Digital Empowerment Pakistan

Internship Batch 3 DEN

Domain:
C++ programming

task :02

Submitted By:

Laiba Husna

Gmail:

Laibahusna163@gmail.com

Task2:

Creating a Contact Management System:

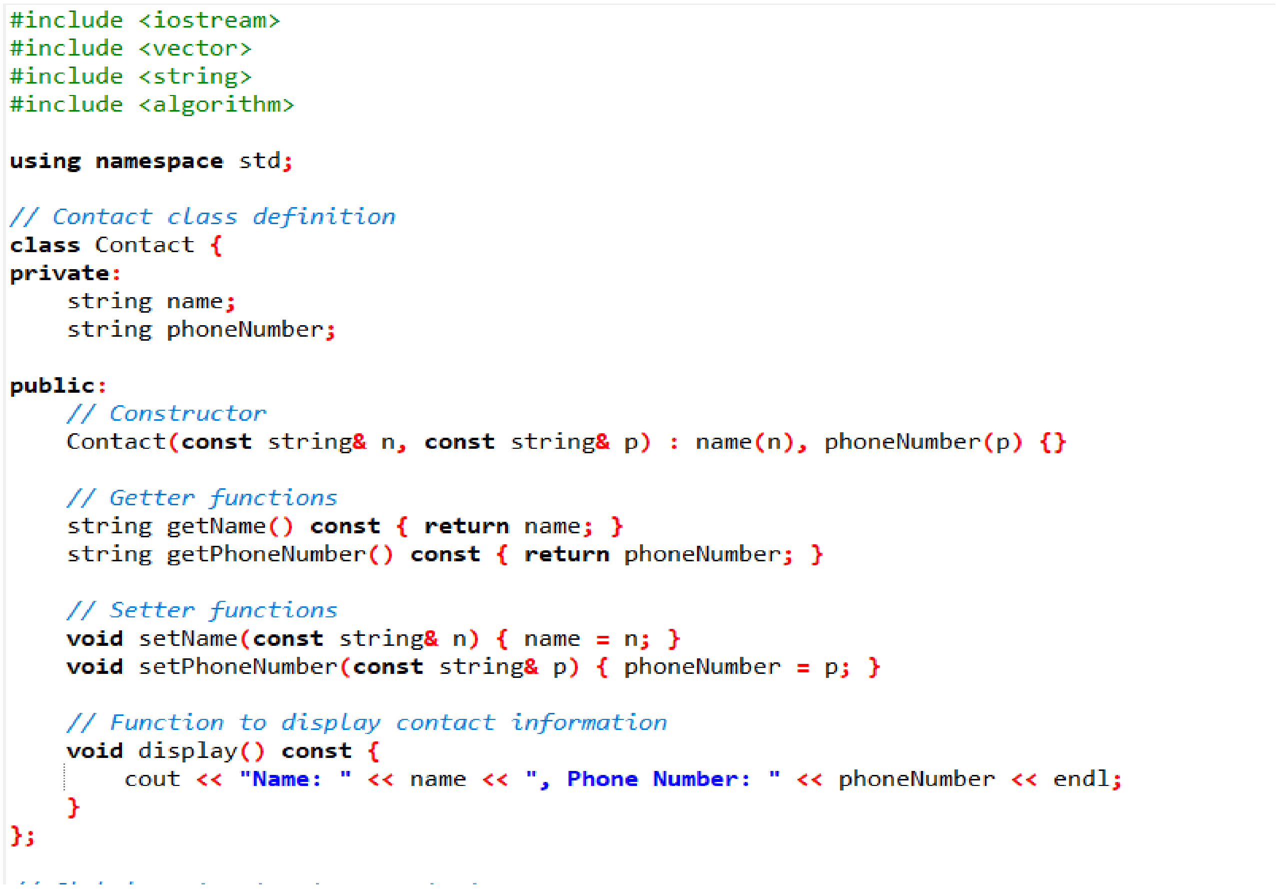
- Objective: Implement a simple system to manage contacts.
- Description: Develop a C++ program that allows users to add, view, and delete contacts. Each contact should have a name and a phone number.

Key Steps:

- Defining a Contact class with appropriate attributes
- Using vectors to store contact objects
- Implementing functions for adding, viewing, and deleting contacts
- Providing a menu-driven interface for user interaction

Solution:

Code:

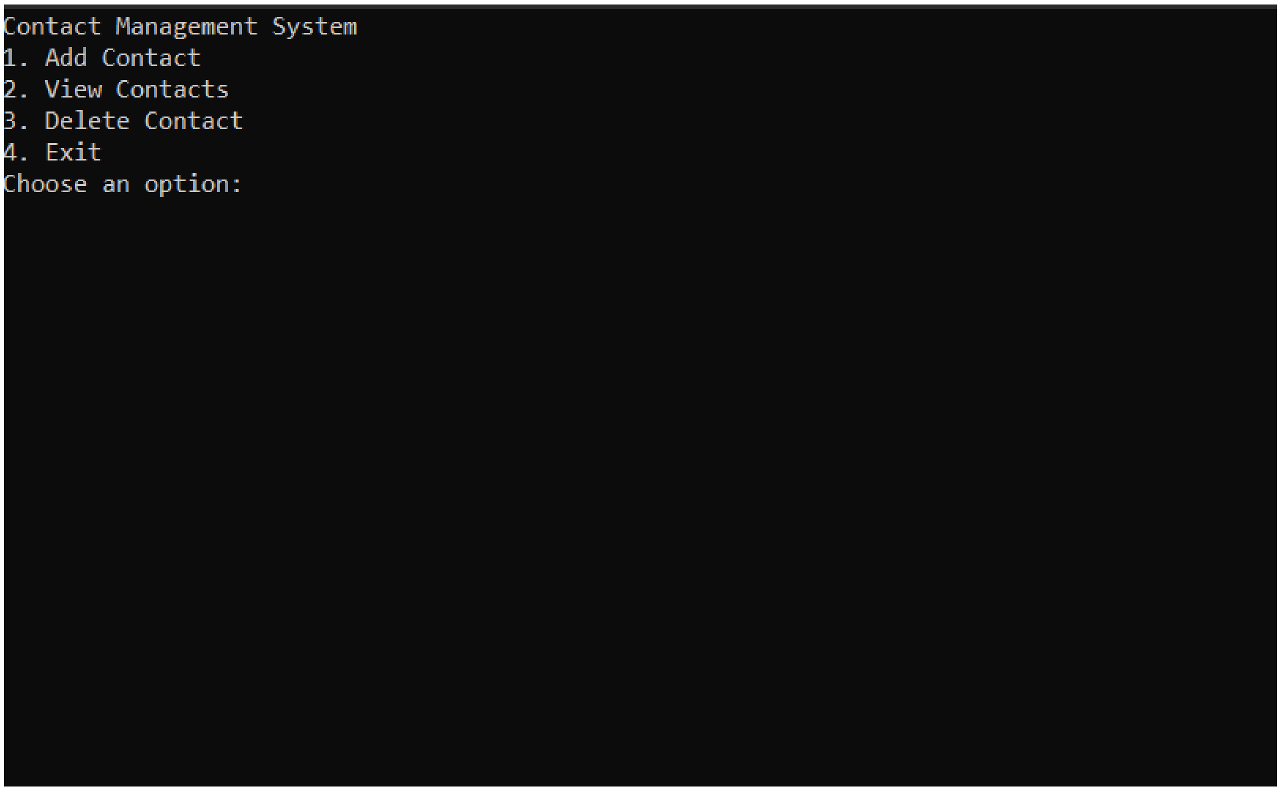


```
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>

using namespace std;

// Contact class definition
class Contact {
private:
    string name;
    string phoneNumber;

public:
    // Constructor
    Contact(const string& n, const string& p) : name(n), phoneNumber(p) {}

    // Getter functions
    string getName() const { return name; }
    string getPhoneNumber() const { return phoneNumber; }

    // Setter functions
    void setName(const string& n) { name = n; }
    void setPhoneNumber(const string& p) { phoneNumber = p; }

    // Function to display contact information
    void display() const {
        cout << "Name: " << name << ", Phone Number: " << phoneNumber << endl;
    }
};
```

