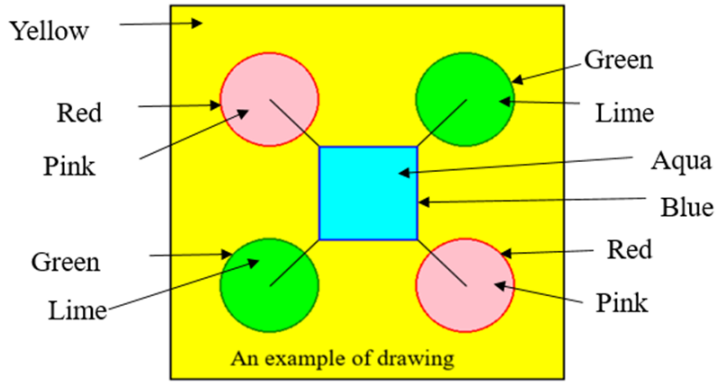
# Python -2 List of IPE Programs

| Sr. No. | Topic | Program Statement |
|---|---|---|
| 1 | Pandas | Use the file data.csv which contains 169 rows and 4 columns.<br>1. Convert this file into pandas Data Frame and Display basic statistics like mean, std, quartiles, etc. for this data frame.<br>2. Print first and last 5 rows. Also print the shape of the dataframe.<br>3. Create a correlation table for the data frame and comment about what kind of correlation is there between Duration and Calories?<br>4. Find whether there any null or NA values, drop all such rows if found in the data frame and print the shape of the data frame after dropping.<br>5. Prepare a scatter matrix for the following data frame.<br>6. Prepare a parallel coordinates for Duration v/s Pulse, Maxpulse and Calories (all 3 other columns).<br>7. Prepare a cross-tabulation for Duration v/s Pulse.<br>8. Do Maxpulse have any outliers? Find using function. |
| 2 | Area plots, box plots, scatter plots, heatmap, regression plot | import random<br><br># Sample data for area plot<br>years = [2010, 2011, 2012, 2013, 2014]<br>sales = [200, 300, 450, 350, 500]<br><br># Sample data for box plot<br>category1 = [random.randint(1, 50) for _ in range(50)]<br>category2 = [random.randint(25, 75) for _ in range(50)]<br>category3 = [random.randint(50, 100) for _ in range(50)]<br><br># Sample data for scatter plot<br>x = [random.uniform(0, 10) for _ in range(50)]<br>y = [random.uniform(0, 10) for _ in range(50)]<br><br># Sample data for heatmap<br>import numpy as np<br>data = np.random.rand(5, 5)<br><br># Sample data for regression plot<br>height = [160, 165, 170, 175, 180, 185]<br>weight = [60, 65, 70, 75, 80, 85]<br><br>Use the above code to generate sample data and then create the following:<br><br>1. Using the sample data for years and sales, create an area plot to visualize the trend in sales over the years. What insights can you gather from this area plot (answer as a comment)?<br><br>2. Utilizing the data in category1, category2, and category3, create a box plot using Matplotlib. How does the box plot reveal the distribution and potential outliers in these three categories (answer as a comment)?<br><br>3. Using the generated data for x and y, create a scatter plot with Matplotlib. What patterns or correlations, if any, can you observe between the x and y values in this scatter plot (answer as a comment)?<br><br>4. Employ the sample data to create a heatmap using Seaborn. What does the heatmap convey about the relationships between the values in the data matrix (answer as a comment)?<br><br>5. With the height and weight data, generate a regression plot using Seaborn. What conclusions can be drawn about the relationship between height and weight from this plot (answer as a comment)? |

