### **Slip 1**

Q2 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).

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

# Load iris data
iris = pd.read_csv('iris.csv')

# Get the frequency of species
species_count = iris['species'].value_counts()

# Create a pie plot
plt.pie(species_count, labels=species_count.index, autopct='%1.1f%%', startangle=140)
plt.title('Species Frequency in Iris Dataset')
plt.show()
```

Q2 B) Write a Python program to view basic statistical details of the data (Use wineequality-red.csv).

```python
import pandas as pd
```

```python
# Load the dataset
wine_data = pd.read_csv('winequality-red.csv')

# View basic statistical details
print(wine_data.describe())
```

### **Slip 2**

Q2 A) Write a Python program for handling missing values. Replace missing values of salary and age columns with the mean of that column (Use Data.csv).

```python
import pandas as pd

# Load dataset
data = pd.read_csv('Data.csv')

# Fill missing values with mean
data['salary'].fillna(data['salary'].mean(), inplace=True)
data['age'].fillna(data['age'].mean(), inplace=True)

print(data)
```

Q2 B) Write a Python program to generate a line plot of name vs salary.

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

# Load dataset
data = pd.read_csv('Data.csv')

# Generate line plot
plt.plot(data['name'], data['salary'], marker='o')
plt.xlabel('Name')
plt.ylabel('Salary')
plt.title('Name vs Salary')
plt.xticks(rotation=45)
plt.show()
```

Q2 C) Download the heights and weights dataset and load the dataset from a given csv file into a dataframe. Print the first, last 10 rows, random 20 rows, and display the shape of the dataset.

```python
import pandas as pd
```

```
# Load dataset

data = pd.read_csv('heights_weights.csv')


# Print first 10 rows

print(data.head(10))


# Print last 10 rows

print(data.tail(10))


# Print random 20 rows

print(data.sample(20))


# Display the shape of the dataset

print(data.shape)
```

### **Slip 3**

Q2 A) Write a Python program to create box plots to see how each feature (Sepal Length, Sepal Width, Petal Length, Petal Width) is distributed across the three species (Use iris.csv dataset).

```python
import pandas as pd

import seaborn as sns
import matplotlib.pyplot as plt

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

# Create box plots for each feature
plt.figure(figsize=(12, 8))
sns.boxplot(x='species', y='sepal_length', data=iris)
plt.title('Sepal Length Distribution by Species')
plt.show()

sns.boxplot(x='species', y='sepal_width', data=iris)
plt.title('Sepal Width Distribution by Species')
plt.show()

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

sns.boxplot(x='species', y='petal_width', data=iris)
plt.title('Petal Width Distribution by Species')
plt.show()
```

Q2 B) Write a Python program to view basic statistical details of the data (Use Heights and Weights Dataset).

```python
import pandas as pd

# Load dataset
data = pd.read_csv('heights_weights.csv')

# View basic statistical details
print(data.describe())
```

### **Slip 4**

Q2 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.

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

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

# Line plot
plt.plot(data, label='Line Chart', color='b', linestyle='--', marker='o')
plt.title('Line Chart')
plt.show()

# Scatter plot
plt.scatter(range(len(data)), data, color='g')
plt.title('Scatter Plot')
plt.show()

# Histogram
plt.hist(data, bins=10, color='r', alpha=0.7)
plt.title('Histogram')
plt.show()

# Box plot
plt.boxplot(data)
plt.title('Box Plot')
plt.show()
```

Q2 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).

```python
import pandas as pd

# Load dataset
```

```
data = pd.read_csv('User_Data.csv')

# Print shape
print(f"Shape: {data.shape}")

# Print number of rows and columns
print(f"Number of rows: {data.shape[0]}")
print(f"Number of columns: {data.shape[1]}")

