

EXPERIMENT-11

Aim: GUI programming: Create a program that uses a graphical user interface (GUI) to allow the user to perform simple calculations.

Source Code:

```
import tkinter as tk
from tkinter import messagebox

1 usage
def calculate():
    try:
        num1 = float(entry_num1.get())
        num2 = float(entry_num2.get())
        operator = operator_var.get()

        if operator == '+':
            result = num1 + num2
        elif operator == '-':
            result = num1 - num2
        elif operator == '*':
            result = num1 * num2
        elif operator == '/':
            if num2 == 0:
                messagebox.showerror(title="Error", message="Division by zero is not allowed")
                return
            result = num1 / num2
        else:
            messagebox.showerror(title="Error", message="Invalid operator")
            return

        result_label.config(text="Result: " + str(result))
    except ValueError:
        messagebox.showerror(title="Error", message="Invalid input")
```

```
# Create the main window
root = tk.Tk()
root.title("Simple Calculator")

# Create input fields
label_num1 = tk.Label(root, text="Number 1:")
label_num1.grid(row=0, column=0)
entry_num1 = tk.Entry(root)
entry_num1.grid(row=0, column=1)

label_num2 = tk.Label(root, text="Number 2:")
label_num2.grid(row=1, column=0)
entry_num2 = tk.Entry(root)
entry_num2.grid(row=1, column=1)

# Create operator selection
operator_var = tk.StringVar()
operator_var.set('+')
operator_label = tk.Label(root, text="Operator:")
operator_label.grid(row=2, column=0)
operator_menu = tk.OptionMenu(root, operator_var, value='+', *values: '-', '*', '/')
operator_menu.grid(row=2, column=1)

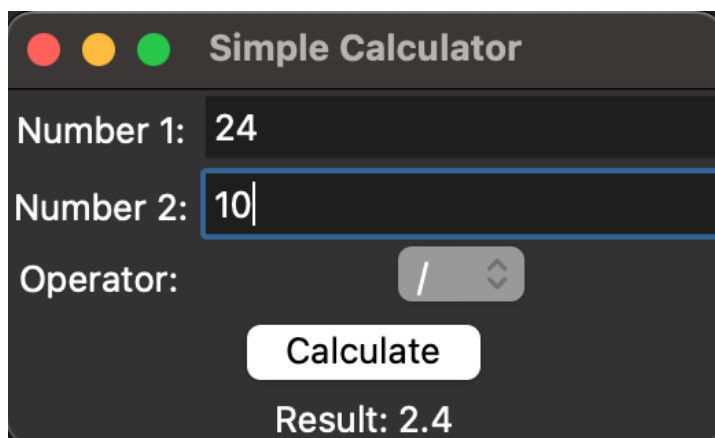
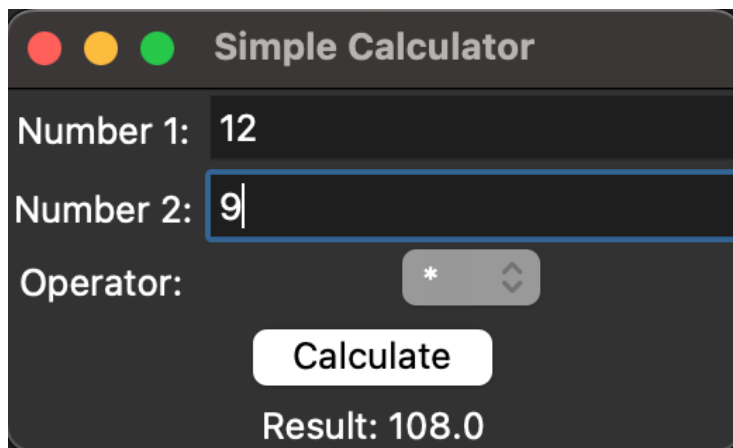
# Create calculate button
calculate_button = tk.Button(root, text="Calculate", command=calculate)
calculate_button.grid(row=3, column=0, columnspan=2)

# Create result label
result_label = tk.Label(root, text="")
```

```
result_label.grid(row=4, column=0, columnspan=2)

root.mainloop()
```

Output:



Viva Qs:

1.What programming language and library did you use to create this calculator?

A1: Python and the Tkinter library.

2. How does the program handle user input when they click on a number button?

A2: When a number button is clicked, the button_click function is called. This function takes the number as an argument and appends it to a global variable named expression. The display_var.set() method then updates the text on the display with the current value of expression.

3. Briefly explain how the program evaluates the expression when the equal button is pressed.

A3: Clicking the equal button calls the `button_equal` function. This function uses the `eval` function to evaluate the expression stored in the expression variable. However, it's important to note that `eval` can be a security risk if used carelessly. It's recommended to implement additional checks and validation to ensure user-entered expressions are safe before evaluation.

4. How does the program handle errors, such as division by zero or invalid expressions?

A4: Currently, the code uses a try-except block within `button_equal` to catch errors during evaluation. If an error occurs, the display is set to "Error" and the expression variable is cleared.

5. How would you enhance this calculator program?

A5: Here are some potential enhancements:

- Implement basic error handling and validation: Add checks to prevent syntax errors and division by zero before evaluation (consider using a parser or expression tree).
- Add memory functionality: Include buttons for storing and recalling values.
- Support scientific notation: Allow entering numbers in scientific notation (e.g., $2.5e-3$).
- Improve user interface: Enhance the layout, add clear labels, and consider using different colors or styles for buttons.

EXPERIMENT-12

Aim: Web scraping: Create a program that uses a web scraping library to extract data from a website and then stores it in a database.

Source Code:

```
import urllib.request
from bs4 import BeautifulSoup
import sqlite3
import ssl

# Function to scrape website and extract data
1 usage
def scrape_website(url):
    try:
        context = ssl._create_unverified_context()
        response = urllib.request.urlopen(url, context=context)
        html = response.read().decode('utf-8')
        soup = BeautifulSoup(html, features='html.parser')
        # Extract quotes and authors
        quotes = []
        for quote in soup.find_all(name='div', class_='quote'):
            text = quote.find('span', class_='text').text.strip()
            author = quote.find('small', class_='author').text.strip()
            quotes.append((text, author))
        return quotes
    except Exception as e:
        print("Error scraping website:", e)
        return []

# Function to create table in SQLite database
1 usage
def create_table():
    try:
        conn = sqlite3.connect('quotes.db')
```

```
def create_table():
    c = conn.cursor()
    # Create table if not exists
    c.execute('''CREATE TABLE IF NOT EXISTS quotes
                (id INTEGER PRIMARY KEY, text TEXT, author TEXT)''')

    conn.commit()
    conn.close()

except Exception as e:
    print("Error creating table in database:", e)

# Function to store data in SQLite database
! usage
def store_in_database(data):
    try:
        conn = sqlite3.connect('quotes.db')
        c = conn.cursor()
        # Insert data into the table
        c.executemany('INSERT INTO quotes (text, author) VALUES (?, ?)', data)
        conn.commit()
        conn.close()
    except Exception as e:
        print("Error storing data in database:", e)

# Main function
! usage
def main():
    url = 'http://quotes.toscrape.com/' # URL of the website to scrape
    data = scrape_website(url)
    create_table() # Ensure table exists before storing data
```

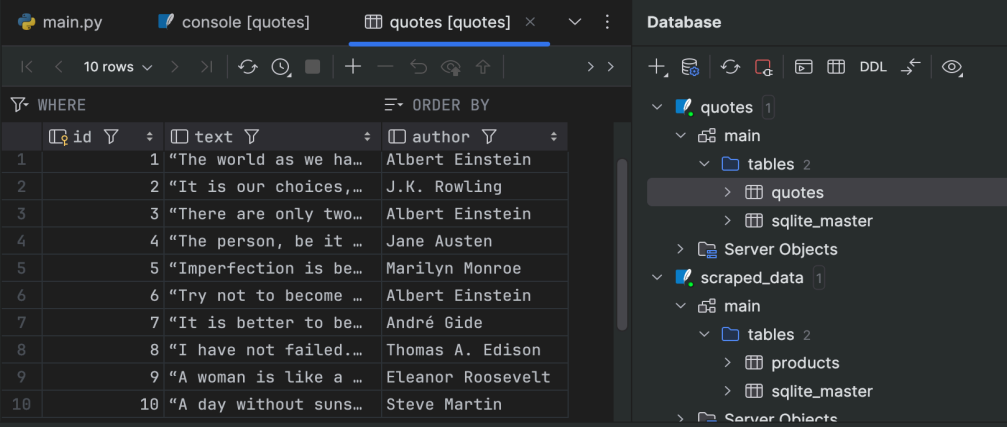
```
store_in_database(data)
print("Data scraped and stored successfully!")

if __name__ == "__main__":
    main()
```

