### **Scenario Overview: Contact Management System**

You have been hired by a company to develop a **Contact Management System**. The purpose of the system is to help the company manage the contact details of users in an organized way. The system will allow the company to efficiently store, update, delete, and display contact details such as first name, last name, and phone numbers.

### **System Functionalities:**

The Contact Management System should include the following capabilities:

1. **Add Contact:**
   1. Allow users to add a new contact with details such as first name, last name, and phone number.
2. **Edit Contact:**
   1. Allow users to search for an existing contact by phone number and edit their details.
3. **Delete Contact:**
   1. Allow users to search for a contact by phone number and delete it from the system.
4. **Display All Contacts:**
   1. Display a list of all saved contacts, showing their full names and phone numbers in a readable format.

### **System Requirements:**

1. **User Class:**
   1. The User class should store all necessary properties, which include the user's **first name**, **last name**, and **phone number**.
   2. A method should be provided to display the contact's full details (first name, last name, and phone number) in a formatted manner.
2. **Contact Class:**
   1. The Contact class will store a list of users and manage the collection of contacts. It will have methods for:
      1. Adding new contacts.
      2. Editing existing contacts by searching with a phone number.
      3. Deleting contacts by searching with a phone number.
      4. Displaying all contacts in a readable format.
3. **User Interaction:**
   1. The system will be controlled via a console-based interface that presents users with a menu to interact with the contact management features.

### **Task Walkthrough and Expected Behavior**

1. **Adding Contacts:**
   1. **User Action:** The user selects the option to add a new contact from the menu.
   2. **System Action:** The system will prompt the user to enter the first name, last name, and phone number of the new contact. Upon entering the details, the system adds the contact to the list and confirms the addition with a success message.
   3. **Expected Outcome:** The contact is successfully added to the system, and the user is notified.
2. **Editing Contacts:**
   1. **User Action:** The user selects the option to edit an existing contact.
   2. **System Action:** The system prompts the user to enter the phone number of the contact they wish to edit. If the contact exists, the user will be asked to provide new details (first name, last name, phone number). If the contact is not found, an error message is displayed.
   3. **Expected Outcome:** If the contact is found, the details are updated; if not, an error message is displayed.
3. **Deleting Contacts:**
   1. **User Action:** The user selects the option to delete a contact.
   2. **System Action:** The system prompts the user to enter the phone number of the contact to be deleted. If the contact exists, it is removed from the list. If not, an error message is displayed.
   3. **Expected Outcome:** The contact is successfully deleted, or an error message is shown if the contact is not found.
4. **Displaying All Contacts:**
   1. **User Action:** The user selects the option to display all contacts.
   2. **System Action:** The system will iterate over the list of contacts and display each contact’s full name (first and last name) along with their phone number.
   3. **Expected Outcome:** The user sees a formatted list of all the contacts currently stored in the system.
5. **Exiting the Program:**
   1. **User Action:** The user selects the option to exit the program.
   2. **System Action:** The system will close the application or stop the loop, ending the program.
   3. **Expected Outcome:** The program exits gracefully.

### **User Interaction Flow:**

1. **Menu Display:**
   1. The system continuously displays a menu with the following options:
      1. **Option 1:** Add Contact
      2. **Option 2:** Edit Contact
      3. **Option 3:** Delete Contact
      4. **Option 4:** Show All Contacts
      5. **Option 5:** Exit
2. **Input Handling:**
   1. For each menu option, the system will prompt the user to input the required details (such as first name, last name, or phone number) and perform the relevant operation.
   2. If the input is invalid (e.g., a phone number that doesn't exist), the system will provide feedback and allow the user to try again.
3. **Error Handling:**
   1. When trying to edit or delete a contact by phone number, if the contact is not found, the system will display an appropriate error message.
   2. If the user provides incorrect input (e.g., entering a non-numeric value for the phone number), the system will request the correct format.

### **Expected User Behavior and System Response:**

1. **Adding a Contact:**
   1. The user enters a valid first name, last name, and phone number.
   2. The system adds the contact and confirms with a success message.
2. **Editing a Contact:**
   1. The user enters an existing phone number.
   2. The system displays the current contact details and prompts for updated information. After submission, the system updates the contact details and confirms the update.
3. **Deleting a Contact:**
   1. The user enters a valid phone number.
   2. The system removes the contact and displays a confirmation message.
   3. If the phone number does not match any contact, an error message is shown.
4. **Displaying All Contacts:**
   1. The user requests to see all contacts.
   2. The system lists all contacts with their first name, last name, and phone number.
5. **Exiting the Program:**
   1. The user selects the exit option.
   2. The system gracefully exits the program.

# Expected Output:

