SLIP 1

```
<html>
<head>
<title>slip 1</title>
<style>
body{background-color: yellow;}
table {background-color: turquoise;}
h3 { font-size: 6pt; color:red;}
</style>
</head>
<body>
<center>
<h3> Project Management </h3>
<table border="3" height="200px" width="400px">
<form name=frmlogin><br>
<tr><td>Enter Project Name:</td><td><input type=text name=t1
placeholder=projectname><br><br></td></tr>
<tr><td>Assignned to:</td><td><select name="Names" id="nm" form="frmlogin">
<option value="ross">Ross Geller</option>
<option value="chuck">Chuck Bass</option>
<option value="rachel">Rachel Green </option>
<option value="dan">Dan James</option>
</select></td></tr>
<tr><td>Start Date:</td><td><input type="date" ></td></tr>
<tr><td>End Date:</td><td><input type="date"> </td></tr>
<tr><td>Priority:</td><br>
<td><input type=radio name=ck value="high">High<br>
<input type=radio name=ck value="avg">Average <br>
```

```
<input type=radio name=ck value="low">Low</td></tr>

<tr><td>Description:</td><td><input type=text name="dt"

value="description"></td></tr>

<tr><td><input type=submit value ="submit"></td>

<td><input type=reset value ="clear"></td></tr>

</table>

</center>

</body>

</html>
```

Q2)

```
import pandas as pd

import matplotlib.pyplot as plt


# Load the iris dataset

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


# Check the first few rows of the dataset

print(iris_data.head())


# Get the frequency of each species

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


# Create a pie plot
```

```
plt.figure(figsize=(8, 8))
plt.pie(species_count, labels=species_count.index, autopct='%1.1f%%', startangle=140)
plt.title('Frequency of Iris Species')
plt.axis('equal')  # Equal aspect ratio ensures that pie chart is circular.
plt.show()
```

Q2 B)

```
import pandas as pd

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

# Display the first few rows of the dataset
print("First few rows of the dataset:")
print(wine_data.head())

# Display basic statistical details
print("\nStatistical summary:")
print(wine_data.describe())

# Display additional information about the dataset
print("\nDataset Information:")
print(wine_data.info())
```

```
<html>
<head>
<title>PUNE</title>
<body style="background-color:Pink;">
<h3 style="font-size:100px; color:blue;"> PUNE</h3>
<li style="color:brown; font-style: italic;">Dagadu Seth Ganpati Temple</li>
<li style="color:red; font-style: oblique;">Saras Baug Temple</li>
<li style="color:purple; font-style: italic;">Phoenix Market City Pune</li>
<img src="https://mittalbuilders.com/wp-content/uploads/2020/12/Reasons-to-
settle-down-in-Pune.png" width="200" height="200" align="bottom">
</body>
</head>
</head>
```

Q2)

```
import pandas as pd

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

# Display the first few rows and check for missing values
print("First few rows of the dataset:")
print(data.head())
print("\nMissing values in each column:")
print(data.isnull().sum())
```

```
# Replace missing values in 'salary' and 'age' with their respective means
data['salary'].fillna(data['salary'].mean(), inplace=True)
data['age'].fillna(data['age'].mean(), inplace=True)

# Check again for missing values after replacement
print("\nMissing values after replacement:")
print(data.isnull().sum())

# Display the modified dataset
print("\nModified dataset:")
print(data.head())

# Optionally, save the modified dataset to a new CSV file
# data.to_csv('Data_modified.csv', index=False)
```

Q2 B)

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

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

# Check the first few rows of the dataset
print("First few rows of the dataset:")
print(iris_data.head())
```

```
# Get the frequency of each species
species_count = iris_data['species'].value_counts()

# Create a bar plot
plt.figure(figsize=(8, 6))
species_count.plot(kind='bar', color='skyblue')
plt.title('Frequency of Iris Species')
plt.xlabel('Species')
plt.ylabel('Frequency')
plt.xticks(rotation=0)  # Rotate x-axis labels if needed
plt.grid(axis='y', linestyle='--', alpha=0.7)  # Optional: Add grid lines
plt.show()
```

Q2 C)

```
import pandas as pd

# Load the dataset from a CSV file
file_path = 'heights_weights.csv'  # Change this to the path of your CSV file
df = pd.read_csv(file_path)

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

# Print the last 10 rows
print("\nLast 10 rows:")
```

```
print(df.tail(10))


# Print 20 random rows

print("\nRandom 20 rows:")

print(df.sample(20))


# Display the shape of the dataset

print("\nShape of the dataset:")

print(df.shape)
```

```
<html>

<head>

<title>Accenture</title></head>

<body style="background-color:#4CBB17">

<h3 style="font-size:60px; color:red; font-style:Comic Sans MS "> Accenture</h3>

<p style=" color:blue; font-size:25px">Accenture plc is an Irish-American professional services company based in Dublin,

specializing in information technology (IT) services and consulting.<br>

As of 2022,Accenture is considered

the largest consulting firm in the world by number of employees.<br>

Accenture is a $61.6-billion-in-annual-revenue technology and consulting company incorporated in Dublin, Ireland.

Led by Chair & CEO Julie Sweet, who prior to her promotion in 2019 served as CEO of Accentures business in North America,

the Fortune Global 500 information technology services company has supplemented its growth

through high-profile acquisitions like that of ad agency Droga5. With 721,000 people worldwide</p>

</html>
```

Q2)

```
import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt


# Load the Iris dataset from a CSV file

file_path = 'iris.csv'  # Update this path if necessary

iris_data = pd.read_csv(file_path)
```

```
# Display the first few rows of the dataset
print(iris_data.head())

# Set the style of seaborn
sns.set(style='whitegrid')

# Create box plots for each feature across species
features = ['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']
species = iris_data['Species'].unique()

# Plot box plots for each feature
plt.figure(figsize=(16, 12))

for i, feature in enumerate(features, start=1):
    plt.subplot(2, 2, i)  # Create a 2x2 grid for subplots
    sns.boxplot(x='Species', y=feature, data=iris_data)
    plt.title(f'Box Plot of {feature} by Species')
    plt.xlabel('Species')
    plt.ylabel(feature)

plt.tight_layout()  # Adjust the layout
plt.show()
```

Q2 B)

```
import pandas as pd

# Load the dataset
file_path = 'heights_weights.csv'  # Update this to your actual file path
data = pd.read_csv(file_path)

# Display the first few rows of the dataset
print("First few rows of the dataset:")
print(data.head())

# Calculate basic statistics
statistics = data.describe()

# Display the statistical details
print("\nBasic Statistical Details:")
print(statistics)

# Optional: Additional statistics
mean_height = data['Height'].mean()
mean_weight = data['Weight'].mean()

print(f"\nMean Height: {mean_height}")
print(f"Mean Weight: {mean_weight}")
```

<html> SLIP 4
<body>
<table border = 1>
<caption>list of book </caption> <tr>
<td rowspan=2>item no </td> <td rowspan=2>Item name </td> <td colspan=2 > price
<tr>
<td >Rs </td>
<td >paise</td>
</td>
</tr>
</tr>
<tr>

<td>1</td>
<td>programming in python </td> <td>500</td><td>50</td>

</tr> <tr>

<td>2</td>
<td>programming in java </td> <td>345</td><td>00</td>

</tr> </body> </html>

Q2)

import numpy as np

import matplotlib.pyplot as plt

Generate a random array of 50 integers between 1 and 100

data = np.random.randint(1, 101, size=50)

Set up the figure and axes

fig, axs = plt.subplots(2, 2, figsize=(12, 10))

fig.suptitle('Random Integer Array Visualization', fontsize=16)

```
# Line chart
axs[0, 0].plot(data, color='blue', marker='o', linestyle='-')
axs[0, 0].set_title('Line Chart')
axs[0, 0].set_xlabel('Index')
axs[0, 0].set_ylabel('Value')
axs[0, 0].grid(True)

# Scatter plot
axs[0, 1].scatter(range(len(data)), data, color='green', alpha=0.7)
axs[0, 1].set_title('Scatter Plot')
axs[0, 1].set_xlabel('Index')
axs[0, 1].set_ylabel('Value')
axs[0, 1].grid(True)

# Histogram
axs[1, 0].hist(data, bins=10, color='purple', edgecolor='black')
axs[1, 0].set_title('Histogram')
axs[1, 0].set_xlabel('Value')
axs[1, 0].set_ylabel('Frequency')

# Box plot
axs[1, 1].boxplot(data, patch_artist=True, boxprops=dict(facecolor='cyan'))
axs[1, 1].set_title('Box Plot')
axs[1, 1].set_ylabel('Value')

# Adjust layout
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
```

```
plt.show()
```

Q2 B)

```
import pandas as pd

# Load the dataset
file_path = 'User_Data.csv'  # Update this to your actual file path
data = pd.read_csv(file_path)

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

# Print the number of rows and columns
num_rows, num_columns = data.shape
print("Number of Rows:", num_rows)
print("Number of Columns:", num_columns)