Output:

```
/Users/anushkaagrani/movies-recommender-system/calculator/.venv/bin/python3 main.py
Data scraped and stored successfully!

Process finished with exit code 0
```



id	text	author
1	"The world as we ha...	Albert Einstein
2	"It is our choices,...	J.K. Rowling
3	"There are only two...	Albert Einstein
4	"The person, be it ...	Jane Austen
5	"Imperfection is be...	Marilyn Monroe
6	"Try not to become ...	Albert Einstein
7	"It is better to be...	André Gide
8	"I have not failed...	Thomas A. Edison
9	"A woman is like a ...	Eleanor Roosevelt
10	"A day without suns...	Steve Martin

Viva Qs:

Q1. What libraries are used in this program, and what do they do?

A1.

- requests: Sends HTTP requests to retrieve data from the web.
- BeautifulSoup: Parses HTML content, allowing us to extract specific information.
- sqlite3: Interacts with a SQLite database for data storage.

Q2. How does the program identify the data to extract?

A2. It uses a list named `data_to_extract` to specify the desired data points (e.g., product name, price). The code then searches for HTML elements with corresponding CSS classes (adjust as needed).

Q3. How does the program handle missing data on a product listing?

A3. The `scrape_product` function checks if the desired element exists. If not, it assigns `None` to that field in the `product_data` dictionary. You can modify this behavior to handle missing data differently (e.g., use default values).

Q4. What is the purpose of the CREATE TABLE statement?

A4. It creates a table named "products" in the database if it doesn't already exist. The table schema defines columns for product name (text) and price (real number).

Q5. How would you ensure the program respects the website's robots.txt and terms of service?

A5. It's crucial to check the website's robots.txt and terms of service to see if scraping is allowed and follow any specific guidelines. You can potentially add logic to check robots.txt and adjust scraping behavior accordingly (not shown in this example).

EXPERIMENT-13

Aim: Data visualisation: Create a program that reads data from a file and then creates a visualisation of that data using a data visualisation library.

Source Code:

```
import matplotlib.pyplot as plt
import csv

# Function to read data from a CSV file
1 usage
def read_data(filename):
    data = []
    with open(filename, 'r', newline='') as file:
        reader = csv.DictReader(file)
        for row in reader:
            data.append((row['Country'], int(row['Population'])))
    return data

# Function to create visualization
1 usage
def create_visualization(data):
    # Unpack data into x and y lists
    countries, populations = zip(*data)

    # Create a bar chart
    plt.figure(figsize=(10, 6)) # Adjust figure size as needed
    plt.bar(countries, populations, color='skyblue')
    plt.xlabel('Country')
    plt.ylabel('Population')
    plt.title('Population of Countries')
    plt.xticks(rotation=45, ha='right') # Rotate x-axis labels for better readability
    plt.tight_layout() # Adjust layout to prevent clipping of labels
```

