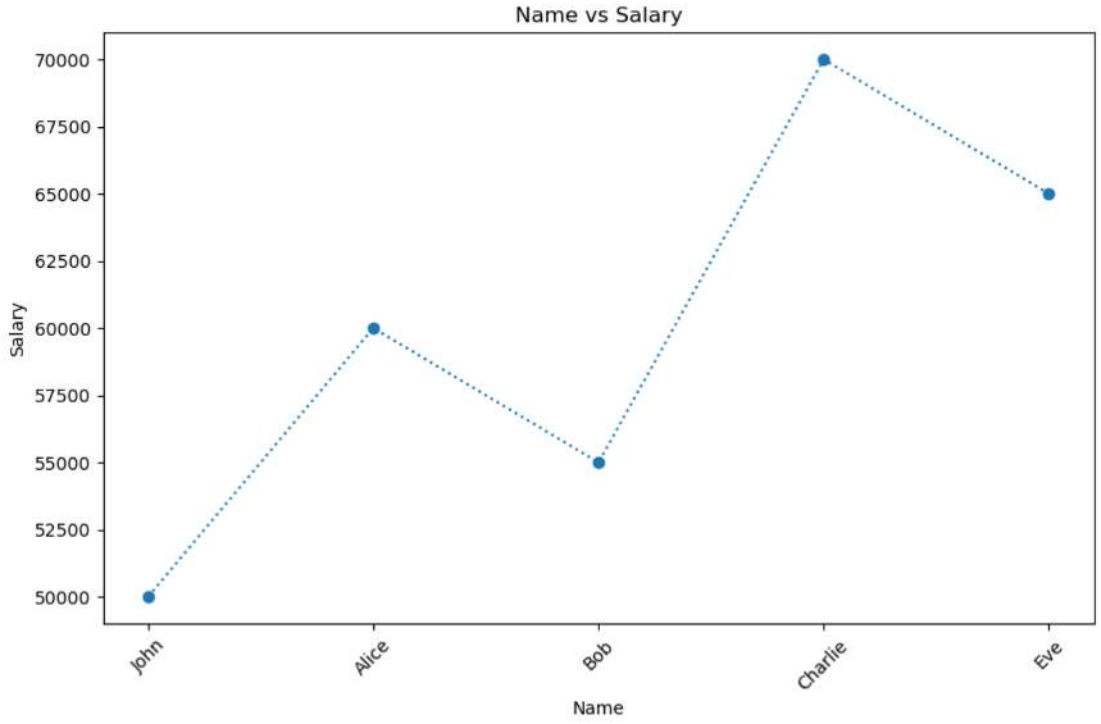
In [10]:
```
import pandas as pd

# Load the data from the CSV file
data = pd.read_csv('Data.csv')

# Display the original data
print("Original Data:")
print(data)

# Calculate the mean of the 'salary' and 'age' columns
mean_salary = data['salary'].mean()
mean_age = data['age'].mean()

# Replace missing values with the mean in the 'salary' and 'age' columns
data['salary'].fillna(mean_salary, inplace=True)
data['age'].fillna(mean_age, inplace=True)

# Display the data after handling missing values
print("\nData after handling missing values:")
print(data)

```

```
Original Data:
      name   salary   age  gender
0     John  50000.0   NaN    Male
1    Alice      NaN  25.0  Female
2      Bob  60000.0  30.0     NaN
3  Charlie      NaN   NaN     NaN
4      Eve  55000.0  28.0  Female

Data after handling missing values:
      name   salary        age  gender
0     John  50000.0  27.666667    Male
1    Alice  55000.0  25.000000  Female
2      Bob  60000.0  30.000000     NaN
3  Charlie  55000.0  27.666667     NaN
4      Eve  55000.0  28.000000  Female
```

Q.2 B) Write a Python program to generate a line plot of name Vs salary

In [16]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the data from the CSV file
data = pd.read_csv('Data1.csv')

# Create a line plot of name versus salary
plt.figure(figsize=(10, 6))
plt.plot(data['name'], data['salary'], linestyle='dotted', marker='o')
plt.title('Name vs Salary')
plt.xlabel('Name')
plt.ylabel('Salary')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability

# Show the plot
plt.show()

```



Q.2 C) Download the heights and weights dataset and load the dataset froma given csv file into a dataframe. Print the first, last 10 rows and random 20 rows also display shape of the dataset.

In [17]:

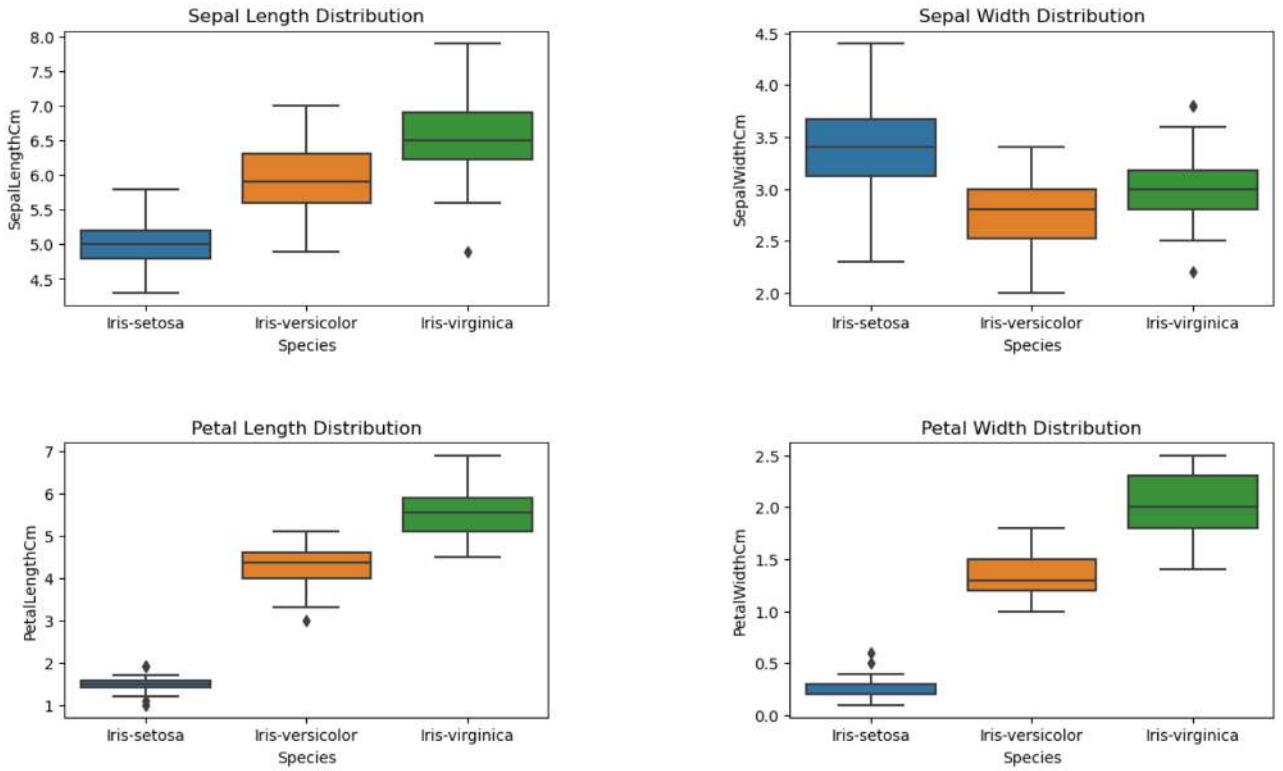
```python
import pandas as pd

# Load the dataset from the CSV file
df = pd.read_csv('height_weight.csv')

# Display the first 10 rows
print("First 10 rows:")
print(df.head(10))

# Display the last 10 rows
print("\nLast 10 rows:")
print(df.tail(10))

# Display random 20 rows
print("\nRandom 20 rows:")
print(df.sample(20))

# Display the shape of the dataset
print("\nShape of the dataset:")
print(df.shape)

```

```
First 10 rows:
   Height  Weight
0     160      55
1     165      60
2     170      65
3     175      70
4     180      75
5     185      80
6     190      85
7     155      50
8     162      57
9     168      62

Last 10 rows:
    Height  Weight
10     172      68
11     178      73
12     182      78
13     188      83
14     195      90
15     150      45
16     158      52
17     164      58
18     169      63
19     174      69

Random 20 rows:
    Height  Weight
15     150      45
0      160      55
5      185      80
4      180      75
14     195      90
3      175      70
8      162      57
12     182      78
7      155      50
17     164      58
10     172      68
11     178      73
2      170      65
16     158      52
18     169      63
13     188      83
9      168      62
6      190      85
19     174      69
1      165      60