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

# Print feature names
print("\nFeature Names:")
print(data.columns.tolist())
```

```
# Print the description of the data

print("\nDescription of the Data:")

print(data.describe(include='all'))  # Include='all' to describe categorical columns as well
```

```
<html lang="en"> <head>

<title>NR Class</title>
<link rel="stylesheet" href="bootstrap.min.css">

</head> <body>

<div class="jumbotron">
<center><h1>My First Bootstrap Page</h1></center> <center><p>This is responsive</p>
</center>

</div>
<div class="row">

<div class="col-3 offset-1 bg-light"> <h4>Personal Information</h4> <p>name</p>

</div>
<div class="col-3 offset-1 bg-light">

<h4>Educational Information</h4>

<p>name</p> </div>

<div class="col-3 offset-1 bg-light"> <h4>Job Profile</h4> <p>name</p>

</div> </div>

</body> </html>
```

Q2)

```
import numpy as np

import matplotlib.pyplot as plt


# Generate a random array of 50 integers between 1 and 100

data = np.random.randint(1, 101, size=50)


# Set up the figure and axes for a 2x2 grid of plots

fig, axs = plt.subplots(2, 2, figsize=(12, 10))
```

```
fig.suptitle('Random Integer Array Visualization', fontsize=16)

# Line chart
axs[0, 0].plot(data, color='blue', marker='o', linestyle='-')
axs[0, 0].set_title('Line Chart', fontsize=14)
axs[0, 0].set_xlabel('Index', fontsize=12)
axs[0, 0].set_ylabel('Value', fontsize=12)
axs[0, 0].grid(True)

# Scatter plot
axs[0, 1].scatter(range(len(data)), data, color='green', alpha=0.7)
axs[0, 1].set_title('Scatter Plot', fontsize=14)
axs[0, 1].set_xlabel('Index', fontsize=12)
axs[0, 1].set_ylabel('Value', fontsize=12)
axs[0, 1].grid(True)

# Histogram
axs[1, 0].hist(data, bins=10, color='purple', edgecolor='black')
axs[1, 0].set_title('Histogram', fontsize=14)
axs[1, 0].set_xlabel('Value', fontsize=12)
axs[1, 0].set_ylabel('Frequency', fontsize=12)

# Box plot
axs[1, 1].boxplot(data, patch_artist=True, boxprops=dict(facecolor='cyan'))
axs[1, 1].set_title('Box Plot', fontsize=14)
axs[1, 1].set_ylabel('Value', fontsize=12)
```

```
# Adjust layout for better spacing
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.show()
```

Q2 B)

```
import pandas as pd

# Load the dataset
file_path = 'User_Data.csv'  # Update this to your actual file path
data = pd.read_csv(file_path)

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

# Print the number of rows and columns
num_rows, num_columns = data.shape
print("Number of Rows:", num_rows)
print("Number of Columns:", num_columns)

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

# Print feature names
```

```
print("\nFeature Names:")

print(data.columns.tolist())


# Print the description of the data

print("\nDescription of the Data:")

print(data.describe(include='all'))  # Include='all' to describe categorical columns as well
```

```
<html>
<head>
<link rel="stylesheet" href="bootstrap.min.css"> <title>NRC</title>
</head>

<body>
<div class="row ">

<div class="col-12 bg-primary"> <h3>header</h3>

</div> </div>

<div class="row">
<div class="col-4 bg-info">

<h4>Menu</h4> </div>

<div class="col-8 bg-warning "> <p>this is collge information</p>

</div> </div>

<div class="row">
<div class="col-12 bg-danger"> <h3>Footer</h3>
</div>

</div> </body> </html>
```

Q2)

```
import pandas as pd


# Load the dataset

file_path = 'Data.csv'  # Update this to your actual file path

data = pd.read_csv(file_path)


# Display the original data

print("Original Data:")

print(data)
```

```
# Check for missing values
print("\nMissing Values in Each Column:")
print(data.isnull().sum())

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

if 'age' in data.columns:
    mean_age = data['age'].mean()
    data['age'].fillna(mean_age, inplace=True)

# Display the data after handling missing values
print("\nData After Handling Missing Values:")
print(data)

# Optional: Save the cleaned data to a new CSV file
data.to_csv('Cleaned_Data.csv', index=False)
```

Q2 B)

```
import pandas as pd
import seaborn as sns
```

```
import matplotlib.pyplot as plt

# Load the Iris dataset
# If you have the dataset as a CSV file, you can use:
# data = pd.read_csv('iris.csv')

# Using seaborn to load the iris dataset directly
data = sns.load_dataset('iris')

# Display the first few rows of the dataset (optional)
print(data.head())

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

# Create a bar plot
plt.figure(figsize=(8, 6))
sns.barplot(x=species_counts.index, y=species_counts.values, palette='viridis')

# Adding titles and labels
plt.title('Frequency of Iris Species', fontsize=16)
plt.xlabel('Species', fontsize=14)
plt.ylabel('Frequency', fontsize=14)

# Show the plot
plt.xticks(rotation=45)
plt.tight_layout()
```

```
plt.show()
```

Q2 C)

```
import pandas as pd

# Load the dataset
file_path = 'heights_weights.csv'  # Update this to your actual file path
data = pd.read_csv(file_path)

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

# Print the first 10 rows
print("\nFirst 10 rows of the dataset:")
print(data.head(10))

# Print the last 10 rows
print("\nLast 10 rows of the dataset:")
print(data.tail(10))

# Print a random sample of 20 rows
print("\nRandom 20 rows of the dataset:")
print(data.sample(n=20))
```

```
<html> <head>

<title>Navigation Bar</title> <style>
li {

display:inline; }

</style>
</head>
<body style="font-size:40px"> <ul>

<li><a href="HOME.asp" style="color:white; background- color:grey;">HOME</a></li>

<li> <a href="JAVA.asp" style="color:blue; background- color:#D3D3D3;">Java</a></li>

<li ><a href="HTML.asp" style="color:blue; background- color:#D3D3D3;">HTML</a></li>

<li><a href="CSS.asp" style="color:blue; background- color:#D3D3D3;">CSS</a></li>
</ul>
</body>

</html>
```

Q2)

```
import pandas as pd

from sklearn.preprocessing import LabelEncoder


# Load the dataset

file_path = 'Data.csv'  # Update this to your actual file path

data = pd.read_csv(file_path)


# Display the original data

print("Original Data:")

print(data)
```

```
# a. Apply One-Hot Encoding on the Country column

data_one_hot = pd.get_dummies(data, columns=['Country'], drop_first=True)


# b. Apply Label Encoding on the Purchased column

label_encoder = LabelEncoder()

data_one_hot['Purchased'] = label_encoder.fit_transform(data_one_hot['Purchased'])


# Display the modified data

print("\nData After Encoding:")

print(data_one_hot)
```

Q2 B)

HTML FILE

```
<html>
<body>
<form action="slip_8.php" method="get">
enter first string:<input type="text" name="str1"><br> enter second string:<input type="text"
name="str2"><br> <input type="submit" value="submit">
</form>
</body>
</html>
```

PHP FILE

```
<?php
$a=$_GET["str1"];
$b=$_GET["str2"];

//$len1=strlen($a); //$len2=strlen($b);

echo "Length $len1 $len2"; $pos=strpos($a,$b); if($pos==0)
{

echo"the small string appears at the start of the large string<br>"; }

else {

echo"small string does not appear at the start of the large string<br>";

}
$pos=strpos($a,$b);
echo"the small string appears at $pos position<br>";

if(strcasecmp($a,$b)==0)
echo "Both Strings are equal<br>";

else if(strcasecmp($a,$b)>0)
echo "first string bigger<br>";

else
echo "second string is bigger<br>";

?>
```

Q2)

```
import pandas as pd

from sklearn.preprocessing import StandardScaler


# Load the dataset

file_path = 'winequality-red.csv'  # Update this to your actual file path

data = pd.read_csv(file_path)


# Display the original data (optional)

print("Original Data:")

print(data.head())


# Separate features from target (assuming 'quality' is the target)

X = data.drop('quality', axis=1)  # Features

y = data['quality']  # Target variable


# Initialize the StandardScaler

scaler = StandardScaler()


# Fit and transform the data

X_standardized = scaler.fit_transform(X)


# Convert the standardized data back to a DataFrame

X_standardized_df = pd.DataFrame(X_standardized, columns=X.columns)


# Display the standardized data
```

```
print("\nStandardized Data:")

print(X_standardized_df.head())


# Optional: Save the standardized data to a new CSV file

standardized_data = pd.concat([X_standardized_df, y], axis=1)

standardized_data.to_csv('standardized_winequality_red.csv', index=False)
```

Q2 B)

HTML FILE

