An algorithm for the **CameraRentalApp**:

1. Start the **CameraRentalApp**.
2. Initialize the **cameraList**, **currentUser**, **walletBalance**, and **scanner** variables.
3. Call the **initializeCameras** function.
4. Call the **login** function.
5. If the **currentUser** is not null, proceed to the next step. Otherwise, exit the application.
6. Call the **showMainMenu** function.
7. Display the main menu options to the user.
8. Prompt the user to enter a choice.
9. If the choice is 1, call the **listCameras** function.
10. If the choice is 2, call the **rentCamera** function.
11. If the choice is 3, call the **walletMenu** function.
12. If the choice is 4, display an exit message and end the application.
13. If the choice is invalid, display an error message and go back to step 7.
14. Go back to step 7 and repeat until the user chooses to exit the application.

Functions:

1. **initializeCameras**:
    - Display a welcome message and prompt the user to enter the number of cameras to add.
    - Iterate over the number of cameras:
        - Prompt the user to enter the camera ID, brand, model, and per-day rental amount.
        - Create a **Camera** object with the provided details and add it to the **cameraList**.
    - Sort the **cameraList** based on camera ID.
2. **login**:
    - Display a login message and prompt the user to enter a username and password.
    - Check if the entered credentials match the predefined username and password.
    - If the credentials match, set the **currentUser** and display a login success message. Otherwise, display an invalid credentials message.
3. **showMainMenu**:
    - Display the main menu options to the user.

- Prompt the user to enter a choice.
- Implement a loop to repeatedly display the menu and process the user's choice until the user chooses to exit.
- Handle each menu option accordingly.

4. **listCameras**:
    - Check if the **cameraList** is empty.
    - If the **cameraList** is not empty, display the camera details in a formatted table.
5. **rentCamera**:
    - Check if the **cameraList** is empty or if the wallet balance is insufficient.
    - Prompt the user to enter a camera ID to rent.
    - Search for the camera with the matching ID in the **cameraList**.
    - If the camera is found and not already rented, set its rented status to true and deduct the rent amount from the wallet balance. Display a success message.
    - If the camera is already rented or the ID is invalid, display an appropriate message.
6. **walletMenu**:
    - Display the wallet menu options to the user.
    - Prompt the user to enter a choice.
    - Implement a loop to repeatedly display the menu and process the user's choice until the user chooses to go back.
    - Handle each menu option accordingly.