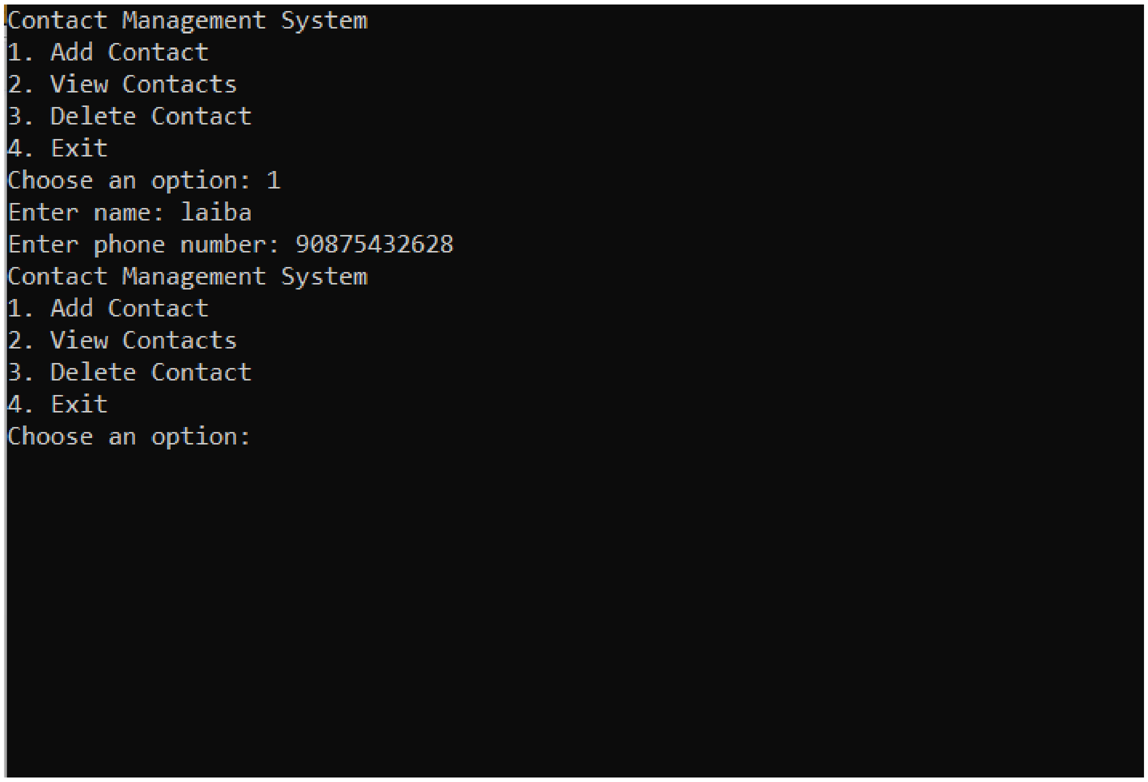
```cpp
// Global vector to store contacts
vector<Contact> contacts;

// Function to add a contact
void addContact(const string& name, const string& phoneNumber) {
    contacts.emplace_back(name, phoneNumber);
}

// Function to view all contacts
void viewContacts() {
    if (contacts.empty()) {
        cout << "No contacts available." << endl;
        return;
    }
    for (const auto& contact : contacts) {
        contact.display();
    }
}

// Function to delete a contact
void deleteContact(const string& name) {
    auto it = remove_if(contacts.begin(), contacts.end(),
                        [&name](const Contact& c) { return c.getName() == name; });
    if (it != contacts.end()) {
        contacts.erase(it, contacts.end());
        cout << "Contact deleted." << endl;
    } else {
        cout << "Contact not found." << endl;
    }
}
```

```cpp
// Function to display the menu
void showMenu() {
    cout << "Contact Management System\n";
    cout << "1. Add Contact\n";
    cout << "2. View Contacts\n";
    cout << "3. Delete Contact\n";
    cout << "4. Exit\n";
    cout << "Choose an option: ";
}

int main() {
    int choice;
    string name, phoneNumber;

    while (true) {
        showMenu();
        cin >> choice;
        cin.ignore(); // To ignore any newline character left in the buffer

        switch (choice) {
            case 1:
                cout << "Enter name: ";
                getline(cin, name);
                cout << "Enter phone number: ";
                getline(cin, phoneNumber);
                addContact(name, phoneNumber);
                break;
            case 2:
                viewContacts();
                break;
            case 3:
                cout << "Enter name of the contact to delete: ";

                break;
            case 3:
                cout << "Enter name of the contact to delete: ";
                getline(cin, name);
                deleteContact(name);
                break;
            case 4:
                cout << "Exiting...\n";
                return 0;
            default:
                cout << "Invalid option. Please try again.\n";
        }
    }
}
```