# Python -2 List of IPE Programs

| Sr. No. | Topic | Program Statement |
|---|---|---|
| 3 | Regex/multithreading | 1. **Email Validator:**<br>Create a program that validates email addresses using regex. It should check if an input string is a valid email address according to common email address rules.<br>The regex pattern should be common one for basic email address validation. It checks for the following:<br><br>-Starts with one or more alphanumeric characters, dots, underscores, percentage signs, plus signs, or hyphens.<br>-Followed by the "@" symbol.<br>-Followed by one or more alphanumeric characters or hyphens.<br>-Followed by a dot (.) and at least two or more alphabetic characters.<br><br>2. **Password Strength Checker:**<br>Create a program that checks the strength of a password using regex. The program should ensure the password meets certain criteria, such as containing at least one uppercase letter, one lowercase letter, one digit, and one special character.<br><br>3. **Phone Number Extractor:**<br>Create a program that extracts phone numbers from a text using regex. It should find and display all valid phone numbers in the input text.<br>The regex pattern should account for various formats, including:<br><br>+91 1234567890<br>9876543210<br>080-12345678<br>+91-9876543210 |
| 4 | Regex/multithreading | **1. URL Extractor:**<br>Create a program that extracts URLs from a text using regex. It should find and display all valid URLs in the input text.<br>Example of URL : www.google.com<br><br>**2. IP Address Validator:**<br>Create a program that validates IP addresses using regex. It should check if an input string is a valid IPv4 address or not.<br>Example of IPv4 address :192.0.2.146<br><br>**3.HTML Tag Extractor:**<br>Create a program that extracts HTML tags from an HTML document using regex. It should find and display all HTML tags in the input text. |
| 5 | Web scraping | For the given fakepython html file, write a python program using BeautifulSoup library and perform following tasks -<br>1. Import beautifulsoup library. Attach the given html file<br>2. Scrape the given html and extract all Python related job titles and print them.<br>3. Extract all job titles, locations and companies and print them.<br>4. Create a pandas data frame with the details of python related job titles, locations and companies |
| 6 | Web scraping | For the given Quotes to Scrape.html file, write a python program using BeautifulSoup library and perform following tasks -<br>1. Import beautifulsoup library. Attach the given html file<br>2. Scrape the given html and extract all Quotes.<br>3. Extract all Quotes and authors and print them.<br>4. Create a pandas data frame with the details of Quotes and authors. |
| 7 | Linear Regression | Write a program to create a Model using linear regression to predict the charges of insurance using the csv file provided named “insurance.csv”. Do the required process in the data before making a model. Find predicted values, co-efficients, intercept and mean squared error. |
| 8 | Linear Regression | Consider variables x and y created from a pandas dataframe “car.csv” .<br>Create new column named “Age_car” (Age_car=2023-year)<br>For multiple linear regression problem, x contains the independent variables ( Age_car , Driven_kms , Fuel_Type , Selling_type , Transmission ) and y contains the dependent (Selling_Price) variable which is to be predicted.Write a Python program to spilt x and y into training and testing datasets with a 20% split. Then create a multiple linear regression model using the training data and print its coefficients ,intercept and mean squared error. |

# Python -2 List of IPE Programs

| Sr. No. | Topic | Program Statement |
|---|---|---|
| 9 | SimpleGUICS2Pygame | Write a python program to make following design in canvas using SimpleGUICS2Pygame module having size of canvas 400x400.<br>  |
| 10 | SimpleGUICS2Pygame | Write a Python Program using Memory Matching Game using the SimpleGUICS2Pygame library. The game involves flipping cards and matching pairs of cards with the same number. Below are instructions and explanations for the code:<br>(1)<br>Initializing global variables:<br>The new_game() function initializes several global variables used in the game. These variables are:<br>deck: A list representing the deck of cards, where each card contains a number from 0 to 7 (duplicated to have a pair of each number).<br>exposed: A list representing the state of each card. If exposed[i] is True, it means the card at index i is currently face-up, otherwise, it's face-down.<br>state: An integer representing the game state. It can take three values: 0, 1, or 2.<br>cIndex1 and cIndex2: Integer variables representing the indices of the two currently flipped cards.<br>(2)<br>Create new_game() function:<br>This function is called to start a new game or reset the current game. It initializes the variables and shuffles the deck to randomize the card positions.<br>(3) Event Handlers:<br>mouseclick(pos): This event handler is called whenever the player clicks on a card. The pos parameter represents the position of the mouse click. The function first determines which card was clicked (by dividing the x-coordinate of the click position by 50), and then it applies game logic based on the current state (state).<br>If the clicked card is face-down (not exposed), it behaves differently depending on the current state.<br>State 0: Just started - Flip the first card, update the state to 1, and mark the card as exposed.<br>State 1: One card flipped - Flip the second card, check for a match, update the state to 2.<br>State 2: Two cards flipped - If the two cards do not match, flip them back (mark them as not exposed). Then, flip the new card, update the state to 1, and mark it as exposed.<br>If the clicked card is already face-up (exposed), do nothing.<br>(4)<br>Creating the GUI:<br>A frame is created with the title "Memory Game" and a size of 800x100 pixels.<br>A "Reset" button is added to the frame.<br>Event handlers are registered to handle mouse clicks (mouseclick) and drawing (draw) on the canvas. |