```
<html>
<body>
<form action="slip_9.php"method="get">
Enter a string:<input type="text" name="str1"><br>
choose a label
<select name="sep" id="sep">
<option value="#">#</option>
<option value="!">!</option>
<option value="@">@</option>
</select><br>
<input type="radio" name="op" value="a">Split The String Using The Seperator<br> <input
type="radio" name="op" value="b">Replace The Occurences Of The Seperator With Another
Seperator<br>
<input type="radio" name="op" value="c">Find The Last Word Of The String<br> <input
type="submit"value="submit">
</form>
</body>
</html>
```

PHP FILE

```
<?php
$str=$_GET['str1'];
$sep=$_GET['sep'];
$op=$_GET['op'];

//echo "$str $sep $op"; switch($op)
{

case 'a':$m=explode($sep,$str); foreach($m as $a){

echo "$a<br>"; }

break;

case 'b':
$cnt=substr_count($str,$sep); $n=str_replace($sep,"!",$str,$cnt); echo "After changing
separators<br>"; echo $n;
break;

case 'c':$ar=explode(" ",$str); $cnt=count($ar);
```

```
echo "This is the last word : ".$ar[$cnt-1];

break; }

?>
```

Q2)

```
import numpy as np

import matplotlib.pyplot as plt


# Step 1: Generate random integers

data = np.random.randint(1, 100, size=50)


# Step 2: Create a figure and axis

plt.figure(figsize=(12, 6))


# Step 3: Create a line chart

plt.subplot(1, 2, 1)

plt.plot(data, color='blue', marker='o', linestyle='-', markersize=5)

plt.title('Line Chart of Random Integers')

plt.xlabel('Index')

plt.ylabel('Value')

plt.grid(True)


# Step 4: Create a scatter plot

plt.subplot(1, 2, 2)

plt.scatter(range(len(data)), data, color='red', s=50)

plt.title('Scatter Plot of Random Integers')
```

```
plt.xlabel('Index')

plt.ylabel('Value')

plt.grid(True)


# Step 5: Show the plots

plt.tight_layout()

plt.show()
```

Q2 B)

```
import matplotlib.pyplot as plt


# Step 1: Define the subject names and marks

subjects = ['Math', 'Science', 'English', 'History', 'Art']

marks = [85, 90, 75, 80, 95]


# Step 2: Create a pie chart

plt.figure(figsize=(8, 8))

plt.pie(marks, labels=subjects, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors)


# Step 3: Add a title

plt.title('Marks Distribution by Subject')
```

```
# Step 4: Display the pie chart
plt.axis('equal')  # Equal aspect ratio ensures the pie chart is a circle.
plt.show()
```

Q2 C)

```
import pandas as pd

# Step 1: Load the dataset
file_path = 'winequality-red.csv'  # Adjust the path if necessary
data = pd.read_csv(file_path)

# Step 2: Describing the dataset
description = data.describe()
print("Dataset Description:")
print(description)

# Step 3: Shape of the dataset
shape = data.shape
print("\nShape of the Dataset:")
print(shape)

# Step 4: Display first 3 rows from dataset
first_three_rows = data.head(3)
print("\nFirst 3 Rows of the Dataset:")
print(first_three_rows)
```

HTML FILE

```
<html> <head>

<title>Assignment 3 Q1</title> </head>
<body>

<form action="slip10.php" value="GET">

Enter Two Numbers<br>
<input type="text" name="n1"><br><br> <input type="text" name="n2"><br><br>

Select An Operation<br>
<input type="radio" name="op" value="mod">Mod Of The Two Numbers<br><br>

<input type="radio" name="op" value="power">Power Of The First Number Raised To The Second<br><br>

<input type="radio" name="op" value="sum">The Sum Of First n Numbers<br><br>

<input type="radio" name="op" value="fact">Factorial Of The Second Number<br><br>

<input type="submit" value="Submit">

</form> </body> </html>
```

PHP FILE

```
<?php $n1=$_GET["n1"]; $n2=$_GET["n2"]; $op=$_GET["op"]; function mod($n1,$n2) {
$n3=0;
if($n2!=0) $n3=$n1%$n2;
return $n3;
}
function power($n1,$n2) {
$n3=1;

for($i=1;$i<=$n2;$i++) $n3=$i*$n1;

return $n3;
}
function sum($n1) {

$n3=0; for($i=1;$i<=$n1;$i++) {
$n3=$n3+$i;
```

```
}
return $n3;
}
function fact($n2)
{
$n3=1; for($i=1;$i<=$n2;$i++) {
$n3=$n3*$i;

}
return $n3;
}
switch($op)
{
case "mod": $result=mod($n1,$n2);

echo "Mod of $n1 and $n2 is $result.";

break;
case "power":$result=power($n1,$n2);

echo "$n1 raised to $n2 is $result."; break;

case "sum": $result=sum($n1);
echo "Sum of first $n1 number is $result.";

break;
case "fact":$result=fact($n2);

echo"Factorial of $n2 is $result."; break;

} ?>
```

Q2)

```
import pandas as pd


# Step 1: Load the dataset

# Adjust the path to where you have the SOCR HeightWeight dataset

file_path = 'HeightWeight.csv'  # Change this to the correct filename if needed

data = pd.read_csv(file_path)
```

```
# Step 2: Display column-wise mean
mean_values = data.mean()
print("Column-wise Mean:")
print(mean_values)

# Step 3: Display column-wise median
median_values = data.median()
print("\nColumn-wise Median:")
print(median_values)
```

Q2 B)

```
import numpy as np

# Sample data: an array of points (2D for simplicity)
# You can replace this with your own dataset
points = np.array([
    [1, 2],
    [3, 4],
    [5, 6],
    [7, 8]
])
```

```
# Function to compute the Manhattan distance
def manhattan_distance(point1, point2):
    return np.abs(point1[0] - point2[0]) + np.abs(point1[1] - point2[1])

# Calculate the sum of Manhattan distances between all pairs of points
def total_manhattan_distance(points):
    total_distance = 0
    num_points = points.shape[0]

    for i in range(num_points):
        for j in range(i + 1, num_points):  # Avoid repeating pairs
            total_distance += manhattan_distance(points[i], points[j])

    return total_distance

# Compute and display the result
sum_distance = total_manhattan_distance(points)
print("Sum of Manhattan distances between all pairs of points:", sum_distance)
```

Q2 C)

HTML FILE

```
<html>
<head>
<link rel=stylesheet href="bootstrap.min.css">

</head>
<body>
<form >
<input type= button value="primary" class="btn btn-primary"> <input type= button value="warning" class="btn btn-warning"> <input type= button value="secondary"class="btn btn-Secondary"> <input type= button value="success" class="btn btn-success"> <input type= button value="info" class="btn btn-info">

<input type= button value="danger" class="btn btn-danger"> </body>
</html>
```

PHP FILE

Q2)

```
import pandas as pd

import matplotlib.pyplot as plt


# Step 1: Load the Iris dataset

# Make sure to replace 'iris.csv' with the correct path to your dataset

file_path = 'iris.csv'  # Change this if needed

data = pd.read_csv(file_path)


# Step 2: Get the frequency of each species

species_counts = data['species'].value_counts()
```

```
# Step 3: Create a pie chart

plt.figure(figsize=(8, 8))

plt.pie(species_counts, labels=species_counts.index, autopct='%1.1f%%', startangle=140,
colors=plt.cm.Paired.colors)


# Step 4: Add a title

plt.title('Frequency of Iris Species')


# Step 5: Display the pie chart

plt.axis('equal')  # Equal aspect ratio ensures the pie chart is a circle

plt.show()
```

Q2 B)

```
import pandas as pd


# Step 1: Load the dataset

file_path = 'winequality-red.csv'  # Change this to the correct path if needed

data = pd.read_csv(file_path)


# Step 2: Display basic statistical details

statistics = data.describe()

print("Basic Statistical Details:")
```

```
print(statistics)


# Optional: Show the data types of the columns

print("\nData Types of Each Column:")

print(data.dtypes)
```

Q2 C)

HTML FILE

```
<html>
<body>
<form action="slip12in.php"method="POST"> first number
<input type=text name=s2><br>
second number
<input type=text name=s1><br>
chose opration from below<br>
addition
<input type=radio value="1" name=op> <br> subtraction
<input type=radio value="2"name=op> <br> <input type= submit value="submit"><br>
</body>
</html>
```

PHP FILE