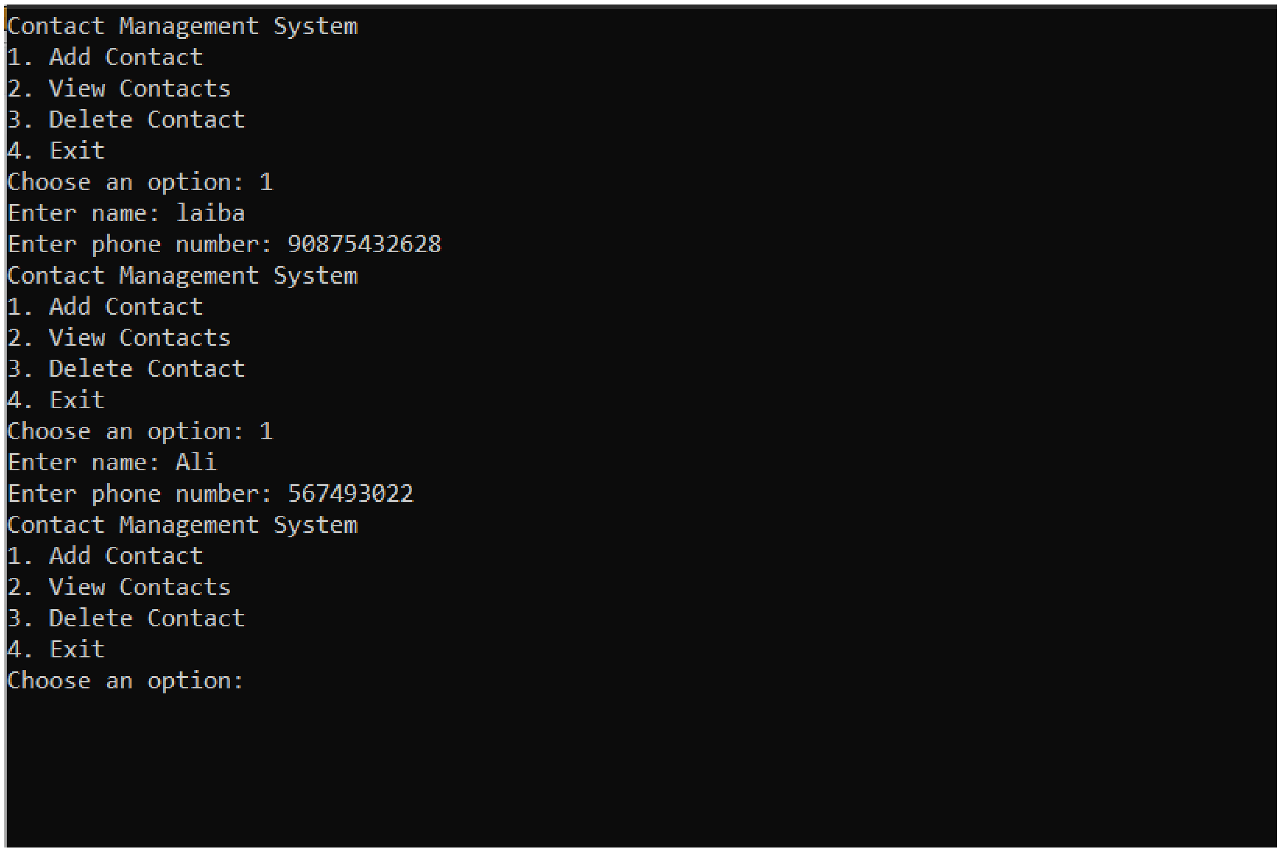
Output:



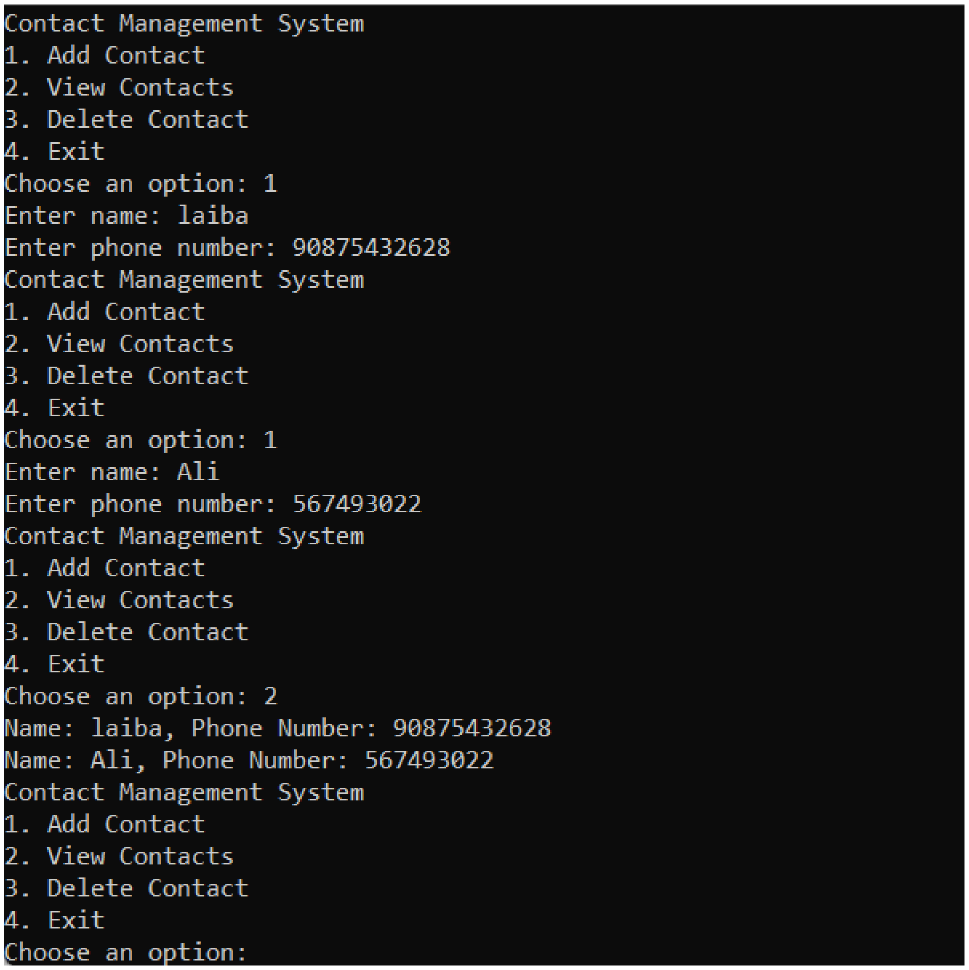
```
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option:
```



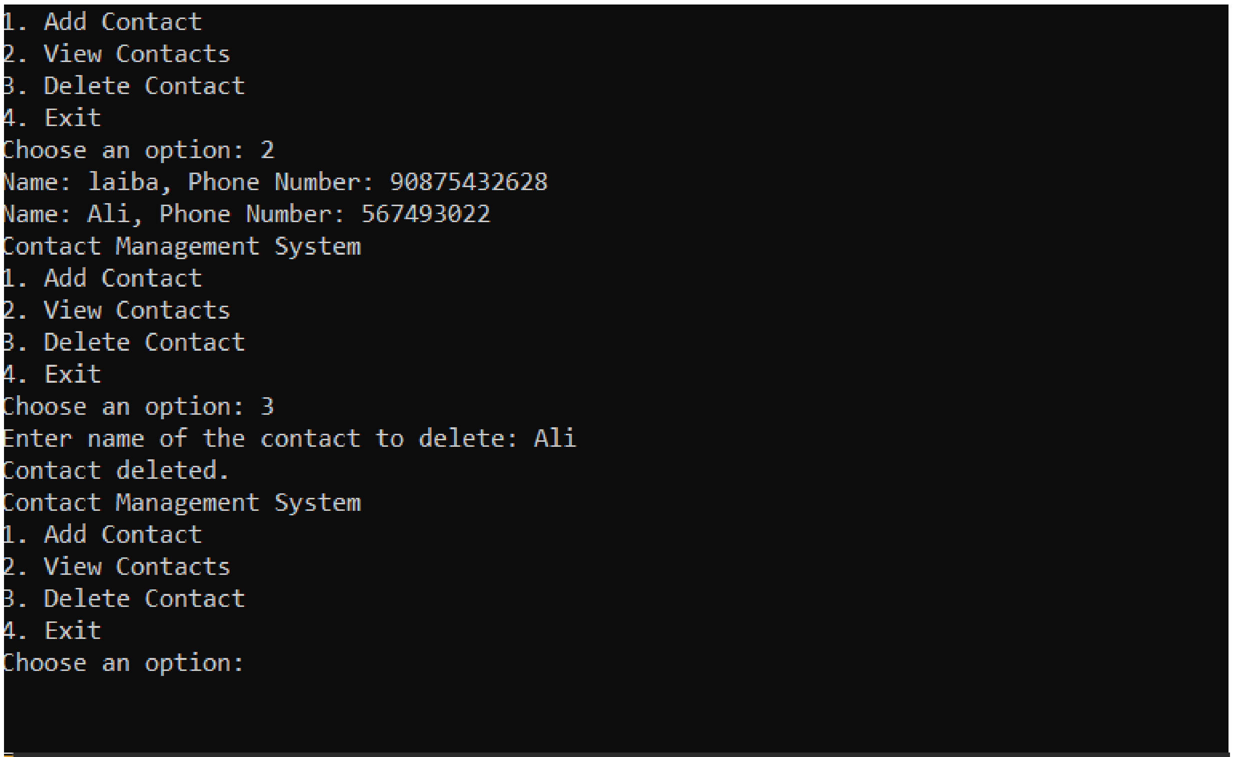
```
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 1
Enter name: laiba
Enter phone number: 90875432628
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option:
```