# Print data types
print("Data types:")
print(data.dtypes)

# Print feature names
print("Feature names:", list(data.columns))

# Print description
print(data.describe())
```

### **Slip 5**

Q2 A) Write a Python program for handling missing values. Replace missing values of salary and age columns with the mean of that column (Use Data.csv).

Same as **Slip 2 Q2 A**.

Q2 B) Write a Python program to generate a line plot of name vs salary.

Same as **Slip 2 Q2 B**.

Q2 C) Download the heights and weights dataset and load the dataset from a given csv file into a dataframe. Print the first, last 10 rows, random 20 rows, and display the shape of the dataset.

Same as **Slip 2 Q2 C**.

Would you like to continue with the remaining Python questions from the other slips? Let me know!

Continuing with the solutions for Python **Q2** questions from the next slips:

### **Slip 6**

Q2 A) Write a Python program to perform the following tasks:

- **Apply OneHot coding on Country column.**

- **Apply Label encoding on purchased column. (Use `Data.csv` with two categorical columns: country and purchased)**

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

# Load dataset
data = pd.read_csv('Data.csv')

# OneHot encoding for country column
one_hot = OneHotEncoder()
encoded_country = one_hot.fit_transform(data[['country']]).toarray()
encoded_country_df = pd.DataFrame(encoded_country,
columns=one_hot.get_feature_names_out(['country']))
data = pd.concat([data, encoded_country_df], axis=1)

# Label encoding for purchased column
label_encoder = LabelEncoder()
data['purchased_encoded'] = label_encoder.fit_transform(data['purchased'])

print(data)
```

### **Slip 7**

Q2 A) Write a Python program to standardize data (transform into a standard Gaussian distribution with mean of 0 and standard deviation of 1) using `winequality-red.csv`.

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

# Load dataset
data = pd.read_csv('winequality-red.csv')

# Standardize data
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data)

# Convert back to dataframe
scaled_data_df = pd.DataFrame(scaled_data, columns=data.columns)
print(scaled_data_df)
```

### **Slip 8**

Q2 A) Generate a random array of 50 integers and display them using a line chart, scatter plot. Apply appropriate color, labels, and styling options.

Same as **Slip 4 Q2 A**.

Q2 B) Create two lists, one representing subject names and the other representing marks obtained in those subjects. Display the data in a pie chart.

```python
import matplotlib.pyplot as plt

# Create lists for subjects and marks
subjects = ['Math', 'Physics', 'Chemistry', 'Biology', 'Computer Science']
marks = [85, 90, 75, 80, 95]

# Create a pie chart
plt.pie(marks, labels=subjects, autopct='%1.1f%%', startangle=140)
plt.title('Marks Distribution in Subjects')
plt.show()
```

### **Slip 9**

Q2 A) Write a Python program to display column-wise mean and median for the SOCR-HeightWeight dataset.

```python
```

```
import pandas as pd

# Load dataset
data = pd.read_csv('SOCR-HeightWeight.csv')

# Calculate mean and median for each column
mean_values = data.mean()
median_values = data.median()

print(f"Mean Values:\n{mean_values}")
print(f"\nMedian Values:\n{median_values}")
```

Q2 B) Write a Python program to compute the sum of Manhattan distances between all pairs of points.

```python
import numpy as np
from scipy.spatial import distance

# Generate random points
points = np.random.rand(10, 2)  # 10 points in 2D space

# Calculate Manhattan distance between all pairs
manhattan_distances = distance.cdist(points, points, 'cityblock')

# Sum of all Manhattan distances
```

```
total_distance = np.sum(manhattan_distances)
print(f"Sum of Manhattan distances between all pairs: {total_distance}")
```

### **Slip 10**

Q2 A) Write a Python program to create a graph to find the relationship between petal length and petal width using the `iris.csv` dataset.

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

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

# Scatter plot for petal length vs petal width
plt.scatter(iris['petal_length'], iris['petal_width'], color='g')
plt.xlabel('Petal Length')
plt.ylabel('Petal Width')
plt.title('Petal Length vs Petal Width')
plt.show()
```

Q2 B) Write a Python program to find the maximum and minimum value of a given flattened array.

```python
import numpy as np

# Create a random array
array = np.random.rand(5, 5)

# Flatten the array
flat_array = array.flatten()

# Find max and min
max_value = np.max(flat_array)
min_value = np.min(flat_array)

print(f"Max value: {max_value}")
print(f"Min value: {min_value}")
```

### **Slip 11**

Q2 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.