```
<?php $x=$_POST['s1']; $y=$_POST['s2']; $op=$_POST['op'];

function add($x=4,$y=2) {

$result=$x+$y; echo"adition is $result"; }

function sub($x=4,$y=2) {

$result=$x-$y; echo"subtraction is $result"; }

switch ($op)
{
case"1": add($x,$y);

break; case"2": sub($x,$y);

break; }

?>
```

PHP FILE

```
<?
```

```
php include'slip12.php';
 ?>
```

Q2)

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

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

# Step 2: Create a figure with subplots
fig, axs = plt.subplots(2, 2, figsize=(12, 10))

# Step 3: Line Chart
axs[0, 0].plot(data, color='blue', marker='o', linestyle='-', markersize=5)
axs[0, 0].set_title('Line Chart of Random Integers')
axs[0, 0].set_xlabel('Index')
axs[0, 0].set_ylabel('Value')
axs[0, 0].grid(True)

# Step 4: Scatter Plot
axs[0, 1].scatter(range(len(data)), data, color='red', s=50)
axs[0, 1].set_title('Scatter Plot of Random Integers')
```

```
axs[0, 1].set_xlabel('Index')

axs[0, 1].set_ylabel('Value')

axs[0, 1].grid(True)


# Step 5: Histogram

axs[1, 0].hist(data, bins=10, color='green', alpha=0.7, edgecolor='black')

axs[1, 0].set_title('Histogram of Random Integers')

axs[1, 0].set_xlabel('Value')

axs[1, 0].set_ylabel('Frequency')

axs[1, 0].grid(axis='y')


# Step 6: Box Plot

axs[1, 1].boxplot(data, patch_artist=True, boxprops=dict(facecolor='lightblue'))

axs[1, 1].set_title('Box Plot of Random Integers')

axs[1, 1].set_ylabel('Value')


# Step 7: Adjust layout and show plots

plt.tight_layout()

plt.show()
```

Q2 B)

```
import pandas as pd

import numpy as np
```

```
# Step 1: Create a DataFrame with some initial data

data = {

    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve', 'Frank', 'Grace', 'Heidi', 'Ivan', 'Judy'],

    'Salary': [70000, 80000, 75000, np.nan, 60000, 90000, np.nan, 80000, 75000, 100000],
# Includes NaN values

    'Department': ['HR', 'Finance', 'IT', 'Marketing', 'HR', 'Finance', 'IT', 'Marketing', 'Finance', 'IT']  #
Duplicate departments

}


# Step 2: Create DataFrame

df = pd.DataFrame(data)


# Step 3: Introduce duplicates

df = df.append(df.iloc[0])  # Duplicate Alice's row

df = df.append(df.iloc[1])  # Duplicate Bob's row


# Step 4: Display the original DataFrame

print("Original DataFrame:")

print(df)


# Step 5: Drop rows with any null or empty values

df_cleaned = df.dropna()


# Step 6: Reset the index of the cleaned DataFrame

df_cleaned.reset_index(drop=True, inplace=True)
```

```
# Step 7: Print the modified DataFrame

print("\nModified DataFrame (after dropping null values):")

print(df_cleaned)
```

HTML FILE

```
<html>
<head>
<head><title>NRC ChessBoard</title> <style>
.clr1
{

background-color:black; }

.clr2 {

background-color:white; }

table {

height:100%; }

</style>
<?php
echo"<table border=1>"; for($i=1;$i<=8;$i++)
{

echo"<tr>"; if($i%2==0) {

for($j=1;$j<=8;$j++) {

width:100%;

} }

else {

}

if($j%2==1)
echo"<td class=clr2></td>";

else
echo"<td class=clr1></td>";

for($j=0;$j<8;$j++) {

if($j%2==0)
echo"<td class=clr1></td>";

else
echo"<td class=clr2></td>";
```

```
}

echo "</tr>"; }

echo"</table>" ;

?> </html>
```

PHP FILE

Q2)

```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns


# Step 1: Load the Iris dataset

# Make sure to replace 'iris.csv' with the correct path to your dataset

file_path = 'iris.csv'  # Change this if needed

data = pd.read_csv(file_path)
```

```
# Step 2: Create a scatter plot

plt.figure(figsize=(10, 6))

sns.scatterplot(data=data, x='petal_length', y='petal_width', hue='species', style='species', s=100)


# Step 3: Add titles and labels

plt.title('Relationship Between Petal Length and Petal Width')

plt.xlabel('Petal Length (cm)')

plt.ylabel('Petal Width (cm)')

plt.grid(True)


# Step 4: Show the plot

plt.legend(title='Species')

plt.show()
```

Q2 B)

```
import numpy as np


# Step 1: Create a 2D array (for example)

array_2d = np.array([[1, 2, 3],

                     [4, 5, 6],

                     [7, 8, 9]])
```

```
# Step 2: Flatten the array
flattened_array = array_2d.flatten()

# Step 3: Find the maximum and minimum values
max_value = np.max(flattened_array)
min_value = np.min(flattened_array)

# Step 4: Print the results
print("Flattened Array:", flattened_array)
print("Maximum Value:", max_value)
print("Minimum Value:", min_value)
```

HTML FILE

```
<html>
<head>
<link rel="stylesheet" href="bootstrap.min.css"> </head>
<body>
<div class=container>

<div class="row bg-light" > <div class="col-4">col1

</div>
<div class="col-4">col2 </div>
<div class="col-4">col3 </div>

</div> </div>

</body> </html>
```

PHP FILE

Q2)

```
import numpy as np


# Step 1: Create a flattened array

flattened_array = np.array([1, 2, 3, 4, 5])
```

```
# Step 2: Define weights for the elements in the array
weights = np.array([0.1, 0.2, 0.3, 0.2, 0.2])  # Weights must sum to 1

# Step 3: Compute the weighted average
weighted_average = np.average(flattened_array, weights=weights)

# Step 4: Print the results
print("Flattened Array:", flattened_array)
print("Weights:", weights)
print("Weighted Average:", weighted_average)
```

Q2 B)

```
import pandas as pd

# Load the CSV file
file_path = 'advertising.csv'  # Make sure to provide the correct path to your file
data = pd.read_csv(file_path)

# Display the first few rows of the dataset
print("First few rows of the dataset:")
print(data.head())
```

```
# Get basic statistical details
statistics = data.describe()

# Display the statistical details
print("\nBasic Statistical Details:")
print(statistics)

# If you want to see the data types of each column
print("\nData Types:")
print(data.dtypes)
```

HTML FILE

```
<html>
<body>
<form action="Slip15.php" method=get>
Enter a String<input type=text name=t1><br>
<input type=radio name=op value=1>Select 5 words<br> <input type=radio name=op
value=2>LowerCase<br> <input type=radio name=op value=3>Padding<br>
<input type=radio name=op value=4>Remove Spaces<br

<input type=radio name=op value=5>Reverse<br><br> <input type=submit value=submit>
</form>
</body>

</html>
```

PHP FILE

```
<?php $str=$_GET['t1']; $ch=$_GET['op']; if($ch==1)
{

$ar=explode(" ",$str,6); foreach($ar as $a)

echo "$a<br>";

}
else if($ch==2) {

$str=strtolower($str); echo $str;

$str=ucwords($str);

echo "<br>$str"; }

else if($ch==3) {

$str=str_pad($str,20,"*",STR_PAD_BOTH);

echo $str; }

else if($ch==4) {
```

```
$str=trim($str);
echo $str; }
else if($ch==5) {
$str=strrev($str);
echo $str; }
?>
```

Q2)

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

# Generate a random array of 50 integers between 1 and 100
random_integers = np.random.randint(1, 101, size=50)

# Set up the figure and axes
fig, axs = plt.subplots(2, 2, figsize=(12, 10))

# Line Chart
axs[0, 0].plot(random_integers, color='blue', marker='o', linestyle='-', linewidth=2,
markersize=5)
```

```
axs[0, 0].set_title('Line Chart of Random Integers', fontsize=14)

axs[0, 0].set_xlabel('Index', fontsize=12)

axs[0, 0].set_ylabel('Value', fontsize=12)

axs[0, 0].grid(True)


# Scatter Plot

axs[0, 1].scatter(range(50), random_integers, color='orange', alpha=0.7)

axs[0, 1].set_title('Scatter Plot of Random Integers', fontsize=14)

axs[0, 1].set_xlabel('Index', fontsize=12)

axs[0, 1].set_ylabel('Value', fontsize=12)


# Histogram

axs[1, 0].hist(random_integers, bins=10, color='green', edgecolor='black')

axs[1, 0].set_title('Histogram of Random Integers', fontsize=14)

axs[1, 0].set_xlabel('Value', fontsize=12)

axs[1, 0].set_ylabel('Frequency', fontsize=12)


# Box Plot

axs[1, 1].boxplot(random_integers, patch_artist=True, boxprops=dict(facecolor='lightblue'))

axs[1, 1].set_title('Box Plot of Random Integers', fontsize=14)

axs[1, 1].set_ylabel('Value', fontsize=12)


# Adjust layout

plt.tight_layout()
```

```
# Show the plots

plt.show()
```

Q2 B)

```
import matplotlib.pyplot as plt


# Define the subjects and their corresponding marks

subjects = ['Math', 'Science', 'English', 'History', 'Art']

marks = [85, 90, 78, 88, 92]


# Create a pie chart

plt.figure(figsize=(8, 8))

plt.pie(marks, labels=subjects, autopct='%1.1f%%', startangle=140,
colors=plt.cm.Paired.colors)

plt.title('Marks Distribution by Subject', fontsize=16)

plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.