Shape of the dataset:
(20, 2)
```

Q.2 A)Write a Python program to create box plots to see how each feature i.e. Sepal Length, Sepal Width, Petal Length, Petal Width are distributed across the three species. (Use iris.csv dataset)

In [36]:

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

# Load the Iris dataset
iris_data = pd.read_csv('iris.csv')

# Create box plots for each feature across the three species
plt.figure(figsize=(14, 8))
plt.subplots_adjust(wspace=0.5,hspace=0.5)  # Adjust the space between subplots for better readability

# Box plot for Sepal Length
plt.subplot(2, 2, 1)
sns.boxplot(x='Species', y='SepalLengthCm', data=iris_data)
plt.title('Sepal Length Distribution')

# Box plot for Sepal Width
plt.subplot(2, 2, 2)
sns.boxplot(x='Species', y='SepalWidthCm', data=iris_data)
plt.title('Sepal Width Distribution')

# Box plot for Petal Length
plt.subplot(2, 2, 3)
sns.boxplot(x='Species', y='PetalLengthCm', data=iris_data)
plt.title('Petal Length Distribution')

# Box plot for Petal Width
plt.subplot(2, 2, 4)
sns.boxplot(x='Species', y='PetalWidthCm', data=iris_data)
plt.title('Petal Width Distribution')


# Show the plots
plt.show()

```



Q.2 B) Write a Python program to view basic statistical details of the data (Use Heights and Weights Dataset)

In [37]:

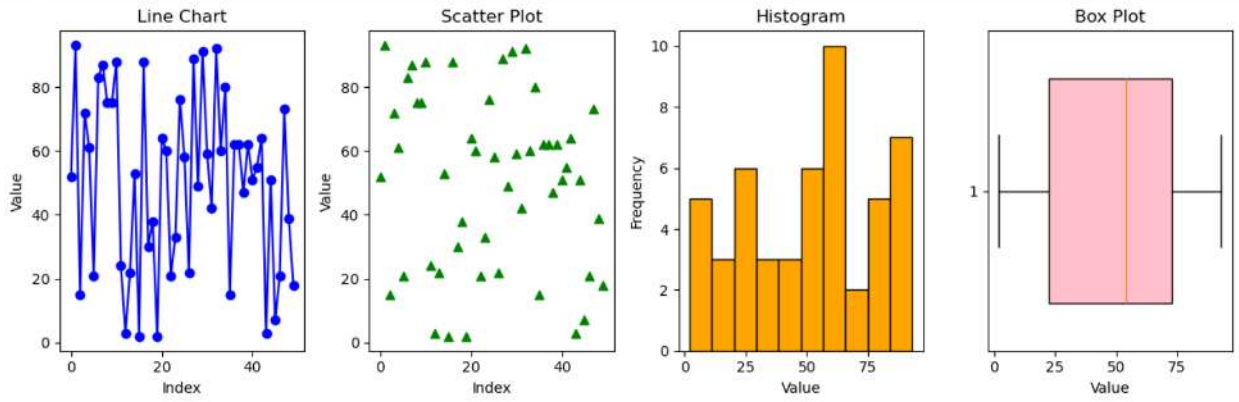
```python
import pandas as pd

# Load the dataset from the CSV file
df = pd.read_csv('height_weight.csv')

# Display basic statistical details of the data
statistical_details = df.describe()
print(statistical_details)

```

```
           Height     Weight
count   20.000000  20.000000
mean   172.000000  66.900000
std     12.238851  12.417729
min    150.000000  45.000000
25%    163.500000  57.750000
50%    171.000000  66.500000
75%    180.500000  75.750000
max    195.000000  90.000000
```

Q.2 A) Generate a random array of 50 integers and display them using a line chart, scatter plot, histogram and box plot. Apply appropriate color, labels and styling options.

In [38]:

```python
import numpy as np
import matplotlib.pyplot as plt

# Set a seed for reproducibility
np.random.seed(42)

# Generate a random array of 50 integers
random_array = np.random.randint(1, 100, 50)

# Line Chart
plt.figure(figsize=(12, 4))
plt.subplot(1, 4, 1)
plt.plot(random_array, marker='o', color='blue')
plt.title('Line Chart')
plt.xlabel('Index')
plt.ylabel('Value')

# Scatter Plot
plt.subplot(1, 4, 2)
plt.scatter(range(len(random_array)), random_array, color='green', marker='^')
plt.title('Scatter Plot')
plt.xlabel('Index')
plt.ylabel('Value')

# Histogram
plt.subplot(1, 4, 3)
plt.hist(random_array, bins=10, color='orange', edgecolor='black')
plt.title('Histogram')
plt.xlabel('Value')
plt.ylabel('Frequency')

# Box Plot
plt.subplot(1, 4, 4)
plt.boxplot(random_array, vert=False, widths=0.7, patch_artist=True, boxprops=dict(facecolor='pink'))
plt.title('Box Plot')
plt.xlabel('Value')

# Adjust layout for better presentation
plt.tight_layout()

# Show the plots
plt.show()
```



Q.2 B) Write a Python program to print the shape, number of rows-columns, data types, feature names and the description of the data. (Use User_Data.csv)

In [40]:

```python
import pandas as pd

# Load the dataset from the CSV file
df = pd.read_csv('user_data.csv')

# Print the shape (number of rows and columns)
print("Shape of the data:", df.shape)

# Print the number of rows and columns
print("Number of rows:", df.shape[0])
print("Number of columns:", df.shape[1])

# Print data types of each column
print("\nData types:")
print(df.dtypes)

# Print feature names
print("\nFeature names:")
print(df.columns)

# Print the description of the data
print("\nDescription of the data:")
print(df.describe())

```

```
Shape of the data: (5, 5)
Number of rows: 5
Number of columns: 5

Data types:
Name      object
Age        int64
Gender    object
Height     int64
Weight     int64
dtype: object

Feature names:
Index(['Name', 'Age', 'Gender', 'Height', 'Weight'], dtype='object')

Description of the data:
             Age      Height     Weight
count   5.000000    5.000000   5.000000
mean   29.600000  174.000000  71.000000
std     4.159327    9.192388  14.317821
min    24.000000  162.000000  55.000000
25%    28.000000  168.000000  60.000000
50%    29.000000  175.000000  70.000000
75%    32.000000  180.000000  80.000000
max    35.000000  185.000000  90.000000
```

SLIP 5 same as SLIP 4

SLIP 6 same as SLIP 2

SLIP 7

Write a Python program to perform the following tasks :

a. Apply OneHot coding on Country column.

b. Apply Label encoding on purchased column

(Data.csv have two categorical column the country column, and the purchased column)

In [41]:

```python
import pandas as pd
from sklearn.preprocessing import LabelEncoder, OneHotEncoder

# Load the dataset from the CSV file
df = pd.read_csv('data2.csv')

# Display the original dataset
print("Original Dataset:")
print(df)

# a. Apply OneHot encoding on the 'Country' column
df_onehot = pd.get_dummies(df, columns=['Country'], prefix='Country')

# Display the dataset after OneHot encoding
print("\nDataset after OneHot encoding:")
print(df_onehot)

# b. Apply Label encoding on the 'Purchased' column
label_encoder = LabelEncoder()
df['Purchased'] = label_encoder.fit_transform(df['Purchased'])

# Display the dataset after Label encoding
print("\nDataset after Label encoding:")
print(df)

```

```
Original Dataset:
   Country  Age  Salary Purchased
0   France   25   50000        No
1    Spain   30   60000       Yes
2  Germany   35   70000        No
3    Spain   28   62000        No
4  Germany   40   80000       Yes
5   France   32   75000       Yes
6    Spain   45   90000        No
7   France   50   95000       Yes
8  Germany   22   55000        No
9   France   48   98000       Yes

Dataset after OneHot encoding:
   Age  Salary Purchased  Country_France  Country_Germany  Country_Spain
0   25   50000        No            True            False          False
1   30   60000       Yes           False            False           True
2   35   70000        No           False             True          False
3   28   62000        No           False            False           True
4   40   80000       Yes           False             True          False
5   32   75000       Yes            True            False          False
6   45   90000        No           False            False           True
7   50   95000       Yes            True            False          False
8   22   55000        No           False             True          False
9   48   98000       Yes            True            False          False

Dataset after Label encoding:
   Country  Age  Salary  Purchased
0   France   25   50000          0
1    Spain   30   60000          1
2  Germany   35   70000          0
3    Spain   28   62000          0
4  Germany   40   80000          1
5   France   32   75000          1
6    Spain   45   90000          0
7   France   50   95000          1
8  Germany   22   55000          0
9   France   48   98000          1
```

SLIP 8

Q.2) Write a program in python to perform following task Standardizing Data (transform them into a standard Gaussian distribution with a mean of 0 and a standard deviation of 1) (Use winequality-red.csv)

In [42]:

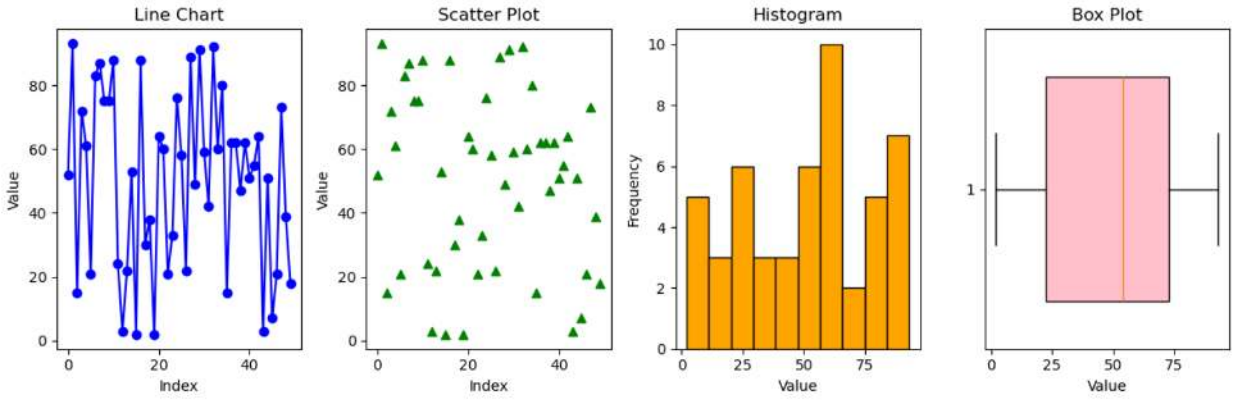
```python
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Load the dataset from the CSV file
df = pd.read_csv('winequality-red.csv')

# Display the original dataset
print("Original Dataset:")
print(df.head())

# Extract the features (excluding the target variable)
features = df.drop('quality', axis=1)

# Initialize the StandardScaler
scaler = StandardScaler()

# Fit and transform the features using StandardScaler
features_standardized = scaler.fit_transform(features)

# Create a new DataFrame with standardized features
df_standardized = pd.DataFrame(features_standardized, columns=features.columns)

# Add the 'quality' column back to the DataFrame
df_standardized['quality'] = df['quality']

# Display the dataset after standardization
print("\nDataset after Standardization:")
print(df_standardized.head())

```

```
Original Dataset:
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
3                 17.0                  60.0   0.9980  3.16       0.58
4                 11.0                  34.0   0.9978  3.51       0.56

   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5

Dataset after Standardization:
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0      -0.498662          0.451753    -0.563489       -0.329398  -0.103362
1      -0.135999          1.630239    -0.563489        0.206831   2.170608
2      -0.135999          0.844582    -0.346763       -0.022981   1.550434
3       2.946642         -2.298048     2.470683       -0.329398  -0.206725
4      -0.498662          0.451753    -0.563489       -0.329398  -0.103362

   free sulfur dioxide  total sulfur dioxide   density        pH  sulphates  \
0            -0.896665             -0.626576  0.731200  1.276466  -0.305196
1             2.406839              0.761143 -0.284356 -1.276466   0.957683
2             0.047193              0.214466 -0.081244 -0.782350   0.641963
3             0.519122              0.466778  0.934311 -1.605877  -0.094716
4            -0.896665             -0.626576  0.731200  1.276466  -0.305196

   alcohol  quality
0 -0.88083        5
1  0.29361        5
2  0.29361        5
3  0.29361        6
4 -0.88083        5
```

Q.2 A) Generate a random array of 50 integers and display them using a line chart, scatter plot. Apply appropriate color, labels and styling options.

In [38]:

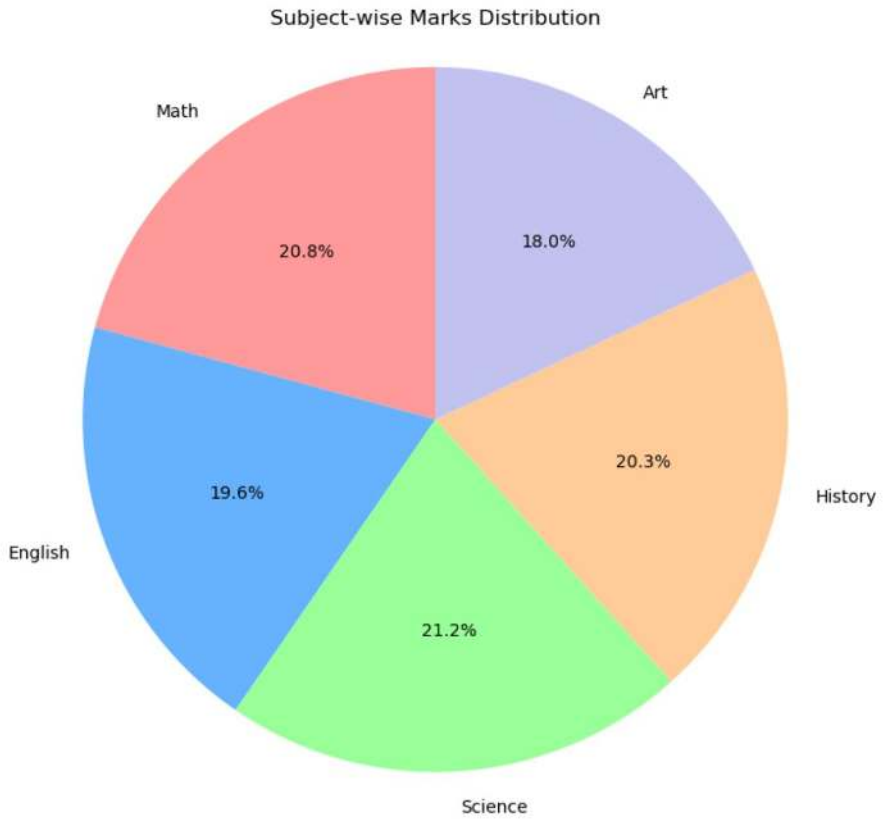
```python
import numpy as np
import matplotlib.pyplot as plt

# Set a seed for reproducibility
np.random.seed(42)

# Generate a random array of 50 integers
random_array = np.random.randint(1, 100, 50)

# Line Chart
plt.figure(figsize=(12, 4))
plt.subplot(1, 4, 1)
plt.plot(random_array, marker='o', color='blue')
plt.title('Line Chart')
plt.xlabel('Index')
plt.ylabel('Value')

# Scatter Plot
plt.subplot(1, 4, 2)
plt.scatter(range(len(random_array)), random_array, color='green', marker='^')
plt.title('Scatter Plot')
plt.xlabel('Index')
plt.ylabel('Value')

# Histogram
plt.subplot(1, 4, 3)
plt.hist(random_array, bins=10, color='orange', edgecolor='black')
plt.title('Histogram')
plt.xlabel('Value')
plt.ylabel('Frequency')

# Box Plot
plt.subplot(1, 4, 4)
plt.boxplot(random_array, vert=False, widths=0.7, patch_artist=True, boxprops=dict(facecolor='pink'))
plt.title('Box Plot')
plt.xlabel('Value')

# Adjust layout for better presentation
plt.tight_layout()

# Show the plots
plt.show()
```



Q.2 B) Create two lists, one representing subject names and the other representing marks obtained in those subjects. Display the data in a pie chart.

In [1]:

```
import matplotlib.pyplot as plt

# Sample data - replace with your own data
subjects = ['Math', 'English', 'Science', 'History', 'Art']
marks = [90, 85, 92, 88, 78]

# Create a pie chart
plt.figure(figsize=(8, 8))
plt.pie(marks, labels=subjects, autopct='%1.1f%%', startangle=90, colors=['#FF9999', '#66B2FF', '#99FF99', '#FFCC99', '#c2c2
plt.title('Subject-wise Marks Distribution')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

# Show the plot
plt.show()

```



Q.2 C) Write a program in python to perform following task (Use winequality-red.csv) Import Dataset and do the followings:

a) Describing the dataset

b) Shape of the dataset

c) Display first 3 rows from dataset

In [2]:

```
import pandas as pd

# a) Describing the dataset
# b) Shape of the dataset
# Load the dataset from the CSV file
df = pd.read_csv('winequality-red.csv')

# Display basic statistical details of the data
print("a) Describing the dataset:")
print(df.describe())

# Display the shape of the dataset
print("\nb) Shape of the dataset:")
print(df.shape)

# c) Display first 3 rows from the dataset
print("\nc) Display first 3 rows from the dataset:")
print(df.head(3))