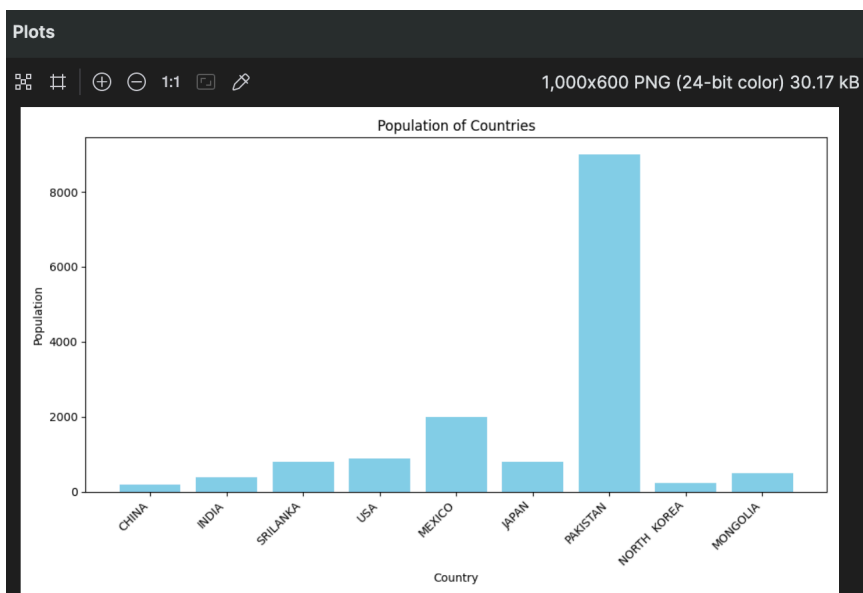
```
plt.show()

# Main function
1 usage
def main():
    filename = '/Users/anushkaagrani/Downloads/population_data.csv - Sheet1.csv'
    data = read_data(filename)
    create_visualization(data)

if __name__ == "__main__":
    main()
```

Output:

Country	Population
CHINA	200
INDIA	400
SRILANKA	800
USA	900
MEXICO	2000
JAPAN	800
PAKISTAN	9000
NORTH KOREA	230
MONGOLIA	503



Viva Qs:

1. What are some common data visualization libraries, and which one did you use in your program?

A1: Popular libraries include Matplotlib, Seaborn (built on top of Matplotlib), Plotly (for interactive visualizations), and Tableau (desktop application with a wider range of functionalities). The specific library used depends on the type of visualization and desired features.

2. How does your program handle different file formats (e.g., CSV, Excel)?

A2: The program can leverage libraries like pandas to read various file formats. Pandas provides functions for reading CSV, Excel (xlsx), and other common data formats.

3. How does your program identify the type of data (numerical, categorical) in the file?

A3: Libraries like pandas offer functions for data type detection (e.g., `pd.api.types.is_numeric_dtype`). You can also analyze data headers or sample values to infer data types.

4. How does your program choose the appropriate visualization type based on the data?

A4: The choice of visualization depends on the data structure and the insights you want to convey. Here are some examples:

- Numerical data: Bar charts for comparing categories, histograms for distribution, scatter plots for relationships between two variables.
- Categorical data: Pie charts for proportions, bar charts for comparing frequencies.

You can use the data types and the number of variables to guide your decision.

5. How does your program handle missing data in the file?

A5: Libraries like pandas provide functions for identifying missing values (e.g., `pd.isna`). You can choose to drop rows with missing data, impute them with an appropriate strategy, or visually represent them differently in the chart.

EXPERIMENT-14

Aim: Machine learning: Create a program that uses a machine learning library to classify images based on their content.