# Show the plot

plt.show()
```

Q1

```
 <html>
<head>
<title> marksheet </title>
</head>
<body>
<form name = "string" action = "slip16.php" method = "get">
student id <input type = "text" name = "v1"/></br></br>
Subject name<input type = "text" name = "v2"/></br></br>
subject marks<input type = "text" name = "v3"/></br></br>
<button type = "submit"> display marklist </button>
</form>
</body>
</html>
```

```
<?php
$a =$_GET['v1'];
$b =$_GET['v2'];
$c =$_GET['v3'];
$sum=0;
echo "<h1> <center>Marksheet</center></h1>";
echo "<h3><center>student id:$a</h3><br>";
$d =explode(",",$b);
$e =explode(",",$c);
echo "<center><table border=2 width=50%>";
 for($i=0;$i<=4;$i++)
```

```
{
 echo "<tr>
 <td>$d[$i]</td>
 <td>$e[$i]</td>
 </tr>";
 $sum=$e[$i]+$sum;
}
$result=$sum/5;
echo "<tr><td>Total marks </td><td>$sum</td>";
echo"<tr><td>Percentage</td><td>$result</td></tr>";
echo "</center>"
?>
```

Q2

A)

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler

# Step 1: Load the dataset
data = pd.read_csv('winequality-red.csv')

# Step 2: Separate features and labels (if necessary)
# Assuming 'quality' is the label and the rest are features
X = data.drop('quality', axis=1)  # Features
```

```
y = data['quality']              # Target variable

# Step 3: Initialize MinMaxScaler
scaler = MinMaxScaler()

# Step 4: Fit and transform the features
X_scaled = scaler.fit_transform(X)

# Step 5: Convert back to DataFrame (optional)
X_scaled_df = pd.DataFrame(X_scaled, columns=X.columns)

# Optional: Combine scaled features with the target variable
scaled_data = pd.concat([X_scaled_df, y.reset_index(drop=True)], axis=1)

# Display the first few rows of the scaled dataset
print(scaled_data.head())
```

B)

```
import pandas as pd
# Sample data for students
data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva'],
    'Graduation Percentage': [85.5, 90.2, 78.4, 88.0, 92.5],
    'Age': [20, 21, 22, 20, 19]
```

```
}

# Create a DataFrame

students_df = pd.DataFrame(data)


# Display the DataFrame

print("Students Information:")

print(students_df)


# Calculate average age

average_age = students_df['Age'].mean()

# Calculate average graduation percentage

average_graduation_percentage = students_df['Graduation Percentage'].mean()


# Display the averages

print(f"\nAverage Age of Students: {average_age:.2f}")

print(f"Average Graduation Percentage: {average_graduation_percentage:.2f}%")
```

Q1)

```
<?php
$a = array("Sagar"=>"31","Vicky"=>"41","Leena"=>"39","Ramesh"=>"40");
echo "sorting in ascending order by value<br>";
asort($a);
print_r($a);
echo "sorting in ascending order by key<br>";
ksort($a);
print_r($a);
echo "sorting in decending order by value<br>";
arsort($a);
print_r($a);
echo "sorting in decending order by key<br>";
krsort($a);
print_r($a);
?>
```

Q2)

A)

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

# Load the Iris dataset
iris = sns.load_dataset('iris')
```

```
# Display the first few rows of the dataset
print(iris.head())


# Create scatter plots to compare two features
# Feature combinations to compare
feature_pairs = [('sepal_length', 'sepal_width'),
                 ('petal_length', 'petal_width'),
                 ('sepal_length', 'petal_length'),
                 ('sepal_width', 'petal_width')]


# Set up the matplotlib figure
plt.figure(figsize=(12, 10))


# Loop through feature pairs and create scatter plots
for i, (feature_x, feature_y) in enumerate(feature_pairs):
    plt.subplot(2, 2, i + 1)
    sns.scatterplot(data=iris, x=feature_x, y=feature_y, hue='species', palette='deep')
    plt.title(f'Scatter Plot of {feature_x} vs {feature_y}')
    plt.xlabel(feature_x)
    plt.ylabel(feature_y)


# Adjust layout
plt.tight_layout()
plt.show()
```

B)

```
import matplotlib.pyplot as plt

# Data: Subject names and corresponding marks
subjects = ['Math', 'Science', 'English', 'History', 'Art']
marks = [85, 90, 75, 80, 95]

# Create a Pie Chart
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)  # 1 row, 2 columns, 1st subplot
plt.pie(marks, labels=subjects, autopct='%1.1f%%', startangle=140)
plt.title('Marks Distribution by Subject')
plt.axis('equal')  # Equal aspect ratio ensures that pie chart is circular.

# Create a Bar Chart
plt.subplot(1, 2, 2)  # 1 row, 2 columns, 2nd subplot
plt.bar(subjects, marks, color='skyblue')
plt.title('Marks Obtained in Each Subject')
plt.xlabel('Subjects')
plt.ylabel('Marks')

# Adjust layout
plt.tight_layout()
```

```
plt.show()
```

Q1)

```
<html>
<head>
<title> String operations </title>
<body>
<form name = "String" action = "s18_q1.php" method = "get">
<h2> Select Operations</h2>
<input type = "radio" name ="s" value=1>Reverse the order</br></br>
<input type = "radio" name ="s" value=2>Traverse the element</br></br>
<input type = "radio" name ="s" value=3>Convert the array elements</br></br>
<input type = "radio" name ="s" value=4>Display the elements of an array along with
key.</br></br>
<input type=submit value=send>
</body>
</html>
```

```
<?php
$a = array("a"=>1,"b"=>21,"c"=> 56);
switch($_GET['s'])
{
case 1:
 echo "Orginal Array is <br>";
 print_r($a);
 echo "reverse Array is <br>";
 $c =array_reverse($a);
 print_r($c);
```

```
 break;
case 2:
 echo "Orginal Array is <br>";
 print_r($a);
 echo "traversing Array is <br>";
 foreach($a as $v)
 echo "$v<br>";
 break;
case 3:
 echo " Original array is<br>";
 print_r($a);
 echo "elements into individual variables<br>";
 $e = extract($a);
 print_r($e);
 break;
case 4:
 echo " Original array is<br>";
 print_r($a);
 echo "key value pair is<br>";
 foreach ($a as $k=>$v)
 {
 print_r("$k=>$v");
 echo"<br>";
 }
 break;
```

```
default:
 echo "Invalid choice!!";
}
?>
```

Q2)

A)

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

# Load the iris dataset
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/iris.csv"
column_names = ['Sepal Length', 'Sepal Width', 'Petal Length', 'Petal Width', 'Species']
iris = pd.read_csv(url, header=None, names=column_names)

# Set the aesthetic style of the plots
sns.set(style="whitegrid")

# Create box plots for each feature
features = ['Sepal Length', 'Sepal Width', 'Petal Length', 'Petal Width']
for feature in features:
    plt.figure(figsize=(10, 6))
    sns.boxplot(x='Species', y=feature, data=iris)
```

```
    plt.title(f'Box plot of {feature} by Species')
    plt.xlabel('Species')
    plt.ylabel(feature)
    plt.show()
```

B)

```
import pandas as pd

# Load the dataset from a CSV file
file_path = 'heights_weights.csv'  # Replace with your actual file path
data = pd.read_csv(file_path)

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

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

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

Q1)

```
<html>

<form action=slip19.php method=get>

Enter a Sentence<input type=text name=t1><br>

Enter a word<input type=text name=t2><br>

Enter position<input type=text name=t3><br>

Enter number of characters to remove<input type=text name=t4><br>

<input type=submit value="Display Result">

</form>

</html>
```

```
<?php

$st=$_GET['t1'];

$wd=$_GET['t2'];

$ps=$_GET['t3'];

$nr=$_GET['t4'];

$dup_st=$st;

$str=substr_replace($st,"",$ps,$nr);

echo "<br>$str<br>";

$str=substr_replace($st,$wd,$ps,0);

echo "<br>$str<br>";

$str=substr_replace($st,$wd,$ps,strlen($wd));

echo "<br>$str<br>";

?>
```

Q2)

```
import pandas as pd

import numpy as np


# 1. Create a DataFrame with columns name, age, and percentage

data = {

   'name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva',

            'Frank', 'Grace', 'Hannah', 'Ian', 'Judy'],

   'age': [23, 25, 22, 24, 26,

           21, 27, 28, 29, 30],

   'percentage': [85.5, 90.0, 78.0, 88.5, 91.0,

                  72.5, 84.0, 95.5, 80.0, 75.5]