```

```
a) Describing the dataset:
       fixed acidity  volatile acidity  citric acid  residual sugar  \
count      10.000000            10.000    10.000000       10.000000
mean        7.950000             0.631     0.104000        2.330000
std         1.162612             0.161     0.194548        1.376025
min         7.300000             0.280     0.000000        1.200000
25%         7.400000             0.585     0.000000        1.825000
50%         7.650000             0.655     0.010000        1.900000
75%         7.800000             0.700     0.055000        2.225000
max        11.200000             0.880     0.560000        6.100000

       chlorides  free sulfur dioxide  total sulfur dioxide    density  \
count  10.000000            10.000000             10.000000  10.000000
mean    0.077000            14.800000             48.900000   0.997080
std     0.010198             4.467164             25.066356   0.001038
min     0.065000             9.000000             18.000000   0.994600
25%     0.071500            11.500000             34.000000   0.996800
50%     0.075000            15.000000             47.000000   0.997400
75%     0.076000            16.500000             59.750000   0.997800
max     0.098000            25.000000            102.000000   0.998000

              pH  sulphates    alcohol    quality
count  10.000000  10.000000  10.000000  10.000000
mean    3.355000   0.589000   9.700000   5.500000
std     0.127997   0.100161   0.359011   0.849837
min     3.160000   0.460000   9.400000   5.000000
25%     3.270000   0.560000   9.400000   5.000000
50%     3.355000   0.565000   9.650000   5.000000
75%     3.480000   0.632500   9.800000   5.750000
max     3.510000   0.800000  10.500000   7.000000

b) Shape of the dataset:
(10, 12)

c) Display first 3 rows from the dataset:
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65

   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
```

SLIP 10

Q.2 A) Write a python program to Display column-wise mean, and median for SOCR- HeightWeight dataset.

In [3]:

```python
import pandas as pd

# Assuming 'SOCR-HeightWeight.csv' is the dataset file
# Make sure the file is in the same directory as your Python script or provide the correct path
df = pd.read_csv('height_weight.csv')

# Display column-wise mean
print("Column-wise Mean:")
mean_values = df.mean()
print(mean_values)

# Display column-wise median
print("\nColumn-wise Median:")
median_values = df.median()
print(median_values)

```

```
Column-wise Mean:
Height    172.0
Weight     66.9
dtype: float64

Column-wise Median:
Height    171.0
Weight     66.5
dtype: float64
```

Q.2 B) Write a python program to compute sum of Manhattan distance between all pairs of points.

In [4]:

```python
import itertools

def manhattan_distance(point1, point2):
    return abs(point1[0] - point2[0]) + abs(point1[1] - point2[1])

# Sample dataset (replace with your own dataset)
points = [(1, 2), (3, 4), (5, 6), (7, 8)]

# Compute the sum of Manhattan distance between all pairs of points
total_distance = sum(manhattan_distance(point1, point2) for point1, point2 in itertools.combinations(points, 2))

# Display the result
print("Sum of Manhattan distance between all pairs of points:", total_distance)

```

```
Sum of Manhattan distance between all pairs of points: 40
```

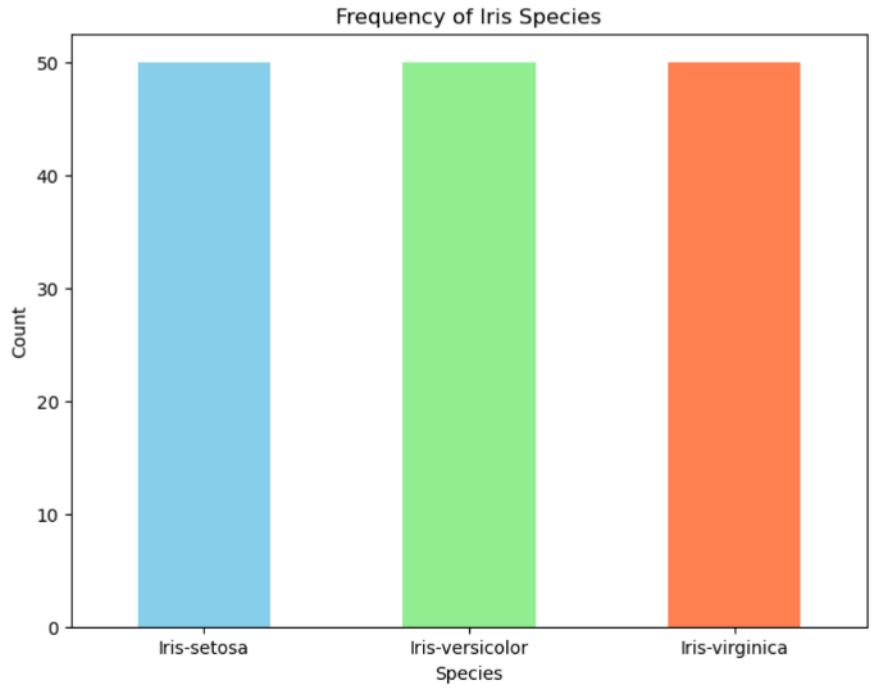
SLIP 11 same as SLIP 1

SLIP 18 same as SLIP 3

Q.2 A) Import dataset “iris.csv”. Write a Python program to create a Bar plot to get the frequency of the three species of the Iris data.