Same as **Slip 4 Q2 A**.

Q2 B) Write a Python program to create a dataframe containing columns: name, salary, department, and add 10 rows with some missing and duplicate values. Drop all null and empty values and print the modified dataframe.

```python
import pandas as pd


# Create a dataframe with missing and duplicate values
data = {
    'name': ['John', 'Sara', 'Tom', 'Emily', 'Anna', 'Tom', 'Mike', None, 'Anna', 'Jack'],
    'salary': [50000, 60000, 45000, 70000, 50000, 45000, None, 54000, 50000, None],
    'department': ['IT', 'HR', 'Sales', 'HR', 'Sales', 'Sales', 'IT', 'IT', 'Sales', None]
}
df = pd.DataFrame(data)


# Drop missing values and duplicates
df_cleaned = df.dropna().drop_duplicates()


print("Cleaned DataFrame:")
print(df_cleaned)
```

### **Slip 12**

Q2 A) Write a Python program to create a graph to find the relationship between petal length and petal width (Use iris.csv dataset).

Same as **Slip 10 Q2 A**.

Q2 B) Write a Python program to find the maximum and minimum value of a given flattened array.

Same as **Slip 10 Q2 B**.

### **Slip 13**

Q2 A) Write a Python NumPy program to compute the weighted average along the specified axis of a given flattened array.

```python
import numpy as np

# Create a random array
array = np.random.rand(5, 5)

# Flatten the array
flat_array = array.flatten()

# Create weights
```

weights = np.random.rand(len(flat_array))

# Compute the weighted average
weighted_avg = np.average(flat_array, weights=weights)
print(f"Weighted average: {weighted_avg}")

Q2 B) Write a Python program to view basic statistical details of the data (Use advertising.csv).

```python
import pandas as pd

# Load dataset
data = pd.read_csv('advertising.csv')

# View basic statistical details
print(data.describe())
```

### **Slip 14**

Q2 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.

Same as **Slip 4 Q2 A**.

Q2 B) Create two lists, one representing subject names and the other representing marks obtained in those subjects. Display the data in a pie chart.

Same as **Slip 8 Q2 B**.

### **Slip 15**

Q2 A) Write a Python program to create two lists, one representing subject names and the other representing marks obtained in those subjects. Display the data in a pie chart and bar chart.

```python
import matplotlib.pyplot as plt

# Create lists for subjects and marks
subjects = ['Math', 'Physics', 'Chemistry', 'Biology', 'Computer Science']
marks = [85, 90, 75, 80, 95]

# Create a pie chart
plt.pie(marks, labels=subjects, autopct='%1.1f%%', startangle=140)
plt.title('Marks Distribution in Subjects')
plt.show()
```

```
# Create a bar chart
plt.bar(subjects, marks, color='blue')
plt.xlabel('Subjects')
plt.ylabel('Marks')
plt.title('Marks in Subjects')
plt.show()
```

Q2 B) Write a Python program to create a dataframe for students’ information such as name, graduation percentage, and age. Display the average age of students and the average graduation percentage.

```python
import pandas as pd

# Create a dataframe for students' information
data = {
    'name': ['John', 'Sara', 'Tom', 'Emily', 'Anna'],
    'graduation_percentage': [85, 90, 75, 80, 95],
    'age': [22, 23, 21, 24, 22]
}
df = pd.DataFrame(data)

# Calculate and display averages
average_age = df['age'].mean()
average_percentage = df['graduation_percentage'].mean()
```

print(f"Average age: {average_age}")

print(f"Average graduation percentage: {average_percentage}")

### **Slip 16**

Q2 A) Write a Python program to draw scatter plots to compare two features of the iris dataset.

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

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

# Scatter plot for Sepal Length vs Sepal Width
plt.scatter(iris['sepal_length'], iris['sepal_width'], color='b')
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.title('Sepal Length vs Sepal Width')
plt.show()
```

Q2 B) Write a Python program to create a dataframe containing columns: name, age, salary, department. Add 10 rows to the dataframe and view the dataframe.