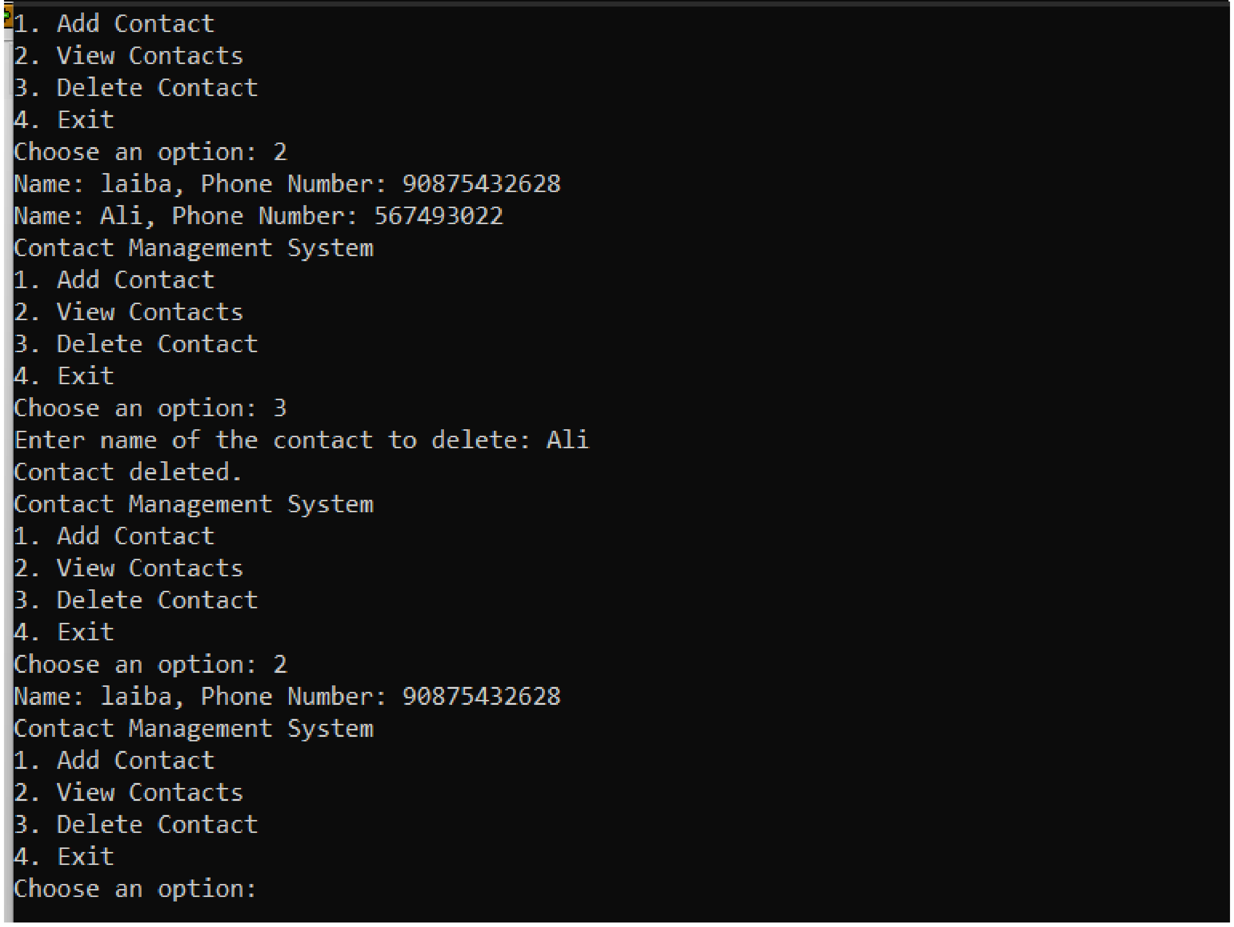
```
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 1
Enter name: laiba
Enter phone number: 90875432628
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 1
Enter name: Ali
Enter phone number: 567493022
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option:
```