In [2]:

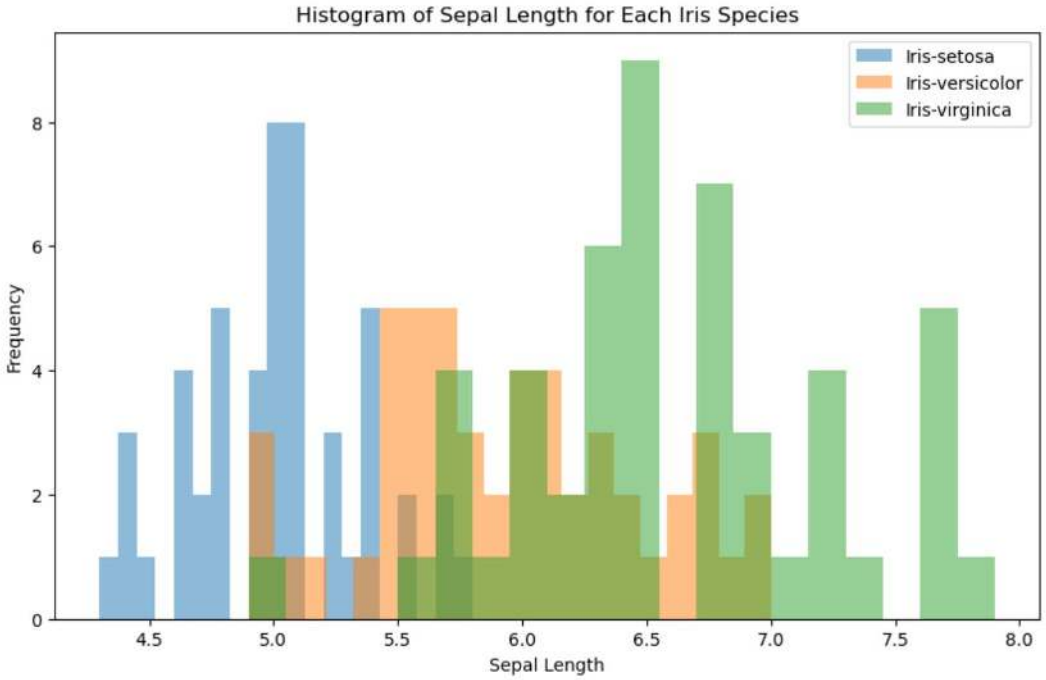
```python
import pandas as pd
import matplotlib.pyplot as plt

# Load the Iris dataset
iris_data = pd.read_csv('iris.csv')

# Count the frequency of each species
species_counts = iris_data['Species'].value_counts()

# Create a bar plot
plt.figure(figsize=(8, 6))
species_counts.plot(kind='bar', color=['skyblue', 'lightgreen', 'coral'])
plt.title('Frequency of Iris Species')
plt.xlabel('Species')
plt.ylabel('Count')
plt.xticks(rotation=0)  # Rotate x-axis labels for better readability

# Show the plot
plt.show()

```



Q.2 B) Write a Python program to create a histogram of the three species of the Iris data.

In [6]:

```python
import pandas as pd
import matplotlib.pyplot as plt

# Load the Iris dataset
iris_data = pd.read_csv('iris.csv')

# Create a histogram for each species
plt.figure(figsize=(10, 6))

# Iterate through each species and create a histogram
for species in iris_data['Species'].unique():
    subset = iris_data[iris_data['Species'] == species]
    plt.hist(subset['SepalLengthCm'], bins=20, alpha=0.5, label=species)

# Add labels and title
plt.title('Histogram of Sepal Length for Each Iris Species')
plt.xlabel('Sepal Length')
plt.ylabel('Frequency')
plt.legend()

# Show the plot
plt.show()

```



SLIP 24 same as SLIP 21