PSEUDOCODE

```
Global Variables:

    •   Selected Numbers: Array to store user-selected lotto numbers (1-52)
    •   Tickets: Array to store ticket objects


Function: init

 1. Display Functions:

   - Display functions to create user interface elements for:

      - Admin and User options

      - Number selection (1-52) with option to pick multiple

      - Number of boards input field

      - Generate ticket button

      - Ticket details display area (ID, date, boards, price)

      - Notification area for alerts and wins

2. Event Listeners:

   - Add event listener for number selection to update selectedNumbers

   - Add event listener for generate ticket button to call handleTicketGeneration


Function: handleNumberSelection (event)

 1. Get all selected options from the number selection element

2. Loop through selected options:

   - Extract the value (number) and convert to integer

   - Add the number to selectedNumbers array


Function: handleTicketGeneration

1. Get the number of boards from the input field and convert to integer
```

```
2. Validate input (e.g., check if number of boards is within limits)
3. Call createTicket function with selectedNumbers and number of boards
4. Add the returned ticket object to the tickets array
5. Call displayTicketDetails function to show details of the generated ticket
```

```
Function: createTicket (selectedNumbers, numBoards)
 1. Generate a unique ticket ID (call separate function)
2. Get current date using getDate function
3. Create an empty array to store boards
4. Loop numBoards times:
   - Create a new board object with a copy of selectedNumbers for its numbers
   - Add the board object to the boards array
5. Calculate total ticket price based on number of boards and potential Lotto Plus entries (call separate function)
6. Create a new ticket object with:
   - ID: generated unique ID
   - Date: retrieved date
   - Boards: array of board objects
   - Total Price: calculated price
7. Return the newly created ticket object
```

```
 Function: displayTicketDetails (ticket)
 1. Access and display ticket details:
   - ID: ticket.ID
   - Date: ticket.date
   - Boards: Loop through boards array and display numbers for each board
```

- Total Price: ticket.totalPrice

Additional Functions:

generateTicketId: Implement logic to generate a unique identifier for each ticket.

calculateTicketPrice: Calculate the total price based on the number of boards and any selected Lotto Plus options.

Notifications:

Implement logic to display confirmation messages before generating tickets and notifications for wins after simulating a draw.

Store winning numbers and draw history after simulation (consider using localStorage).

Persistence:

Use localStorage to store user-selected numbers for persistence across sessions.

HUMAN READABLE ALGORITHM FOR LOTTO APP

1. User Interface Setup

- Display separate sections for Admin and User roles.
- In the User section:
- Show a number selection area with options for all numbers (1-52). Users can pick multiple numbers.
- Provide an input field for users to specify the desired number of boards for their ticket with limitations,
- Include a "Generate Ticket" button.
- Have a designated area to display details of generated tickets (ID, date, boards with chosen numbers, total price).
- Consider adding an area for notifications (purchase confirmation, winning alerts).

- Admin section can have additional functionalities.

2. User Interaction

- When a user selects numbers, store their choices in a list.
- When the user enters the number of boards and clicks "Generate Ticket":
- Validate the number of boards entered (within allowed limits).
- Call a function to create a new ticket object.

3. Creating a Ticket

- Generate a unique identifier for the ticket.
- Capture the current date.
- Based on the number of boards chosen:
- Create an empty list to store individual boards.
- Loop through the desired number of boards:
- Copy the user-selected numbers into a new board object.
- Add this board object to the list of boards.
- Calculate the total price based on the number of boards and any additional options like Lotto Plus (if applicable).
- Create a new ticket object with properties like:
- ID (the unique identifier generated earlier)
- Date (the captured current date)
- Boards (the list of board objects)
- Total Price (the calculated total cost)
- Add the newly created ticket object to a list of all tickets.
- Call a function to display the details of the generated ticket for the user.

4. Displaying Ticket Details

- Access and display the following information from the ticket object:
- Ticket ID

- Date of purchase
- Boards: Loop through each board and display the chosen numbers for that board.
- Total ticket price

5. Additional Considerations

- Implement logic to display confirmation messages before generating a ticket and notifications for winning tickets after a simulated draw.
- Consider using localStorage to store user-selected numbers for persistence across sessions.
- Include error handling for user input (e.g., invalid number of boards).

6. Admin Functionality

- Develop separate functionalities for the Admin section, such as:
- Simulating a Lotto draw by randomly selecting winning numbers.
- Identifying winning tickets based on the drawn numbers and user selections.
- Managing the history of draws and winning tickets (potentially using databases).

7. Future Enhancements

- Integrate functionalities for Lotto Plus entries (selecting numbers and calculating additional cost).
- Implement user account login for managing tickets and purchase history.
- Improve user experience with visual elements and interactive features.