Source Code:

```
1  # importing the necessary libraries
2  import tensorflow as tf
3  import numpy as np
4  import matplotlib.pyplot as plt
5  # storing the dataset path
6  clothing_fashion_mnist = tf.keras.datasets.fashion_mnist
7
8  # loading the dataset from tensorflow
9  (x_train, y_train), (x_test, y_test) = clothing_fashion_mnist.load_data()
10
11 # displaying the shapes of training and testing dataset
12 print('Shape of training cloth images: ', x_train.shape)
13
14 print('Shape of training label: ', y_train.shape)
15
16 print('Shape of test cloth images: ', x_test.shape)
17
18 print('Shape of test labels: ', y_test.shape)
19
20 # storing the class names as it is
21 # not provided in the dataset
22 label_class_names = ['T-shirt/top', 'Trouser',
23                       'Pullover', 'Dress', 'Coat',
24                       'Sandal', 'Shirt', 'Sneaker',
25                       'Bag', 'Ankle boot']
26
```

```

27 # display the first images
28 plt.imshow(x_train[0])
29 plt.colorbar() # to display the colourbar
30 plt.show()
31
32 x_train = x_train / 255.0 # normalizing the training data
33 x_test = x_test / 255.0 # normalizing the testing data
34
35 plt.figure(figsize=(15, 5)) # figure size
36 i = 0
37 while i < 20:
38     plt.subplot(2, 10, i+1)
39
40     # showing each image with colourmap as binary
41     plt.imshow(x_train[i], cmap=plt.cm.binary)
42
43     # giving class labels
44     plt.xlabel(label_class_names[y_train[i]])
45     i = i+1
46
47 plt.show() # plotting the final output figure
48
49

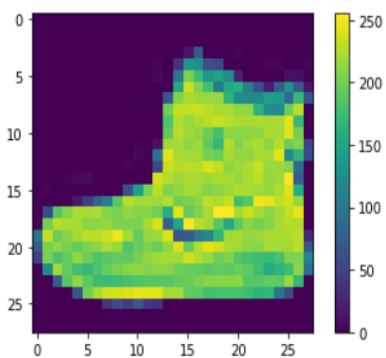
```

Output:

```

Shape of training cloth images: (60000, 28, 28)
Shape of training label: (60000,)
Shape of test cloth images: (10000, 28, 28)
Shape of test labels: (10000,)

```





Viva Qs:

1. What machine learning library did you use for image classification, and why?

A1: Popular libraries for image classification include TensorFlow, Keras (built on top of TensorFlow), PyTorch, and scikit-learn (with limitations for image data). The choice depends on factors like project complexity, ease of use, and desired functionalities.

2. Briefly explain the steps involved in your image classification program.

A2: The general steps typically involve:

- Data preparation: Loading images, resizing, normalization (converting pixel values to a common scale).
- Feature extraction: Using techniques like convolutional neural networks (CNNs) to automatically extract relevant features from the images.
- Model training: Splitting data into training and testing sets, training a machine learning model on the training data.
- Model evaluation: Evaluating the model's performance on the testing data using metrics like accuracy, precision, and recall.
- Prediction: Using the trained model to classify new, unseen images.

3. How does your program handle different image sizes and formats?

A3: The program usually resizes all images to a common size before feeding them into the model. Libraries like TensorFlow and PyTorch provide functions for image preprocessing, including resizing and normalization.

4. How does your program ensure the model doesn't overfit to the training data?

A4: Overfitting occurs when the model learns the training data too well but performs poorly on unseen data. Here are some techniques to prevent overfitting:

- Data augmentation: Artificially increasing the training data by applying random transformations (e.g., flipping, cropping) to existing images.
- Regularization techniques: Adding penalties to the model's cost function to discourage overly complex models that focus on memorizing training data instead of generalizing.
- Early stopping: Stopping the training process when the model's performance on the validation set starts to degrade.

5. How do you evaluate the performance of your image classification model?

A5: We can use metrics like accuracy (percentage of predictions correct), precision (ratio of true positives to predicted positives), and recall (ratio of true positives to all actual positives). You can also visualize the results using confusion matrices to understand how the model performs on different classes.

EXPERIMENT-15

Aim: Networking: Create a program that uses a networking library to communicate with a server and retrieve data from it.

Source Code:

```
import requests

# Function to communicate with the server and retrieve data
1 usage
def get_data_from_server(url):
    try:
        response = requests.get(url)
        if response.status_code == 200:
            return response.text
        else:
            print(f"Error: Failed to retrieve data from server (Status Code: {response.status_code})")
    except requests.exceptions.RequestException as e:
        print(f"Error: {e}")

# Main function
1 usage
def main():
    url = 'https://jsonplaceholder.typicode.com/posts' # Replace this with the actual URL of the server
    data = get_data_from_server(url)
    if data:
        print("Data from server:")
        print(data)

if __name__ == "__main__":
    main()
```

