Object Oriented Programming:

Assignment 1:

Scenario:  
Make a console-based messaging application which has various functions to store messages, send messages, display messages, delete messages, send message to a particular person, display all seen messages, display all unseen messages and shuffle all messages.

Discussing the internal working of code and explanation of output:  
As it is menu- driven and user-friendly application. It will display some details on the console which is shown below:

After the menu is explained the user can choose numbers between 1 – 10.

1. To see users contact list:

If the user selects this option by selecting 1, the application displays the list of contacts stored in the system.

Example output:

1. Send a message:

This option allows a user to send predefined messages to various contacts. Here contacts can be stored with receivers number too but usually people have saved contacts to whom they frequently talk to. The message will be stored with a unique id., status (seen / unseen) and a timestamp.

Example output:

3- Display all Messages:

This option displays all messages sent to various contacts. It loops through all contacts and their respective messages and prints all necessary details associated with a message such as Message id, Message content, Message status, Message timestamp.

And so on….

1. Delete a message:

This option allows the user to delete a specific message by its message Id . This function will find and remove the messages and shift the remaining messages to maintain order.

It displayed that the message has been deleted after taking user input. Now if we again display the messages. The message with id 004 will no longer be displayed.

1. Send a message to a specific user:

This option allows the user to manually input the recipient’s name, message content and the status whether the message has been seen or not.

Now if we again display the messages this message will be added to the message array.

1. Search for a particular message by receiver:

The user can search all messages which were exchanged with a particular contact by entering their name. This will return all messages with that particular contact.

1. To view all seen messages:  
   This option filters and displays all messages which ae marked as seen.

And so on….

1. View all unseen messages:  
   This option prints all those messages which are left unseen by the receiver.

Example Output:

And so on….

1. Shuffle all messages:

This option Randomly redistribute messages across all contacts. It then redistributes the messages into the message array.

Example output:

Messages are now displayed randomly.

10-Exit the application:  
Selecting this option will terminate the program.

Message App class functions explained:

1. Send Message:

First the method checks that if the receiver No exists in the contact Array

If it is found, then it checks if there is space in message array.

It creates a new message object with all instance variables and generates a new id.

1. Display Message:

This method loops through all the contacts stored in the contact array and then calls displayMessage() in Message class and prints all details.

3-Delete Message:

This method searches for the message with the message Id in the array ,

If it is found it deletes the message from that array.

The remaining messages are shifted forward.

Message count is decremented.

4-send Receiver:

The user is prompted to enter the receiver name, message content and status.

5-Search:

This method loops through all contact array , if the contact matches receiver no , it displays all messages exchanged with that contact by calling displaymessage().

6-seenMessages:

It loops through all messages for each contact and prints only the ones which have status true.

7- unseenMessages:

It loops through all messages for each contact and prints only the ones which have status false.

8-shuffleMessages:

It collects all the messages from messages array into a single array   
The messages are shuffled using a random number generator.

Then the messages are reassigned back into the messages array.

The UML Diagram is pasted below :