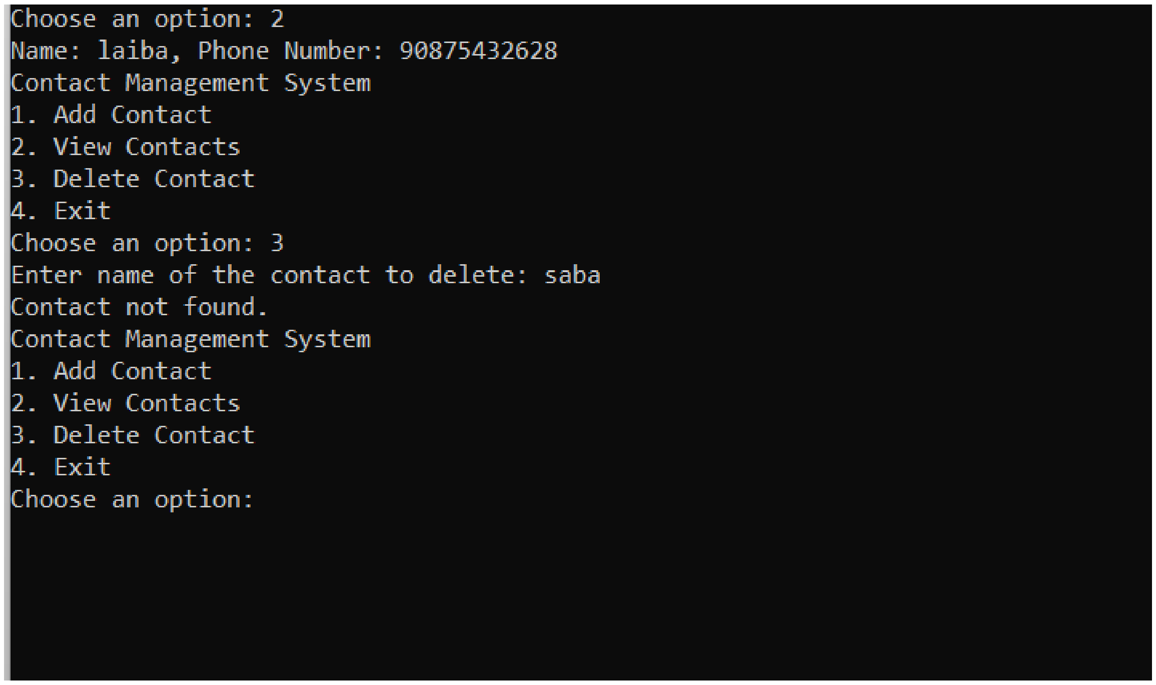
```
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 1
Enter name: laiba
Enter phone number: 90875432628
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 1
Enter name: Ali
Enter phone number: 567493022
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 2
Name: laiba, Phone Number: 90875432628
Name: Ali, Phone Number: 567493022
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option:
```