```python
import pandas as pd

# Create a dataframe with 10 rows
data = {
    'name': ['John', 'Sara', 'Tom', 'Emily', 'Anna', 'Mike', 'Jack', 'Harry', 'Leo', 'Lucy'],
    'age': [22, 23, 25, 24, 22, 26, 21, 27, 23, 25],
    'salary': [50000, 60000, 45000, 70000, 50000, 54000, 50000, 59000, 55000, 63000],
    'department': ['IT', 'HR', 'Sales', 'HR', 'Sales', 'IT', 'Sales', 'HR', 'IT', 'Sales']
}
df = pd.DataFrame(data)

# View the dataframe
print(df)
```

### **Slip 17**

Q2 A) Write a Python program to create box plots to see how each feature (Sepal Length, Sepal Width, Petal Length, Petal Width) is distributed across the three species (Use iris.csv dataset).

Same as **Slip 3 Q2 A**.

Q2 B) Use the heights and weights dataset and load the dataset from a given csv file into a dataframe. Print the first, last 5 rows, and random 10 rows.

```python
import pandas as pd

# Load dataset
data = pd.read_csv('heights_weights.csv')

# Print first 5 rows
print("First 5 rows:")
print(data.head())

# Print last 5 rows
print("\nLast 5 rows:")
print(data.tail())

# Print random 10 rows
print("\nRandom 10 rows:")
print(data.sample(10))
```

Would you like to continue with the next set of slips, or is there any specific question that you need further clarification on? Let me know!

Continuing with the Python **Q2** answers from the next slips:

### **Slip 18**

Q2 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.

Same as **Slip 4 Q2 A**.

Q2 B) Add two outliers to the above data and display the box plot.

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

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

# Add two outliers
data = np.append(data, [200, 250])
```

# Box plot with outliers

plt.boxplot(data)

plt.title('Box Plot with Outliers')

plt.show()

~~~

Slip 19

Q2 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.

```python

import pandas as pd

import matplotlib.pyplot as plt

# Load iris dataset

iris = pd.read_csv('iris.csv')

# Get the frequency of each species

species_count = iris['species'].value_counts()

# Create a bar plot

species_count.plot(kind='bar', color='c')

plt.title('Frequency of Iris Species')

plt.xlabel('Species')
~~~

```
plt.ylabel('Count')
plt.show()
```

**Q2 B) Write a Python program to create a histogram of the three species of the Iris data.**

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

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

# Create histogram for sepal length grouped by species
iris[iris['species'] == 'setosa']['sepal_length'].hist(alpha=0.5, label='Setosa', color='r')
iris[iris['species'] == 'versicolor']['sepal_length'].hist(alpha=0.5, label='Versicolor', color='g')
iris[iris['species'] == 'virginica']['sepal_length'].hist(alpha=0.5, label='Virginica', color='b')

plt.legend()
plt.title('Sepal Length Distribution by Species')
plt.xlabel('Sepal Length')
plt.ylabel('Frequency')
plt.show()
```

~~~

Slip 20

Q2 A) Write a Python program to perform the following tasks on the dataset `winequality-red.csv`:

- **Rescale the dataset using `MinMaxScaler`.**

- **Standardize the dataset into a Gaussian distribution with mean 0 and standard deviation 1.**

- **Normalize the dataset using `Normalizer` (rescale each observation to a length of 1).**

```python
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Normalizer

# Load dataset
data = pd.read_csv('winequality-red.csv')

# Rescale data using MinMaxScaler
min_max_scaler = MinMaxScaler()
rescaled_data = min_max_scaler.fit_transform(data)

# Standardize data
standard_scaler = StandardScaler()
~~~

```
standardized_data = standard_scaler.fit_transform(data)

# Normalize data
normalizer = Normalizer()
normalized_data = normalizer.fit_transform(data)

print("Rescaled Data (MinMaxScaler):\n", rescaled_data)
print("\nStandardized Data (StandardScaler):\n", standardized_data)
print("\nNormalized Data (Normalizer):\n", normalized_data)
```

### **Slip 21**

Q2 A) Write a Python program to perform rescaling, standardization, and binarization on the dataset `winequality-red.csv`.

- **Rescaling using `MinMaxScaler`.**
- **Standardizing using a Gaussian distribution.**
- **Binarizing using `Binarizer`.**

