M-2

Q.1] Simple Complete Python Programme to Demonstrate Creating Database, Creating Customer table, inserting, Selecting, Deleting and updating rows from table.

```
import mysql.connector
connection = mysql.connector.connect(
    host = "localhost",
    user = "root",
    password = "Admin@123"
)
cursor = connection.cursor()
try:
    cursor.execute("CREATE DATABASE IF NOT EXISTS CustomerDB")
    print("Database 'CustomerDB' created successfully (If not
          existed already).")
except mysql.connector.Error as err:
    print(f"Error: {err}")
connection.database = 'CustomerDB'
create_table_query = """
CREATE TABLE IF NOT EXISTS Customers (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50),
    email VARCHAR(100),
    age INT
)
"""
```

M-2 OUTPUT - Q.1

Database 'CustomerDB' created successfully (If not existed already).
Table 'Customers' created successfully (If not existed already).
Customer 'Tushar' add successfully.
Customer 'Sonali' add successfully.

Customer Records:
(1, 'Tushar', 'tushar@gmail.com', 30)
~~(2, 'Tushar',~~
(3, ~~'Sonali',~~ 'Tushar', 'tushar@gmail.com', 28)
(4, ~~'Tushar',~~ 'Sonali', 'Sonali@gmail.com', 34)
(5, 'Tushar', 'tushar@gmail.com', ~~24~~ 28)
(6, 'Sonali', 'sonali@gmail.com', 34)
(7, 'Tushar', 'tushar@gmail.com', 28)
(8, 'Sonali', 'sonali@gmail.com', 34)
(9,, 'Tushar', 'tushar@gmail.com', 28)
(10, 'Sonali', 'sonali@gmail.com', 34)
(11, 'Tushar', '~~tu~~shar@gmail.com', 28)
(12, 'Sonali', 'sonali@gmail.com', 34)
(13, 'Tushar', 'tushar@gmail.com', 28)
(14, 'Sonali', 'Sonali@gmail.com', 34)
Customer with ID 1 ~~upload~~ updated successfully.

Customer Records:
(1, 'Tushar', 'tushar@gmail.com', 30)
(3, 'Tushar', 'tushar@gmail.com', 28)
(4, 'Sonali', 'Sonali@gmail.com', 34)
(5, 'Tushar', 'tushar@gmail.com', 28)
(6, 'Sonali', 'sonali@gmail.com', 34)
(7, 'tushar', 'tushar@gmail.com', 28)
(8, 'sonali', 'sonali@gmail.com', 34)
(9, 'Fushar', 'tushar@gmail.com', 28)

```
cursor.execute(create_table_query)
print("Table 'customersDB' 'Customers' created successfully (If
      not existed already).")
def insert_customer(name, email, age):
    insert_query = "INSERT INTO Customers (name, email, age)
                    VALUES (%s, %s, %s)"
    data = (name, email, age)
    cursor.execute(insert_query, data)
    connection.commit()
    print(f"Customer '{name}' add successfully.")
insert_customer("Tushar", "tushar@gmail.com", 28)
insert_customer("Sonali", "sonali@gmail.com", 34)
def select_all_customers():
    select_query = "SELECT * FROM Customers"
    cursor.execute(select_query)
    rows = cursor.fetchall()
    print("\n Customer Records: ")
    for row in rows:
        print(row)
select_all_customers()
def update_customer_age(customer_id, new_age):
    update_query = "UPDATE Customers SET age = %s where id = %s"
    data = (new_age, customer_id)
    cursor.execute(update_query, data)
    connection.commit()
    print(f"Customer with ID {customer_id} updated
                                    successfully.")
```

OUTPUT - Q.1

(10, 'Sonali', 'Sonali@gmail.com', 34)
(11, 'Tushar', 'tushar@gmail.com', 28)
(12, 'Sonali', 'Sonali@gmail.com', 34)
(13, 'Tushar', 'tushar@gmail.com', 28)
(14, 'Sonali', 'Sonali@gmail.com', 34)
Customer with ID 2 deleted successfully.

