## Algorithm for Lottery Application

1. **Initialization:**
    - Initialize system variables, interfaces, and data structures.
2. **User Interface:**
    - Provide options to switch between admin and user modes.
    - Display interface for user to select 6 Lotto numbers from 1 to 52.
    - Allow user to choose number of boards (up to 10) and select numbers for each board.
    - Calculate and display total ticket cost based on selections.
    - Option to select same numbers for Lotto Plus 1 and Lotto Plus 2
3. **Ticket Generation:**
    - For each ticket:
        - Generate a unique ticket identifier.
        - Assign selected boards (up to 10 per ticket).
        - Calculate total price for the ticket based on:
            - R5.00 per Lotto board.
            - R2.50 per Lotto Plus 1 board.
            - R2.50 per Lotto Plus 2 board.
        - Store ticket information including selected numbers and associated costs.

4. **Draw Simulation (Admin):**

   - Admin mode:
     - Simulate a draw to generate random winning numbers.
     - Compare drawn numbers with user tickets to determine winning tickets based on matching criteria (e.g., 3 balls match and higher).
     - Record draw date, drawn numbers, and winning tickets.

5. **Notifications and Alerts:**

   - Notify users if their tickets have won anything after each draw.
   - Alert admin about total winning tickets and draw results.

6. **Historical Data Management:**

   - Maintain a database or file storage for:
     - All entered tickets (for historical reference).
     - Winning tickets and corresponding prizes.
     - Drawn numbers and dates.

## Pseudocode  Lottery Application

### Initialize data structures

```
 Initialize empty lists for tickets, winning_tickets
 Initialize variables for total_cost, draw_results
```

### Function to display user interface

```
function display_user_interface():
    Display options to switch between admin and user modes
    Prompt user to select 6 unique Lotto numbers from 1 to 52
    Prompt user to select number of boards (1 to 10) for their ticket
    For each board:
        Prompt user to select numbers ensuring ball colors match specified ranges
    Calculate total ticket cost based on selected boards:
        total_cost = 0
        For each board selected:
            if board is Lotto:
                total_cost += 5.00
            else if board is Lotto Plus 1 or Lotto Plus 2:
                total_cost += 2.50
    Display total_cost to user
    Option to select same numbers for Lotto Plus 1 and Lotto Plus 2
```

**Function to generate tickets**

```
function generate_tickets():
    For each ticket:
        Generate a unique ticket_id
        Initialize empty list boards for the ticket
        For each selected board:
            Add selected numbers to boards list
        Calculate total price for the ticket:
            ticket_price = 0
            For each board in boards:
                if board is Lotto:
                    ticket_price += 5.00
                else if board is Lotto Plus 1 or Lotto Plus 2:
                    ticket_price += 2.50
        Add ticket_id, boards, and ticket_price to tickets list
```

**Function for admin to simulate draw**

```
function simulate_draw():
    Generate random winning_numbers (6 unique numbers from 1 to 52)
    For each ticket in tickets:
        match_count = count_matching_numbers(ticket.boards, winning_numbers)
        if match_count >= 3:
            Add ticket_id and match_count to winning_tickets list
    Record draw_results with date, winning_numbers, and winning_tickets
```

**Function to notify users and admin about draw results**

```
function notify_results():
    For each ticket in winning_tickets:
        Notify user with ticket_id that they have won
    Notify admin with total count of winning_tickets and draw_results
```

**Function to manage historical data**

```
function manage_historical_data():
    Store all tickets in database for historical reference
    Store winning tickets and corresponding prizes
    Store draw_results including date and drawn numbers
```

**Main Program Execution**

```
if __name__ == "__main__":
    # Display user interface and gather user selections
    display_user_interface()

    # Generate tickets based on user selections
    generate_tickets()

    # Admin mode: Simulate draw and determine winning tickets
    simulate_draw()

    # Notify users and admin about draw results
    notify_results()

    # Manage and store historical data
    manage_historical_data()
```