}


df = pd.DataFrame(data)


# View the DataFrame

print("Initial DataFrame:")

print(df)


# 2. Print shape, number of rows-columns, data types, feature names and description

print("\nDataFrame Shape:", df.shape)

print("Number of Rows:", df.shape[0])

print("Number of Columns:", df.shape[1])

print("Data Types:")
```

```
print(df.dtypes)

print("Feature Names:", df.columns.tolist())

print("Description:")

print(df.describe())


# 3. Add 5 rows with duplicate values and missing values

duplicate_data = {

   'name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva',

            None, 'Frank', 'Grace', None, 'Hannah'],

   'age': [23, 25, 22, 24, 26,

           23, None, 28, 29, 30],

   'percentage': [85.5, 90.0, 78.0, 88.5, 91.0,

                  85.5, 92.0, None, 80.0, 75.5]

}


duplicate_df = pd.DataFrame(duplicate_data)


# Add a 'remarks' column with empty values

duplicate_df['remarks'] = ""


# Combine original and duplicate DataFrames

combined_df = pd.concat([df, duplicate_df], ignore_index=True)


# Display the combined data

print("\nCombined DataFrame with Duplicates and Missing Values:")
```

```
print(combined_df)
```

Q1)

SLIP 20

```
<html>
 <body>
 <form action="slip20.php" method="GET">
Enter Your Choice:<br>
 <input type="radio" name="ch" value=1> Split an array into chunks<br>
 <input type="radio" name="ch" value=2> Sort array by values without changing key
values <br>
 <input type="radio" name="ch" value=3> Filter even elements from array <br>
 <input type="submit" value="Submit">
 </form>
 </body>
</html>


<?php
$choice=$_GET['ch'];
$arr=array('a'=>1,'b'=>20,'c'=>13,'d'=>5,'e'=>18,'f'=>12,'g'=>7,'h'=>8,'i'=>15,'j'=>10);


 switch($choice)
 {
 case 1:
 print_r(array_chunk($arr,2));
 break;


 case 2:
 asort($arr);
```

```
echo "Array in ascending order:<br>";
print_r($arr);
break;

case 3:
function even($var)
{
return $var%2==0?1:0;

}
$br=array_filter($arr,"even");
print_r($br);
break;
}
?>
```

Q2)

A)

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

# Generate a random array of 50 integers between 1 and 100
random_array = np.random.randint(1, 101, size=50)
```

```
# Set up the figure and axes

fig, axs = plt.subplots(2, 2, figsize=(12, 10))


# 1. Line Chart

axs[0, 0].plot(random_array, color='blue', marker='o', linestyle='-')

axs[0, 0].set_title('Line Chart of Random Integers', fontsize=14)

axs[0, 0].set_xlabel('Index', fontsize=12)

axs[0, 0].set_ylabel('Value', fontsize=12)

axs[0, 0].grid(True)


# 2. Scatter Plot

axs[0, 1].scatter(range(len(random_array)), random_array, color='green', alpha=0.7)

axs[0, 1].set_title('Scatter Plot of Random Integers', fontsize=14)

axs[0, 1].set_xlabel('Index', fontsize=12)

axs[0, 1].set_ylabel('Value', fontsize=12)

axs[0, 1].grid(True)


# 3. Histogram

axs[1, 0].hist(random_array, bins=10, color='orange', edgecolor='black', alpha=0.7)

axs[1, 0].set_title('Histogram of Random Integers', fontsize=14)

axs[1, 0].set_xlabel('Value', fontsize=12)

axs[1, 0].set_ylabel('Frequency', fontsize=12)


# 4. Box Plot

axs[1, 1].boxplot(random_array, patch_artist=True, boxprops=dict(facecolor='lightblue',
color='blue'))
```

```
axs[1, 1].set_title('Box Plot of Random Integers', fontsize=14)

axs[1, 1].set_ylabel('Value', fontsize=12)


# Adjust layout

plt.tight_layout()


# Show the plots

plt.show()
```

B)

```
import numpy as np

import matplotlib.pyplot as plt


# Generate a random array of 50 integers between 1 and 100

random_array = np.random.randint(1, 101, size=50)


# Add two outliers

outliers = [150, 200]  # Adding two high outliers

data_with_outliers = np.concatenate((random_array, outliers))


# Set up the figure

plt.figure(figsize=(8, 6))


# Box Plot

plt.boxplot(data_with_outliers, patch_artist=True, boxprops=dict(facecolor='lightblue',
color='blue'))
```

```
plt.title('Box Plot of Random Integers with Outliers', fontsize=14)

plt.ylabel('Value', fontsize=12)


# Show the plot

plt.grid(True)

plt.show()
```

Q1)

```
<?php
$temp_array=range(31,45);
$tot_temp = 0;
$count = count($temp_array);
echo "Total temp values are: ".$count;
foreach($temp_array as $temp)
{
$tot_temp += $temp;
}
$avg_high_temp = $tot_temp/$count;
echo "<br> Average Temperature is : ".$avg_high_temp."
";
sort($temp_array);
echo " <br> List of five lowest temperatures :";
$res1= array_slice($temp_array,0,5);
foreach($res1 as $high_temp)
{
echo "<br> $high_temp";
}
echo "<br> List of five highest temperatures :";
$res1= array_slice($temp_array,10);
foreach($res1 as $high_temp)
{
echo "<br> $high_temp";
}
```

?>

Q2)

A)

```
# Import necessary libraries

import pandas as pd

import matplotlib.pyplot as plt


# Load the dataset

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


# Check the first few rows to understand the data structure

# print(data.head())


# Get the frequency of each species

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


# Create the bar plot

plt.figure(figsize=(8, 5))

species_count.plot(kind='bar', color=['lightblue', 'lightgreen', 'lightcoral'])


# Set title and labels

plt.title('Frequency of Iris Species', fontsize=16)

plt.xlabel('Species', fontsize=12)
```

```
plt.ylabel('Frequency', fontsize=12)

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

B)

```
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt

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

# List of species
species = data['species'].unique()

# Set the figure size
plt.figure(figsize=(10, 6))

# Plot a histogram for each species for the sepal_length
for sp in species:
    subset = data[data['species'] == sp]
    plt.hist(subset['sepal_length'], alpha=0.5, label=sp, bins=10)

# Set title and labels
```

```
plt.title('Histogram of Sepal Length by Species', fontsize=16)

plt.xlabel('Sepal Length (cm)', fontsize=12)

plt.ylabel('Frequency', fontsize=12)


# Show the legend

plt.legend(title='Species')


# Show the plot

plt.show()
```

Q1)

```
<html>
<body>
<form method=get action="slip 23.php">
<input type=radio name=ch value=1> Insert in queue<br>
Enter No to insert<input type=text name=n1><br>
<input type=radio name=ch value=2> Delete in queue<br>
<input type=radio name=ch value=3> Display<br>
<input type=submit>
</form>
</body>
</html>
```

```
<?php
$stk=array(1,2,3,4,5);
$ch=$_GET['ch'];
else if($ch==1)
{
 echo "Insert element in queue <br>";
 $n4=$_GET['n4'];
 array_push($stk,$n4);//Insert element at last
 print_r($stk);
}
else if($ch==2)
{
```

```
 echo "Delete element from queue. <br>";
 $res=array_shift($stk); //at begining
 echo "Deleted element is:".$res;
}
else if($ch ==3)
{
 echo"Given array is: <br>";
 print_r($stk);
}
?>
```

Q2)

```
# Import necessary libraries
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Normalizer

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

# Display first few rows of the dataset
# print(data.head())

# Features and target separation (assuming 'quality' is the target variable)
X = data.drop(columns=['quality'])  # Features (independent variables)
```

```
y = data['quality']  # Target (dependent variable)

# Task a: Rescaling (Normalization) using MinMaxScaler
min_max_scaler = MinMaxScaler()
X_minmax = min_max_scaler.fit_transform(X)

# Convert rescaled data back to a DataFrame for readability
X_minmax_df = pd.DataFrame(X_minmax, columns=X.columns)
print("MinMax Scaled Data (first 5 rows):")
print(X_minmax_df.head())

# Task b: Standardizing Data using StandardScaler (mean=0, std=1)
standard_scaler = StandardScaler()
X_standardized = standard_scaler.fit_transform(X)

# Convert standardized data back to a DataFrame for readability
X_standardized_df = pd.DataFrame(X_standardized, columns=X.columns)
print("\nStandardized Data (first 5 rows):")
print(X_standardized_df.head())

# Task c: Normalizing Data using Normalizer (unit norm)
normalizer = Normalizer()
X_normalized = normalizer.fit_transform(X)