```python
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Binarizer

# Load dataset
```

```
data = pd.read_csv('winequality-red.csv')

# Rescale data using MinMaxScaler
min_max_scaler = MinMaxScaler()
rescaled_data = min_max_scaler.fit_transform(data)

# Standardize data using StandardScaler
standard_scaler = StandardScaler()
standardized_data = standard_scaler.fit_transform(data)

# Binarize data
binarizer = Binarizer(threshold=0.5)
binarized_data = binarizer.fit_transform(data)

print("Rescaled Data:\n", rescaled_data)
print("\nStandardized Data:\n", standardized_data)
print("\nBinarized Data:\n", binarized_data)
```

### **Slip 22**

Q2 A) Write a Python program to create a Bar plot to get the frequency of the three species of the Iris data (Use iris.csv).

Same as **Slip 19 Q2 A**.

Q2 B) Write a Python program to create a histogram of the three species of the Iris data.

Same as **Slip 19 Q2 B**.

### **Slip 23**

Q2 A) Write a Python program to create a dataframe with the following columns: name, age, percentage. Add 10 rows to the dataframe. Perform the following tasks:

- **Print the shape, number of rows and columns, data types, feature names, and description of the data.**
- **Add 5 rows with duplicate and missing values.**
- **Add a column 'remarks' with empty values and display the dataframe.**

```python
import pandas as pd

# Create a dataframe
data = {
    'name': ['John', 'Sara', 'Tom', 'Emily', 'Anna', 'Mike', 'Jack', 'Harry', 'Leo', 'Lucy'],
    'age': [22, 23, 25, 24, 22, 26, 21, 27, 23, 25],
    'percentage': [85, 90, 75, 80, 95, 60, 70, 85, 95, 88]
}
```

```
df = pd.DataFrame(data)

# Print shape, number of rows-columns, and data types
print(f"Shape: {df.shape}")
print(f"Data types:\n{df.dtypes}")
print(f"Feature names: {list(df.columns)}")
print(f"Description:\n{df.describe()}")

# Add 5 rows with duplicate and missing values
extra_data = {
    'name': ['Tom', None, 'Lucy', None, 'Sara'],
    'age': [25, None, 25, 24, None],
    'percentage': [75, 90, 88, None, None]
}
extra_df = pd.DataFrame(extra_data)
df = df.append(extra_df, ignore_index=True)

# Add a 'remarks' column with empty values
df['remarks'] = None

# Display the dataframe
print("\nDataFrame with duplicates and missing values:")
print(df)
```

### **Slip 24**

Q2 A) Write a Python program to 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.

Same as **Slip 4 Q2 A**.

Q2 B) Create two lists, one representing subject names and the other representing marks obtained in those subjects. Display the data in a pie chart.

Same as **Slip 8 Q2 B**.

### **Slip 25**

Q2 A) Write a Python program to 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.

Same as **Slip 4 Q2 A**.

Q2 B) Create two lists, one representing subject names and the other representing marks obtained in those subjects. Display the data in a bar chart.

Same as **Slip 15 Q2 B**.

### **Slip 26**

Q2 A) Create a dataset `data.csv` having two categorical columns (the country column and the purchased column).

- **Apply OneHot encoding on the `Country` column.**
- **Apply Label encoding on the `Purchased` column.**

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

# Create a sample dataset
data = {
    'country': ['India', 'USA', 'India', 'Canada', 'Canada', 'USA'],
    'purchased': ['Yes', 'No', 'Yes', 'No', 'Yes', 'No']
}
df = pd.DataFrame(data)

# OneHot encoding on country column
one_hot_encoder = OneHotEncoder()
country_encoded = one_hot_encoder.fit_transform(df[['country']]).toarray()
country_encoded_df = pd.DataFrame(country_encoded, columns=one_hot_encoder.get_feature_names_out(['country']))
df = pd.concat([df, country_encoded_df], axis=1)

# Label encoding on purchased column
label_encoder = LabelEncoder()
```

```
df['purchased_encoded'] = label_encoder.fit_transform(df['purchased'])

print(df)
```