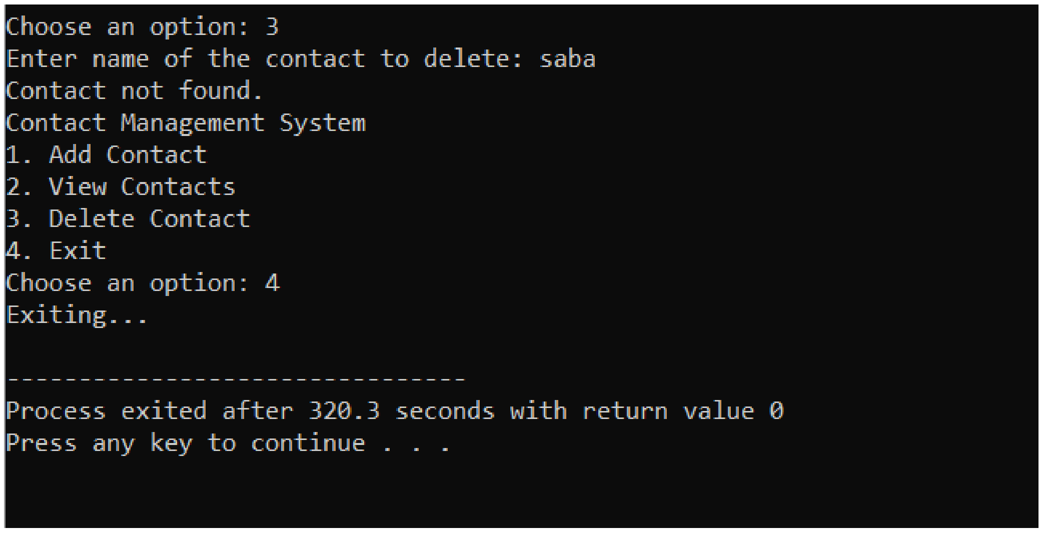
```
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 2
Name: laiba, Phone Number: 90875432628
Name: Ali, Phone Number: 567493022
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 3
Enter name of the contact to delete: Ali
Contact deleted.
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option:
```



```
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 2
Name: laiba, Phone Number: 90875432628
Name: Ali, Phone Number: 567493022
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 3
Enter name of the contact to delete: Ali
Contact deleted.
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 2
Name: laiba, Phone Number: 90875432628
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option:
```



```
Choose an option: 2
Name: laiba, Phone Number: 90875432628
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 3
Enter name of the contact to delete: saba
Contact not found.
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option:
```



```
Choose an option: 3
Enter name of the contact to delete: saba
Contact not found.
Contact Management System
1. Add Contact
2. View Contacts
3. Delete Contact
4. Exit
Choose an option: 4
Exiting...

--------------------------------
Process exited after 320.3 seconds with return value 0
Press any key to continue . . .
```

Explanation:

Overview

This program manages a list of contacts with a simple menu interface. You can add, view, or delete contacts.

Key Components

1. **Header Files**:

- #include <iostream>: For input and output operations.
- #include <vector>: For storing the list of contacts.
- #include <string>: For handling text data like names and phone numbers.
- #include <algorithm>: For functions like removing items from the list.

2. **Namespace**:

    - using namespace std; simplifies code by avoiding std:: before standard library names.

3. **Contact Class**:

    - **Attributes**: Stores a contact's name and phoneNumber.
    - **Constructor**: Initializes a contact with a name and phone number.
    - **Methods**:
    - display(): Shows the contact’s details on the screen.

4. **Global Vector**:

    - vector<Contact> contacts;: Stores all the contacts.

5. **Functions**:

    - **`addContact(name, phoneNumber)`**: Adds a new contact to the list.
    - **`viewContacts()`**: Shows all contacts. If there are none, it informs the user.
    - **`deleteContact(name)`**: Removes a contact by name.

6. **Menu System**:

    - **`showMenu()`**: Displays the options (Add, View, Delete, Exit).
    - **`main()`**: Handles user input and menu navigation. Based on the user's choice, it calls the appropriate function to add, view, or delete contacts.

Program Flow

1. **Display Menu**: Shows options to the user.
2. **User Choice**:

    - **Add Contact**: Prompts for name and phone number, then adds the contact.
    - **View Contacts**: Lists all contacts.
    - **Delete Contact**: Asks for the contact name to delete and removes it if found.
    - **Exit**: Ends the program.

3. **Repeat**: The menu is displayed again until the user chooses to exit.