# Convert normalized data back to a DataFrame for readability
```

```
X_normalized_df = pd.DataFrame(X_normalized, columns=X.columns)

print("\nNormalized Data (first 5 rows):")

print(X_normalized_df.head())
```

Q1) HTML FILE


```
<html>
<body>
<form method=get action="slip23.php">
<input type=radio name=ch value=1> Insert in stack<br>
Enter No to insert<input type=text name=n1><br>
<input type=radio name=ch value=2> Delete in stack<br>
<input type=radio name=ch value=3> Display<br>
<input type=submit>
</form>
</body>
</html>
```

PHP FILE

```
<?php
$stk=array(1,2,3,4,5);
$ch=$_GET['ch'];
if($ch==1)
{
echo "Insert element in stack <br>";
 $n1=$_GET['n1'];
 array_push($stk,$n1); //at end
 print_r($stk);
}
else if($ch==2)
{
```

```
 echo "Delete element in stack <br>";
 array_pop($stk);
 print_r($stk);
}
else if($ch == 3)
{
 echo"Given array is: <br>";
 print_r($stk);
}
?>
```

Q2)

```
# Import necessary libraries
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Binarizer

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

# Display the first few rows of the dataset
# print(data.head())

# Features and target separation (assuming 'quality' is the target variable)
X = data.drop(columns=['quality'])  # Features (independent variables)
```

```
y = data['quality']  # Target (dependent variable)

# Task a: Rescaling (Normalization) using MinMaxScaler
min_max_scaler = MinMaxScaler()
X_minmax = min_max_scaler.fit_transform(X)

# Convert rescaled data back to a DataFrame for readability
X_minmax_df = pd.DataFrame(X_minmax, columns=X.columns)
print("MinMax Scaled Data (first 5 rows):")
print(X_minmax_df.head())

# Task b: Standardizing Data using StandardScaler (mean=0, std=1)
standard_scaler = StandardScaler()
X_standardized = standard_scaler.fit_transform(X)

# Convert standardized data back to a DataFrame for readability
X_standardized_df = pd.DataFrame(X_standardized, columns=X.columns)
print("\nStandardized Data (first 5 rows):")
print(X_standardized_df.head())

# Task c: Binarizing Data using Binarizer (with threshold)
threshold_value = 0.5  # Set the threshold value for binarization
binarizer = Binarizer(threshold=threshold_value)
X_binarized = binarizer.fit_transform(X)
```

```
# Convert binarized data back to a DataFrame for readability
X_binarized_df = pd.DataFrame(X_binarized, columns=X.columns)
print("\nBinarized Data (first 5 rows, threshold=0.5):")
print(X_binarized_df.head())
```

Q1) HTML FILE

```
<html lang="en">
<head>
 <title>NRC File Handling</title>
</head>
<body>
 <form action="slip24.php" method="get">
 Enter File name to read <input type="text" name="fname1">
 Enter File name to write <input type="text" name="fname2">
 <input type="submit" value="Copy">
 </form>
</body>
</html>
```

PHP FILE

```
<?php
$fname1=$_GET['fname1'];
$fname2=$_GET['fname2'];
$fp1=fopen($fname1,"r");
$fp2=fopen($fname2,"a");
$size=filesize($fname1);
$str=fread($fp1,$size);
fwrite($fp2,$str,$size);
echo "Append Successfull";
?>
```

Q2)

A)

```
# Import necessary libraries

import pandas as pd

import matplotlib.pyplot as plt


# Load the dataset

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


# Get the frequency of each species

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


# Create the bar plot

plt.figure(figsize=(8, 5))

species_count.plot(kind='bar', color=['lightblue', 'lightgreen', 'lightcoral'])


# Set title and labels

plt.title('Frequency of Iris Species', fontsize=16)

plt.xlabel('Species', fontsize=12)

plt.ylabel('Frequency', fontsize=12)


# Show the plot

plt.show()
```

B)

```
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt

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

# List of species
species = data['species'].unique()

# Set the figure size
plt.figure(figsize=(10, 6))

# Plot a histogram for each species for the sepal_length
for sp in species:
    subset = data[data['species'] == sp]
    plt.hist(subset['sepal_length'], alpha=0.5, label=sp, bins=10)

# Set title and labels
plt.title('Histogram of Sepal Length by Species', fontsize=16)
plt.xlabel('Sepal Length (cm)', fontsize=12)
plt.ylabel('Frequency', fontsize=12)

# Show the legend
plt.legend(title='Species')
```

```
# Show the plot

plt.show()
```

Q1) HTML FILE

```
<html lang="en">
<head>
 <title>Document</title>
</head>
<body>
 <form action="slip25.php" method="get">
 Enter File name to read <input type="text" name="fname1"><br>
 <input type="radio" name="op" value="1">Display Type of File<br>
 <input type="radio" name="op" value="2">Display Modification Time<br>
 <input type="radio" name="op" value="3">Display File Size<br>
 <input type="radio" name="op" value="4">Delete File<br>
 <input type="submit" value="Copy">
</form>
</body>
</html>
```

PHP FILE

```
<?php
$fp=$_GET['fname1'];
$op=$_GET['op'];
switch($op)
{
 case 1:
 echo basename($fp);
 break;
```

```
 case 2: $mtime=stat($fp);
 echo Date("d/M/Y h:m:s",$mtime['mtime']);
 break;
 case 3:echo filesize($fp);
 break;
 case 4:unlink($fp);
 echo "File deleted";
 break;
}
```

Q2)

A)

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt

# Generate a random array of 50 integers between 1 and 100
data = np.random.randint(1, 101, size=50)

# Set up a figure with multiple subplots
plt.figure(figsize=(14, 10))

# Line chart
plt.subplot(2, 2, 1)
plt.plot(data, color='blue', marker='o', linestyle='-', linewidth=2)
```

```
plt.title('Line Chart of Random Integers', fontsize=14)
plt.xlabel('Index', fontsize=12)
plt.ylabel('Value', fontsize=12)
plt.grid(True)

# Scatter plot
plt.subplot(2, 2, 2)
plt.scatter(range(len(data)), data, color='red', marker='x')
plt.title('Scatter Plot of Random Integers', fontsize=14)
plt.xlabel('Index', fontsize=12)
plt.ylabel('Value', fontsize=12)
plt.grid(True)

# Histogram
plt.subplot(2, 2, 3)
plt.hist(data, bins=10, color='green', edgecolor='black')
plt.title('Histogram of Random Integers', fontsize=14)
plt.xlabel('Value', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.grid(True)

# Box plot
plt.subplot(2, 2, 4)
plt.boxplot(data, patch_artist=True, boxprops=dict(facecolor='orange', color='black'))
plt.title('Box Plot of Random Integers', fontsize=14)
```

```
plt.ylabel('Value', fontsize=12)

plt.grid(True)


# Adjust layout to prevent overlap

plt.tight_layout()


# Show the plots

plt.show()
```

B)

```
# Import necessary libraries

import matplotlib.pyplot as plt


# Create two lists: one for subjects and one for marks

subjects = ['Math', 'Physics', 'Chemistry', 'English', 'Biology']

marks = [85, 90, 78, 92, 88]


# Create a pie chart

plt.figure(figsize=(8, 8))

plt.pie(marks, labels=subjects, autopct='%1.1f%%', startangle=90, colors=['lightblue',
'lightgreen', 'lightcoral', 'gold', 'lightpink'])


# Set the title

plt.title('Marks Distribution by Subject', fontsize=16)
```

```
# Equal aspect ratio ensures that pie is drawn as a circle

plt.axis('equal')


# Show the pie chart

plt.show()
```

Q1) HTML FILE

```
<html>
<form action=slip26.php method=get>
Enter Hospital Name<input type=text name=t1><br>
<input type=submit value="Display Doctors">
</form>
</html>
```

PHP File

```
<?php
$con=pg_connect("host=localhost user=postgres password=nrc dbname=slip26");
$cn=$_GET['t1'];
$rs=pg_query($con,"select * from doctor,hospital where hname='$cn' and hsno=hno");
while($row=pg_fetch_array($rs))
{
 echo "Id:$row[0] Name:$row[1] Address:$row[2] City:$row[3]
Pin:$row[4]<br>";
}
?>
```

Q2)

A)

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
```

```
# Generate a random array of 50 integers between 1 and 100

data = np.random.randint(1, 101, size=50)


# Set up a figure with multiple subplots

plt.figure(figsize=(14, 10))


# Line chart

plt.subplot(2, 2, 1)

plt.plot(data, color='blue', marker='o', linestyle='-', linewidth=2)

plt.title('Line Chart of Random Integers', fontsize=14)

plt.xlabel('Index', fontsize=12)

plt.ylabel('Value', fontsize=12)

plt.grid(True)


# Scatter plot

plt.subplot(2, 2, 2)

plt.scatter(range(len(data)), data, color='red', marker='x', s=100)  # s controls the size of points

plt.title('Scatter Plot of Random Integers', fontsize=14)

plt.xlabel('Index', fontsize=12)

plt.ylabel('Value', fontsize=12)

plt.grid(True)