Customer Records:
(1, 'Tushar', 'tushar@gmail.com', 30)
(3, 'Tushar', 'tushar@gmail.com', 28)
(4, 'Sonali', 'sonali@gmail.com', 34)
(5, 'Tushar', 'tushar@gmail.com', 28)
(6, 'Sonali', 'Sonali@gmail.com', 34)
(7, 'Tushar', 'tushar@gmail.com', 28)
(8, 'Sonali', 'Sonali@gmail.com', 34)
(9, 'Tushar', 'tushar@gmail.com', 28)
(10, 'Sonali', 'Sonali@gmail.com', 34)
(11, 'Tushar', 'tushar@gmail.com', 28)
(12, 'Sonali', 'Sonali@gmail.com', 34)
(13, 'Tushar', 'tushar@gmail.com', 28)
(14, 'Sonali', 'Sonali@gmail.com', 34)

```
update_customer_age (1,30)
select_all_customers ()
def delete_customer (customer_id):
    delete_query = "DELETE FROM Customers WHERE id=%"
    data = (customer_id,)
    delete cursor.execute (delete_query, data)
    connection.commit()
    print (f"Customer with ID {customer_id} delete
                                    successfully.")
delete_customer (2)

select_all_customers ()
cursor.close()
connection.close()
print ("\n Database connection closed.")
```

M-2 OUTPUT - Q.2

Enter the first number : 3

Enter the Second Number : 2

Result : 1.5

Q.2] Write a Python Program that takes two numbers as input from the user and performs division. Handle ZeroDivisionError if the user tries to divide by zero and ValueError if non-numeric input is provided.

```
try:
    num1 = float(input("Enter the first number: "))
    num2 = float(input("Enter the second number: "))

    result = num1 / num2
    print(f"Result: {result}")

except ZeroDivisionError:
    print("Error: Division by zero in not allowed.")

except ValueError:
    print("Error: Please enter valid number.")
```

M-2 OUTPUT- Q.3

Enter a positive number: 33

You entered a positive number: 33.04

Q.3] Create a custom exception called NegativeNumberError that is raised when a user enters a negative numbers. Write a program that prompts the user for a positive number and raise ~~Negative~~ NegativeNumberError if the input is negative.

```
class NegativeNumberError (Exception):
    pass

try:
    number = float (input ("Enter a positive number:"))

    if number < 0:
        raise NegativeNumberError ("Error: Negative number
                                    are not allowed:")
    print (f"You entered a positive number:", {number})

except NegativeNumberError as e:
    print (e)
except ValueError:
    print ("Error: Please enter a valid number.")
```

M-2 OUTPUT - Q.4

File Content

Hello, My name is Sanjay.

Q.4] Write a Python program that attempts to open and read a file named ~~data.textbd~~ data.txt. If the file doesn't exist or isn't readable, log an appropriate error message using Python's logging module.

```
import logging

logging.basicConfig(filename = "file_error.log",
                    level = logging.ERROR,
                    format = "%(asctime)s - %(levelname)s - %(message)s")
try:
    with open ('data.txt", "r") as file:
        content = file.read()
        print ("File Content")
        print (content)

except FileNotFoundError as e:
    print("Error: The file does not exist.")
    logging.error("FileNotFoundError : 'data.txt' not found.")

except IOError as e:
    print ("Error: Could not read the file.")
    logging.error("IO Error: Faild to read 'data.txt'.")
```

Q.5] Write a UDP client - server program in Python where the client sends a message to the server, and the server sends a response back.

CODE :-

SERVER CODE :-

```
import socket
server_socket = socket.socket(socket.AF_INET,
                              socket.SOCK_STREAM)
server_socket.bind(('localhost', 65432))
server_socket.listen()

print("Server is waiting for a connection ...")

conn, addr = server_socket.accept()
print(f"Connected by {addr}")

data = conn.recv(1024).decode()
print(f"Recieved from client : {data}")

conn.sendall("Message recieved".encode())

conn.close()
server_socket.close()
```

M-2

OUTPUT- Q.5

Server Terminal:

```
Server is waiting for a connection...
Connected by ('127.0.0.1', 39648)
Recieved from client: Hello, Server!
```

Client Terminal:

```
Recieved from server: Message recieved
```

CLIENT CODE :-

```
import socket

client_socket = socket.socket(socket.AF_INET,
                              socket.SOCK_STREAM)
client_socket.connect(('localhost', 65432))

client_socket.sendall("Hello, Server!".encode())

data = client_socket.recv(1024).decode()
print(f"Received from server: {data}")

client_socket.close()
```

Q.6] Write a UDP client-server program in Python where the client sends a message to the server, and the server sends a response back.