### **Slip 27**

Q2 A) Write a Python program to read the `student.dat` file, calculate the percentage, and display the data from the file in tabular format.

```python
import pandas as pd

# Assuming the 'student.dat' file has the following format: rollno, name, OS, WT, DS, Python, Java, CN
# Load dataset
column_names = ['rollno', 'name', 'OS', 'WT', 'DS', 'Python', 'Java', 'CN']
df = pd.read_csv('student.dat', sep=',', names=column_names)

# Calculate percentage
df['total_marks'] = df[['OS', 'WT', 'DS', 'Python', 'Java', 'CN']].sum(axis=1)
df['percentage'] = df['total_marks'] / 6

# Display data in tabular format
print(df[['rollno', 'name', 'total_marks', 'percentage']])
```

~~~

Slip 28

Q2 A) Consider the following entities and their relationships:

- **Event (eno, title, date)**

- **Committee (cno, name, head, from_time, to_time, status)**

The relationship between Event and Committee is many-to-many. Write a Python script to accept the title of an event and modify the status of the committee as "working".

Since this question is about modifying relational data, you would typically use a database. However, here is a simplified Python script simulating this process with dictionaries:

```python
# Define event and committee data

events = [{'eno': 1, 'title': 'Tech Conference', 'date': '2024-10-22'}]

committees = [{'cno': 1, 'name': 'Logistics', 'head': 'Alice', 'from_time': '09:00',
'to_time': '17:00', 'status': 'not working'},

              {'cno': 2, 'name': 'Sponsorship', 'head': 'Bob', 'from_time': '10:00',
'to_time': '16:00', 'status': 'not working'}]

# Accept event title from user

event_title = input('Enter the event title: ')
~~~

```
# Modify the status of the committees associated with the event
for event in events:
    if event['title'] == event_title:
        for committee in committees:
            committee['status'] = 'working'

print("Updated Committees:")
for committee in committees:
    print(committee)
```

### **Slip 29**

Q2 A) Consider the following entities and their relationships:
- **Student (Stud_id, name, class)**
- **Competition (c_no, c_name, type)**

The relationship between Student and Competition is many-to-many, with attributes rank and year. Write a Python program to accept a competition name from the user and display the information of the student who secured 1st rank in that competition.

```python
# Define student and competition data
```

```
students = [{'Stud_id': 1, 'name': 'John', 'class': '12th'},
            {'Stud_id': 2, 'name': 'Sara', 'class': '12th'}]
competitions = [{'c_no': 1, 'c_name': 'Math Olympiad', 'type': 'Academic'},
                {'c_no': 2, 'c_name': 'Science Fair', 'type': 'Academic'}]
results = [{'Stud_id': 1, 'c_no': 1, 'rank': 1, 'year': 2024},
           {'Stud_id': 2, 'c_no': 2, 'rank': 1, 'year': 2024}]

# Accept competition name
competition_name = input('Enter competition name: ')

# Find competition and display student with 1st rank
for competition in competitions:
    if competition['c_name'] == competition_name:
        for result in results:
            if result['c_no'] == competition['c_no'] and result['rank'] == 1:
                for student in students:
                    if student['Stud_id'] == result['Stud_id']:
                        print(f"Student who secured 1st rank: {student['name']}, Class: {student['class']}")
```

### **Slip 30**

Q2 A) Write a Python program to 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.

Same as **Slip 4 Q2 A**.

Q2 B) Create two lists, one representing subject names and the other representing marks obtained in those subjects. Display the data in a bar chart.

```python
import matplotlib.pyplot as plt

# Create lists for subjects and marks
subjects = ['Math', 'Physics', 'Chemistry', 'Biology', 'Computer Science']
marks = [85, 90, 75, 80, 95]

# Create a bar chart
plt.bar(subjects, marks, color='green')
plt.xlabel('Subjects')
plt.ylabel('Marks')
plt.title('Marks in Subjects')
plt.show()
```