# Histogram

plt.subplot(2, 2, 3)
```

```
plt.hist(data, bins=10, color='green', edgecolor='black', alpha=0.7)

plt.title('Histogram of Random Integers', fontsize=14)

plt.xlabel('Value', fontsize=12)

plt.ylabel('Frequency', fontsize=12)

plt.grid(axis='y')


# Box plot

plt.subplot(2, 2, 4)

plt.boxplot(data, patch_artist=True, boxprops=dict(facecolor='orange', color='black'))

plt.title('Box Plot of Random Integers', fontsize=14)

plt.ylabel('Value', fontsize=12)

plt.grid(True)


# Adjust layout to prevent overlap

plt.tight_layout()


# Show the plots

plt.show()
```

B)

```
# Import necessary libraries

import matplotlib.pyplot as plt


# Create two lists: one for subjects and one for marks
```

```
subjects = ['Math', 'Physics', 'Chemistry', 'English', 'Biology']

marks = [85, 90, 78, 92, 88]


# Create a bar chart

plt.figure(figsize=(10, 6))

plt.bar(subjects, marks, color='skyblue', edgecolor='black')


# Set the title and labels

plt.title('Marks Obtained in Subjects', fontsize=16)

plt.xlabel('Subjects', fontsize=14)

plt.ylabel('Marks', fontsize=14)


# Show the plot

plt.grid(axis='y', linestyle='--', alpha=0.7)  # Add gridlines for better readability

plt.xticks(rotation=15)  # Rotate subject labels for better visibility

plt.ylim(0, 100)  # Set y-axis limits


plt.show()
```

Q1) HTML FILE

```
<html lang="en">
<head>
 <title>NRC File Handling</title>
</head>
<body>
 <form action="slip27.php" method="get">
 Enter File name to read <input type="text" name="fname1">
 Enter File name to write <input type="text" name="fname2">
 <input type="submit" value="Copy">
 </form>
</body>
</html>
```

PHP FILE

```
<?php
$fname1=$_GET['fname1'];
$fname2=$_GET['fname2'];
$fp1=fopen($fname1,"r");
$fp2=fopen($fname2,"w");
$size=filesize($fname1);
$str=fread($fp1,$size);
fwrite($fp2,$str,$size);
echo "Append Successfull";
?>
```

Q2)

```
import pandas as pd

# Create a sample dataset
data = {
    'country': ['USA', 'Canada', 'USA', 'Mexico', 'Canada', 'Mexico', 'USA', 'Canada'],
    'purchased': ['Yes', 'No', 'Yes', 'No', 'Yes', 'Yes', 'No', 'Yes']
}

# Convert to DataFrame
df = pd.DataFrame(data)

# Save the DataFrame to a CSV file
df.to_csv('data.csv', index=False)

# Display the created DataFrame
print("Created Dataset:")
print(df)

from sklearn.preprocessing import OneHotEncoder, LabelEncoder

# Load the dataset
df = pd.read_csv('data.csv')
```

```
# Apply One-Hot Encoding to the 'country' column
one_hot_encoder = OneHotEncoder(sparse=False)
country_encoded = one_hot_encoder.fit_transform(df[['country']])

# Convert the encoded array to a DataFrame
country_encoded_df = pd.DataFrame(country_encoded,
columns=one_hot_encoder.get_feature_names_out(['country']))

# Combine the original DataFrame with the new one
df_one_hot = pd.concat([df, country_encoded_df], axis=1)

# Apply Label Encoding to the 'purchased' column
label_encoder = LabelEncoder()
df_one_hot['purchased_encoded'] = label_encoder.fit_transform(df_one_hot['purchased'])

# Display the final DataFrame
print("\nFinal DataFrame with Encodings:")
print(df_one_hot)
```

Q1) HTML FILE

```
<html>
<form action=slip29.php method=get>
Enter Event Name:<input type=text name=t1><br>
<input type=submit value="Change Status">
</form>
</html>
```

PHP File

```
<?php
 $con=pg_connect("host=localhost user=postgres password=nrc bname=practicals22");
//echo $con;
$en=$_GET['t1'];
$ws='Working';
$x=pg_query($con,"update comm_mem set cstatus='$ws' from event_comm,event
where comm_mem.cno=event_comm.cno and event_comm.eno=event.eno and
etitle='$en'");
if($x>0)
 echo "Working status updated";
else
 echo "Status not updated";
 ?>
```

Q2) DATASET

```
import pandas as pd
```

```
# Create a sample dataset with categorical columns
data = {
   'country': ['USA', 'Canada', 'USA', 'Mexico', 'Canada',
               'Mexico', 'USA', 'Canada', 'Mexico', 'USA'],
   'purchased': ['Yes', 'No', 'Yes', 'No', 'Yes',
                 'Yes', 'No', 'Yes', 'No', 'Yes']
}

# Convert the dictionary to a DataFrame
df = pd.DataFrame(data)

# Save the DataFrame to a CSV file
df.to_csv('data.csv', index=False)

# Display the created DataFrame
print("Created Dataset:")
print(df)



from sklearn.preprocessing import OneHotEncoder, LabelEncoder

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

# 1. Apply One-Hot Encoding to the 'country' column
```

```
one_hot_encoder = OneHotEncoder(sparse=False)

country_encoded = one_hot_encoder.fit_transform(df[['country']])


# Convert the encoded array to a DataFrame

country_encoded_df = pd.DataFrame(country_encoded,
columns=one_hot_encoder.get_feature_names_out(['country']))


# Combine the original DataFrame with the new one

df_one_hot = pd.concat([df, country_encoded_df], axis=1)


# 2. Apply Label Encoding to the 'purchased' column

label_encoder = LabelEncoder()

df_one_hot['purchased_encoded'] = label_encoder.fit_transform(df_one_hot['purchased'])


# Display the final DataFrame with encodings

print("\nFinal DataFrame with Encodings:")

print(df_one_hot)
```

Q1) HTML FILE

```
<html>

<form action=slip30.php method=get>

Enter Comp. Name<input type=text name=t1><br>

<input type=submit value="Display Ranker">

</form>

</html>
```

PHP File

```
<?php

$con=pg_connect("host=localhost user=postgres password=nrc dbname=slip30");

$cn=$_GET['t1'];

$rs=pg_query($con,"select * from student,competition,stud_comp where cname='$cn'

and srank=1 and id=sid and competition.cno=stud_comp.cno");

while($row=pg_fetch_array($rs))

{

 echo "$row[0] $row[1] $row[2]<br>";

}

?>
```

Q2)

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt
```

```
# Part a: Generate a random array of 50 integers and display various plots


# Generate a random array of 50 integers between 1 and 100

data = np.random.randint(1, 101, size=50)


# Set up a figure with multiple subplots

plt.figure(figsize=(14, 10))


# Line chart

plt.subplot(2, 2, 1)

plt.plot(data, color='blue', marker='o', linestyle='-', linewidth=2)

plt.title('Line Chart of Random Integers', fontsize=14)

plt.xlabel('Index', fontsize=12)

plt.ylabel('Value', fontsize=12)

plt.grid(True)


# Scatter plot

plt.subplot(2, 2, 2)

plt.scatter(range(len(data)), data, color='red', marker='x', s=100)  # s controls the size of points

plt.title('Scatter Plot of Random Integers', fontsize=14)

plt.xlabel('Index', fontsize=12)

plt.ylabel('Value', fontsize=12)

plt.grid(True)


# Histogram
```

```
plt.subplot(2, 2, 3)
plt.hist(data, bins=10, color='green', edgecolor='black', alpha=0.7)
plt.title('Histogram of Random Integers', fontsize=14)
plt.xlabel('Value', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.grid(axis='y')

# Box plot
plt.subplot(2, 2, 4)
plt.boxplot(data, patch_artist=True, boxprops=dict(facecolor='orange', color='black'))
plt.title('Box Plot of Random Integers', fontsize=14)
plt.ylabel('Value', fontsize=12)
plt.grid(True)

# Adjust layout to prevent overlap
plt.tight_layout()

# Show the plots
plt.show()

# Part b: Create two lists for subject names and marks and display them in a bar chart

# Create two lists: one for subjects and one for marks
subjects = ['Math', 'Physics', 'Chemistry', 'English', 'Biology']
marks = [85, 90, 78, 92, 88]
```

```
# Create a bar chart
plt.figure(figsize=(10, 6))
plt.bar(subjects, marks, color='skyblue', edgecolor='black')

# Set the title and labels
plt.title('Marks Obtained in Subjects', fontsize=16)
plt.xlabel('Subjects', fontsize=14)
plt.ylabel('Marks', fontsize=14)

# Show the plot
plt.grid(axis='y', linestyle='--', alpha=0.7)  # Add gridlines for better readability
plt.xticks(rotation=15)  # Rotate subject labels for better visibility
plt.ylim(0, 100)  # Set y-axis limits

plt.show()
```