SERVER CODE :-

```
import socket

server_socket = socket.socket(socket.AF_INET,
                              socket.SOCK_DGRAM)
server_socket.bind(('localhost', 65431))

data, addr = server_socket.recvfrom(1024)
print(f"Received from client: {data.decode()}")

server_socket.sendto("Message received".encode(), addr)
server_socket.close()
```

M-2 OUTPUT- Q.6

SERVER TERMINAL -

```
Received from client: Hello, UDP server!
```

CLIENT TERMINAL -

```
Received from server: Message received
```

CLIENT CODE :-

```
import socket

client_socket = socket.socket(socket.AF_INET,
                              socket.SOCK_DGRAM)
client_socket.sendto("Hello, UDP Server!".encode(),
                     ('localhost', 65432))

data, _ = client_socket.recvfrom(1024)
print(f"Received from server: {data.decode()}")

client_socket.close()
```

M-2 OUTPUT-Q.7

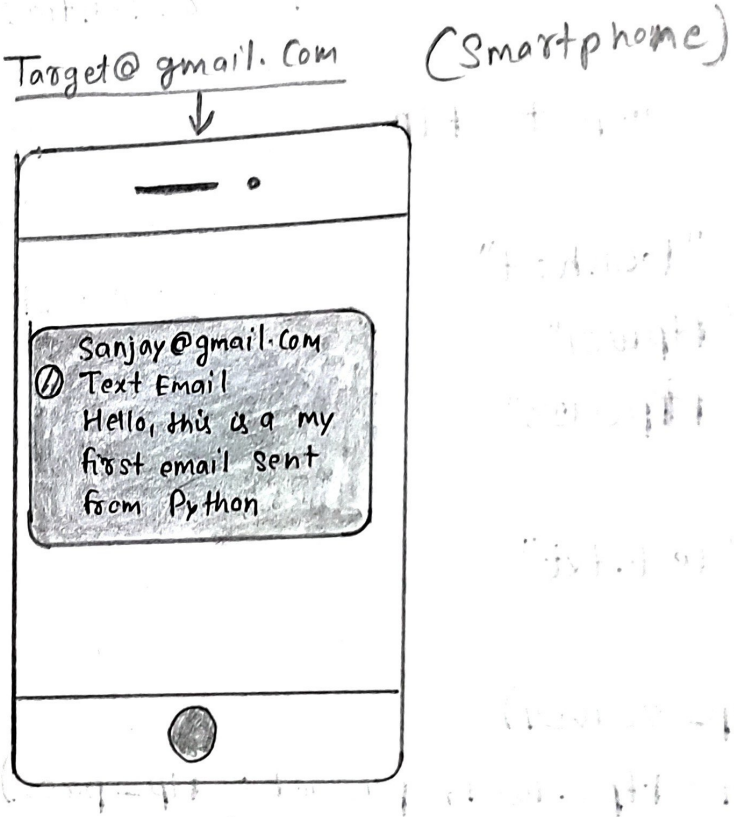
Connected to FTP server
text.txt uploaded successfully

Q.7] Write a Python script to upload a file to an FTP server. Assume you have the server credentials.

```
from ftplib import FTP

ftp_server = "localhost"
ftp_user = "ftpuser"
ftp_pass = "Ftp@123"

file_path = "test.txt"

ftp = FTP(ftp_server)
ftp.login(user=ftp_user, passwd=ftp_pass)
print("Connected to FTP server")

with open(file_path, 'rb') as file:
    ftp.storbinary(f"STOR {file_path}", file)
    print(f"{file_path} uploaded successfully")

ftp.quit()
```

M-2 OUTPUT - Q.8

Email Sent Successfully



8.8] Write a Python script to send a simple email using SMTP. Assume you have the sender's email and password.