# Python -2 List of IPE Programs

| Sr. No. | Topic | Program Statement |
|---|---|---|
| 11 | SimpleGUICS2Pygame | House Reveal Game Question:<br><br>You have been assigned the task of creating a House Reveal Game using the SimpleGUICS2Pygame library in Python. In this game, players can interactively reveal and hide different parts of a house by clicking on buttons. The house consists of three parts: a main body, a roof (triangle), and a circular window.<br>Instructions:<br>1. The canvas size should be set to 400x400 pixels.<br>2. The house is drawn on the canvas using geometric shapes. The main body of the house is a rectangle with a width of 100 pixels and a height of 100 pixels. The roof is a triangle that fits perfectly on top of the main body. The circular window has a radius of 15 pixels.<br>3. Define three boolean variables house_visible, roof_visible, and window_visible. These variables control the visibility of different parts of the house. When a part is visible, it will appear in the specified color when the corresponding button is clicked.<br>4. Insert draw_house() function. This function handles the drawing of the house on the canvas, including the main body, roof, and circular window.<br>5. The program should involve use of three buttons labeled "Reveal House (Red)", "Reveal Roof (Yellow)" and "Reveal Window (Blue)" to allow players to toggle the visibility and color of the corresponding house part.<br>6. When a button is clicked, the corresponding part of the house is revealed in the chosen color and also clicking the same button again will hide the corresponding part.<br>7. First, the main body should be revealed and only then the other parts should be revealed. |
| 12 | SimpleGUICS2Pygame | Write a program using SimpleGUICS2Pygame module of Python using the following instructions:<br><br>Create three buttons: circles, triangles, and squares.<br><br>Upon clicking the circles button, 10 circles (with different colours) should appear on the canvas at random positions. Every one second, their positions should randomly keep changing but they should remain within the canvas.<br><br>Upon clicking the triangles button, 10 triangles (with different colours) should appear on the canvas at random positions. Every one second, their positions should randomly keep changing but they should remain within the canvas.<br><br>Upon clicking the squares button, 10 squares (with different colours) should appear on the canvas at random positions. Every one second, their positions should randomly keep changing but they should remain within the canvas.<br><br>At any given time, if any of the shape's button is clicked again, then that shape should stop appearing on the canvas.<br>E.g., if the user has clicked circles and circles are visible on the canvas and the user clicks circles again, then circles should disappear from the canvas. So, if circles are not visible then clicking the button should make them visible and if circles are visible then clicking the button should make them disappear.<br><br>Displaying multiple shapes at the same time on the screen should also work.<br><br>E.g., if the user clicks circles and then clicks triangles, then both, circles and triangles should appear on the canvas. |

# Python -2 List of IPE Programs

| Sr. No. | Topic | Program Statement |
|---|---|---|
| 13 | SimpleGUICS2Pygame | Write a program to create Tic Tac Toe game using the SimpleGUICS2Pygame module in Python.<br><br>Game Instructions –<br><br>1. The game board consists of a 3x3 grid, and two players take turns to place their symbols ('X' or 'O') on the board until one player wins or the game ends in a draw.<br>2. The player who places three of their symbols in a horizontal, vertical, or diagonal line wins the game.<br>3. The completed game should display the Tic Tac Toe board, allow players to make moves by clicking on the board, and correctly display the winner on the canvas when the game is over.<br>4. The game should also have a "New Game" button to reset the board and start a new game.<br>5. Ensure the characters ('X' and 'O') are centered correctly within each cell of the game board.<br><br>Assessment Tasks –<br><br>1. Implement the draw_board function to display the Tic Tac Toe board, characters, and lines on the canvas.<br>2. Implement the mouseclick function to allow players to make moves when they click on an empty cell on the board.<br>3. Implement the check_winner function to check for a winning combination on the board after each move.<br>4. Display the winner's symbol ("X" or "O") on the canvas when the game is over.<br>5. Add functionality to the "New Game" button, so it resets the board and starts a new game when clicked.<br>6. Ensure the game board and characters are visually appealing and centered correctly. |
| 14 | SimpleGUICS2Pygame | Write a python program for shape shifting by using key down handler. Use SimpleGUICS2Pygame library.<br><br>height and width of the frame should be 200.<br><br>shapes = ["Square", "Circle", "Triangle"]<br>if user press d from keyboard then shape will change from left to right direction means shape will change from square to circle and circle to triangle.<br>if user press s from keyboard then shape will change from right to left direction means shape will change from triangle to circle and circle to square.<br>Draw shapes in the center of the frame with suitable dimension.<br><br>colors = ["DeepPink", "Red", "DarkOrange", "Yellow", "Lime", "Green", "Blue", "Aqua", "Purple", "Magenta"]<br>if user press v from keyboard then fill color will change from left to right direction.<br>if user press c from keyboard then fill color will change from right to left direction.<br><br>if user press x from keyboard then size of shapes will increase 10.<br>if user press z from keyboard then size of shapes will decrease 10.<br><br>if user press f from the keyboard then color should fill in the shapes and if user press again f from keyboard then fill color should remove from the shape. |