Output:

```
/Users/anushkaagrani/movies-recommender-system/calculator/.venv/bin/python /Users/anushkaagrani/movies-recommender-s
Data from server:
[
  {
    "userId": 1,
    "id": 1,
    "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
    "body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas
  },
  {
    "userId": 1,
    "id": 2,
    "title": "qui est esse",
    "body": "est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\nfugiat blanditii
  },
  {
    "userId": 1,
    "id": 3,
    "title": "ea molestias quasi exercitationem repellat qui ipsa sit aut",
    "body": "et iusto sed quo iure\nvoluptatem occaecati omnis eligendi aut ad\nvoluptatem doloribus vel accusantium
  },
  {
    "userId": 1,
    "id": 4,
    "title": "eum et est occaecati",
    "body": "ullam et saepe reiciendis voluptatem adipisci\nsit amet autem assumenda provident rerum culpa\nquis hic
  },
  {
    "userId": 10,
    "id": 97,
    "title": "quas fugiat ut perspiciatis vero provident",
    "body": "eum non blanditiis soluta porro quibusdam voluptas\nvel voluptatem qui placeat dolores qui velit aut\nv
  },
  {
    "userId": 10,
    "id": 98,
    "title": "laboriosam dolor voluptates",
    "body": "doloremque ex facilis sit sint culpa\nsoluta assumenda eligendi non ut eius\nsequi ducimus vel quasi\nv
  },
  {
    "userId": 10,
    "id": 99,
    "title": "temporibus sit alias delectus eligendi possimus magni",
    "body": "quo deleniti praesentium dicta non quod\naut est molestias\nmolestias et officia quis nihil\nitaque dol
  },
  {
    "userId": 10,
    "id": 100,
    "title": "at nam consequatur ea labore ea harum",
    "body": "cupiditate quo est a modi nesciunt soluta\nipsa voluptas error itaque dicta in\nautem qui minus magnam
  }
]

Process finished with exit code 0
```

Viva Qs:

1. What networking library did you use in your program, and why?

A1: Popular networking libraries include sockets (low-level, more control), requests (higher-level, easier to use), and asyncio (for asynchronous communication). The choice depends on the specific needs of the program, such as level of control, simplicity, or handling concurrent connections.

2. Explain the steps involved in your program's communication with the server.

A2: The general steps typically involve:

- Importing the networking library: Include the chosen library (e.g., `import socket`, `import requests`).
- Specifying server details: Define the server's address (IP or hostname) and port number.
- Establishing a connection: Use library functions to create a socket connection (sockets) or send an HTTP request (requests).
- Sending and receiving data: Depending on the protocol (TCP/UDP for sockets, HTTP for requests), send a request message (optional for TCP) and receive the server's response.
- Parsing the response: Interpret the received data according to the server's response format (e.g., JSON, text) using appropriate tools.
- Closing the connection: Terminate the communication using library functions (e.g., `socket.close()` or handling request context closure).

3. How does your program handle errors that might occur during communication (e.g., connection failure, invalid data format)?

A3: Robust error handling is crucial. The program can use exception handling mechanisms within the library (e.g., `try-except` blocks) to catch potential errors like connection timeouts, invalid server responses, or unexpected data formats. The program can then handle these errors gracefully, such as retrying the request, logging the error, or displaying an informative message to the user.

4. What type of data can your program retrieve from the server (e.g., text, binary)?

A4. The program can retrieve various data formats depending on the server's response. Common formats include:

- Text: Often used for human-readable data (e.g., JSON formatted information).
- Binary: Used for images, audio files, compressed data, or other non-textual content.

The program should handle the received data format appropriately based on the server's communication protocol and data type.

5. How does your program ensure the security of the communication with the server (e.g., encryption)?

A5: Security is a significant aspect. Here are some ways to improve security:

- Using secure protocols: If possible, choose protocols like HTTPS (encrypted HTTP) for data transmission, especially when dealing with sensitive information.
- Verifying server certificates: If using HTTPS, verify the server's SSL/TLS certificate to ensure authenticity and prevent man-in-the-middle attacks.
- Data validation: Validate the received data from the server to ensure its integrity and prevent potential attacks like data injection.