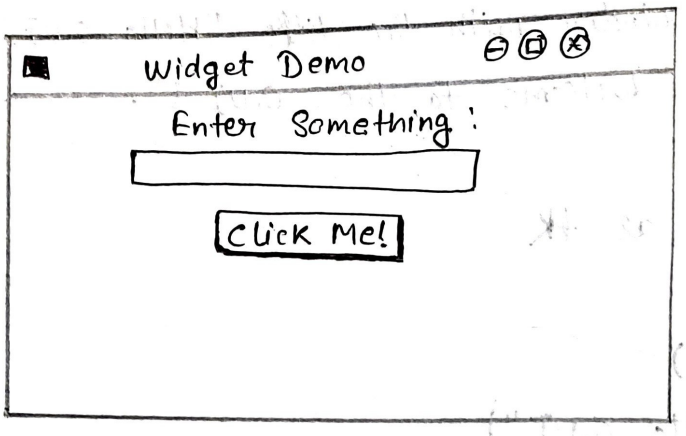
```
import smtplib

from email.mime.text import MIMEText

smtp_server = "smtp.gmail.com"
smtp_port = 587
sender_email = "sanjay@gmail.com"
sender_password = "Your App Password"
recipient_email = "target@gmail.com"

subject = "Text Email"
body = "Hello, this is my first text email sent from Python"

msg = MIMEText(body)
msg["Subject"] = subject
msg["From"] = sender_email
msg["To"] = recipient_email

with smtplib.SMTP(smtp_server, smtp_port) as server:
    server.starttls()
    server.login(sender_email, sender_password)
    server.sendmail(sender_email, recipient_email, msg.as_string())
    print("Email sent successfully")
```

M-2 OUTPUT - Q.9

Q.9] Create a simple GUI application in Python using tkinter that displays a window with the title "Hello GUI" and a label that says "Welcome to the GUI!".

```
import tkinter as tk

root = tk.Tk()
root.title("Hello GUI")
root.geometry("300x200")

label = tk.Label(root, text = "Welcome to the GUI")

root.mainloop()
```

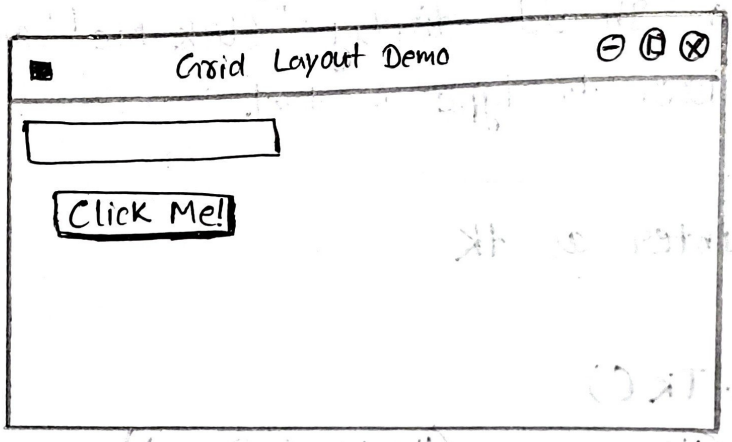
M-2 OUTPU- Q.10



Q.10] Add a Button and an Entry widget to the GUI from Question 9. The button should display "Click Me!" and the entry should allow the user to type in text.

```
import tkinter as tk

root = tk.Tk()
root.title("Widget Demo")
root.geometry("300x200")

label = tk.Label(root, text="Enter Something:")
label.pack()

entry = tk.Entry(root, width=20)
entry.pack()

button = tk.Button(root="Click Me!")
button.pack()

root.mainloop()
```