# Python -2 List of IPE Programs

| Sr. No. | Topic | Program Statement |
|---|---|---|
| 15 | Django Framework | **DJANGO SAMPLEAPP PROJECT**<br>**Task 1: Set Up the Project**<br>1. Task: Create a new Django project named "SampleApp."<br>2. Task: Create a Django app within the project named "sample"<br>3. Task: Ensure that Django is properly installed and the project can run without errors using the development server.<br><br>**Task 2: Define URL Patterns**<br>1. Task: Define three URL patterns in the "sample" app's `urls.py` file:<br>- '/' should route to the 'home' view.<br>- '/about/' should route to the 'about' view.<br>- '/contact/' should route to the 'contact' view.<br>2. Task: Ensure that each URL pattern is named 'home,' 'about,' and 'contact' respectively.<br><br>**Task 3: Create Views**<br>1. Task: Create three views in the "sample" app's `views.py` file:<br>- 'home' view should render the 'sample/home.html' template.<br>- 'about' view should render the 'sample/about.html' template.<br>- 'contact' view should render the 'sample/contact.html' template.<br><br>**Task 4: Create Templates**<br>1. Task: Create HTML templates for the 'home,' 'about,' and 'contact' views in the 'templates/sample' directory.<br>2. Task: The 'home.html' template should display a welcoming message.<br>3. Task: The 'about.html' template should contain information about the project or organization.<br>4. Task: The 'contact.html' template should provide contact information.<br><br>**Task 5: Configure Project Settings**<br>- Register the app in the project settings.<br><br>**Task 6: Configure Project URLs**<br>1. Task: Include the 'sample' app's URLs in the project's 'urls.py' file.<br>2. Task: Create a URL pattern that routes the root URL ('/') to the 'home' view.<br><br>**Task 7: Run Migrations and Start the Development Server**<br>1. Task: Run migrations to create the necessary database tables.<br>2. Task: Start the development server and ensure that the project is accessible in a web browser.<br>3. Task: Verify that the 'home,' 'about,' and 'contact' pages are accessible at the expected URLs.<br><br>**Task 8: Additional Functionality**<br>1. Task: Implement additional functionality, such as creating a '404 Not Found' page and linking it to an invalid URL.<br>2. Task: Add a navigation menu or links to navigate between the 'home,' 'about,' and 'contact' pages. |

# Python -2 List of IPE Programs

| Sr. No. | Topic | Program Statement |
|---|---|---|
| 16 | Django Framework | **Task 1: Project Setup**<br>a. Create a new Django project named "Bookstore."<br>b. Set up a Django app named "books."<br>**Task 2: Database and Models**<br>a. Define a Django model named "Book" with the following fields:<br>• Title (CharField)<br>• Author (CharField)<br>• Published Date (DateField)<br>• Price (DecimalField)<br>• ISBN (CharField)<br>b. Create and apply the necessary database migrations to create the "Book" model.<br>**Task 3: Admin Panel**<br>a. Register the "Book" model in the Django admin panel.<br>b. Create a superuser account with the username **admin** and a password **Ijiet123**.<br>c. Use the admin panel to add at least three sample books with different details.<br>**Task 4: Views and Templates**<br>a. Create a view to display a list of all books in the database. Use a template to render this list.<br>b. Create a view to display detailed information about a single book, including all its fields.<br>c. Create templates for both views, ensuring they have appropriate HTML structure.<br>**Task 5: URL Routing**<br>a. Define URL patterns to route requests to the views you created in Task 4.<br>b. Implement a homepage that displays a list of all books.<br>c. Implement URLs for displaying detailed book information.<br>**Task 6: Search Task**<br>a. Implement a search functionality that allows users to search for books by title or author. |
| 17 | Django Framework | • Create python Django project with name 'moviereview'<br>• Create an app called movie<br>• Create home.html file in movieapp.<br>• Code for home.html<br><body><br><h1>My movie app </h1><br><h3>Enter data </h3><br><form action="" ><br><label for="data">Data:</label><br><input type="text" name=" " ><br><br><br><button type="submit" >Search</button><br></form><br></body><br>• Create model named Movie with attributes Title,Actor,Date of Relaease.<br>• Create super user with your enrollment number and password will be your name.(it is compulsory)<br>• Log in to the django admin portal with this user and Enter the following data in Movie table.<br>Title Actor Date of Release<br>JAWAN SRK 8-Sept-2023<br>GADAR-2 SunnyD 25-Aug-2023<br>OH MY GOD-2 Akshay K 18-Aug-2023<br><br>• Make necessary adjustment to your code to let user search for data from this database by Title on home page. |

# Python -2 List of IPE Programs

| Sr. No. | Topic | Program Statement |
|---|---|---|
| 18 | Django Framework | • Create python Django project with name 'myproject'<br>• Create an app called myapp<br>• Create home.html file in myapp.<br>• Code for home.html<br><body><br><h1>My app</h1><br><h3>Enter data </h3><br><form action="" ><br><label for="data">Data:</label><br><input type="text" name=" " ><br><br><br><button type="submit" >Search</button><br></form><br></body><br>• Create model named Mydata with attributes name,branch,roll no.<br>• Create super user with your enrollment number and password will be your name.(it is compulsory)<br>• Log in to the django admin portal with this user and Enter the following data in Mydata table.<br>name branch roll no<br>Yaksh CE 111<br>Rohan IT 222<br>Radha CST 333<br><br>• Make necessary adjustment to your code to let user search for data from this database by name on home page. |
| 19 | Django Framework | 1. Create a Django Project named "music"<br>2. Create an App named 'song'<br>3. Create Home Page by making template home.html in 'song' App.<br>4. Code for 'home.html' is as below.<br><br><body><br><h2> Songs </h2><br><h4> Enter Song Name </h4><br><form action="" ><br><label for="search">Search for Song </label><br><input type="text" name="SearchSong" /><br><button type="submit" >Search</button><br></form><br></body><br><br>5. Create Model with name 'Song' with attributes songname, singers, musicdirector, year.<br>6. Create a superuser and using the username and password, enter the details of Song from admin panel. Details are given as below.<br>7. Make Necessary changes to your code to show all the songs on home page ordered by year.<br>8. Search the particular song using search box should show the particular searched song details after clicking search button. |

# Python -2 List of IPE Programs

| Sr. No. | Topic | Program Statement |
|---|---|---|
| 20 | Django Framework | DJANGO TEMPLATE ENGINE PROJECT<br>Task 1: Project Setup and Template Configuration<br><br>1. Task: Verify project setup and template configuration.<br>- Description: Confirm that the Django project and app have been created, and that template settings in `settings.py` are correctly configured.<br><br>Task 2: Create a Basic Template<br><br>2. Task: Create a basic HTML template.<br>- Description: Develop a simple HTML template named `hello.html` inside the app's `templates` directory, as shown in the project setup.<br><br>Task 3: Create a View to Render the Template<br><br>3. Task: Develop a view to render the template.<br>- Description: Create a view function named `hello_view` in the app's `views.py` that renders the `hello.html` template.<br><br>Task 4: Define a URL Pattern for the View<br><br>4. Task: Define a URL pattern for the `hello_view` in the app's `urls.py`.<br>- Description: Create a URL pattern that maps to the `hello_view` function, making sure it includes the `/demo/hello/` URL path.<br><br>Task 5: Configure Main URLs<br><br>5. Task: Verify main URL configuration.<br>- Description: Confirm that the app's URLs are included in the main project's `urls.py` correctly.<br><br>Task 6: Start the Development Server<br><br>6. Task: Run the development server.<br>- Description: Start the Django development server using the command `python manage.py runserver`. Verify that the server runs without errors.<br><br>Task 7: Access the Template via URL<br><br>7. Task: Access the template via its URL.<br>- Description: Access the template at `http://localhost:8000/demo/hello/` using a web browser or a tool like `curl`. Ensure that the template is displayed as expected, showing "Hello, Django User!"<br><br>Task 8: Modify the Template Context<br><br>8. Task: Modify the template context.<br>- Description: In the `hello_view`, change the value of the `name` variable in the context to a different name (e.g., "John"). Verify that the template updates accordingly.<br><br>Task 9: Template Inheritance<br><br>9. Task: Implement template inheritance .<br>- Description: Create a base template that includes common elements like headers and footers. Then, create a child template that extends the base template and adds content unique to the child template.<br><br>Task 10: Template Tags<br><br>10. Task: Explore and use additional template tags<br>- Description: Experiment with Django's template tags (e.g., `for`, `if`, `include`) to enhance the template's functionality or appearance. |