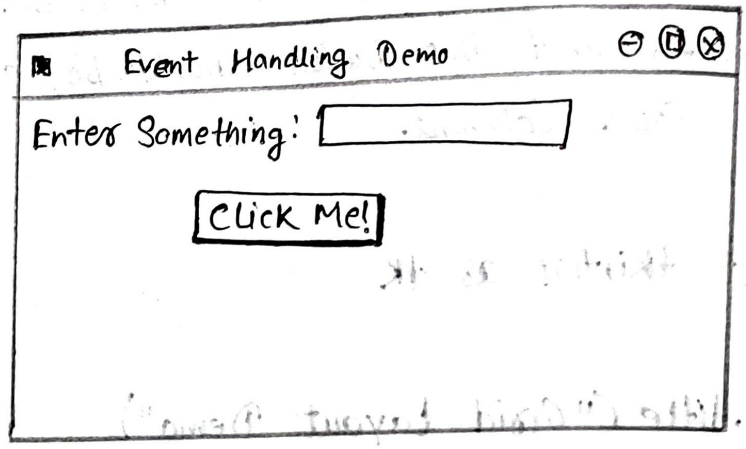
M-2 OUTPUT - Q.11



Q.11] Use a grid layout to arrange the widgets in the GUI from Question 10, The label should be in row 0, column 0, the entry widget in row 0, column 1, and the button should be centered in row 1, spanning both ~~four~~ columns.

```
import tkinter as tk

root.title("Grid Layout Demo")
root.geometry("300x200")

label = tk.Label(root, text="Enter Something:")
label.grid(row=0, column=0, padx=10, pady=10)

entry = tk.Entry(root, width=20)
entry.grid(row=0, column=0, padx=10, pady=10)

button = tk.Button(root, text="Click Me!")
button.grid(row=1, column=0, columnspan=2, pady=10)

root.mainloop()
```

M-2 OUTPUT - Q.12



Q.12] Add event handling to the button in the GUI from Question 11. When the button is clicked, it should display the text entered in the entry widget on a new label below the button.

```
import tkinter as tk

root = tk.Tk()
root.title("Event Handling Demo")
root.geometry("300x200")

def on_button_click():
    entered_txt = entry.get()
    result_label.config(text=f"You Entered : {entered_text}")

label = tk.Label(root, text="Enter Something: ")
label.grid(row=0, column=0, padx=10, pady=10)

entry = tk.Entry(root, width=20)
entry.grid(row=0, column=1, padx=10, pady=10)

button = tk.Button(root, text="Click Me!", command=on_button_
                                                    click)
button.grid(row=1, column=0, columnspan=2, pady=10)
result_label = tk.Label(root, text="")
~~result.grid~~ result_label.grid(row=2, column=0, columnspan=2)

root.mainloop()
```

M-2

OUTPUT - Q.13

Array :

[[1 2 3]
[4 5 6]
[7 8 9]]

Sum of all element : 45

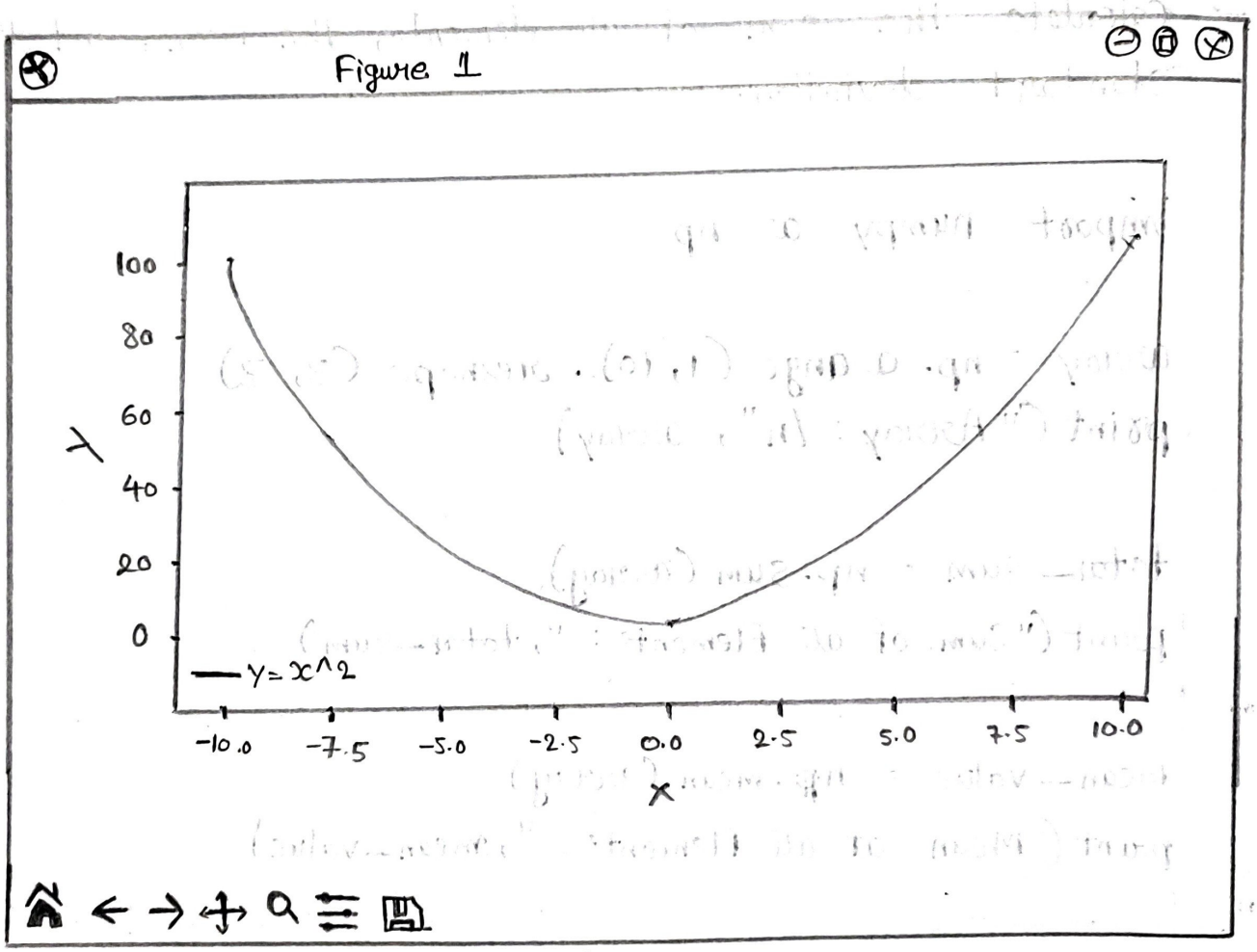
Mean of all Element : 5.0

Standard deviation : 2.581988897471611

8.13] Create a 3x3 NumPy array with values from 1 to 9. Then, calculate the sum of all elements, the mean, and the standard deviation.

```
import numpy as np

array = np.arange(1, 10).reshape(3, 3)
print("Array: \n", array)

total_sum = np.sum(array)
print("Sum of all Elements: ", total_sum)

mean_value = np.mean(array)
print("Mean of all Elements: ", mean_value)

std_dev = np.std(array)
print("standard deviation: ", std_dev)
```

M-2 OUTPUT - Q.14



Q.14] Create a simple line graph using Matplotlib to plot the function $y = x^2$ for x values from -10 to 10.

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

x = np.linspace(-10, 10, 100)
y = x**2

plt.plot(x, y, label='y = x^2', color='blue')
plt.title("Line Graph of y = x^2")
plt.xlabel("x")
plt.ylabel("y")

plt.legend()

plt.show()
```

M-2 OUTPUT - Q.15

Data Frame:

	Name	Age	City
0	Alice	24	New York
1	Bob	30	Los Angeles
2	Charlie	22	Chicago

Average Age: 25.333333333333332

8.15] Create a DataFrame with three columns: "Name", "Age", and "City". Add data for three people, then display the DataFrame and calculate the average age.

```
import pandas as pd

data = {
    "Name": ["Alice", "Bob", "Charlie"],
    "Age": [24, 30, 22],
    "City": ["New York", "Los Angeles", "Chicago"]
}

df = pd.DataFrame(data)
print("DataFrame :\n", df)

average_age = df["Age"].mean()
print("Average Age:", average_age)
```

M-2

OUTPUT - Q.16

Solution (x, y): [2.8 0.8]

Q.16] Use SciPy to solve the following system of linear equations:

$2x + 3y = 8$

$x + y = 2$

```
import numpy as np
from scipy.linalg import solve

A = np.array([[2, 3],
              [1, -1]])

B = np.array([8, 2])

solution = solve(A, B)
print("Solution (x, y):", solution)
```