|  |  |
| --- | --- |
| Name: | Ayesha Javed |
| Registration number: | Sp24-bse-020 |
| Section: | B |
| Assignment number: | 1 |
| Course instructor: | Sir Shahid |
| Date: | 11-10-2024 |



*Course: Object Oriented Programming (OOP)*

*Topic:*

“SMS Managing System”

SMS Managing System:

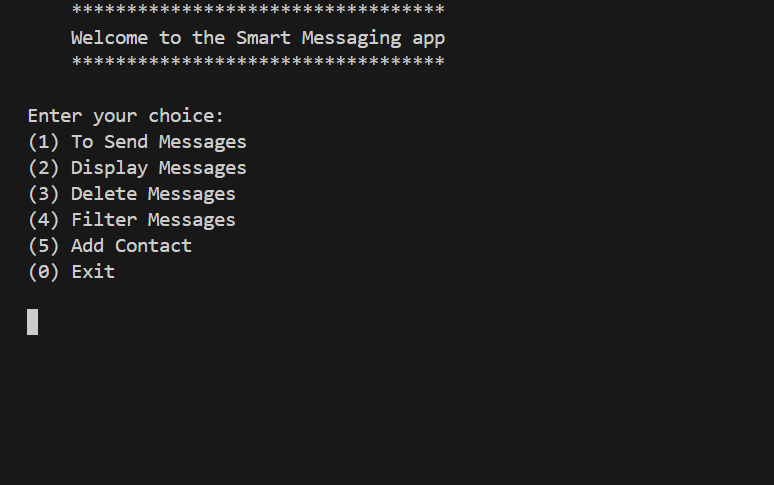
This is a simple SMS managing System that has various features and functions implementing the concepts of Object-Oriented Programming, 2D-array, method Overloading, Overriding and composition.

Functionalities:

1. *Send Message*
2. *Display message (Single contact, all contacts)*
3. *Filter Messages (Contact wise (status wise), status wise)*
4. *Delete Messages (Single contact (Using ID), all messages)*
5. *Add contact*
6. *Sorting messages(sent time)*

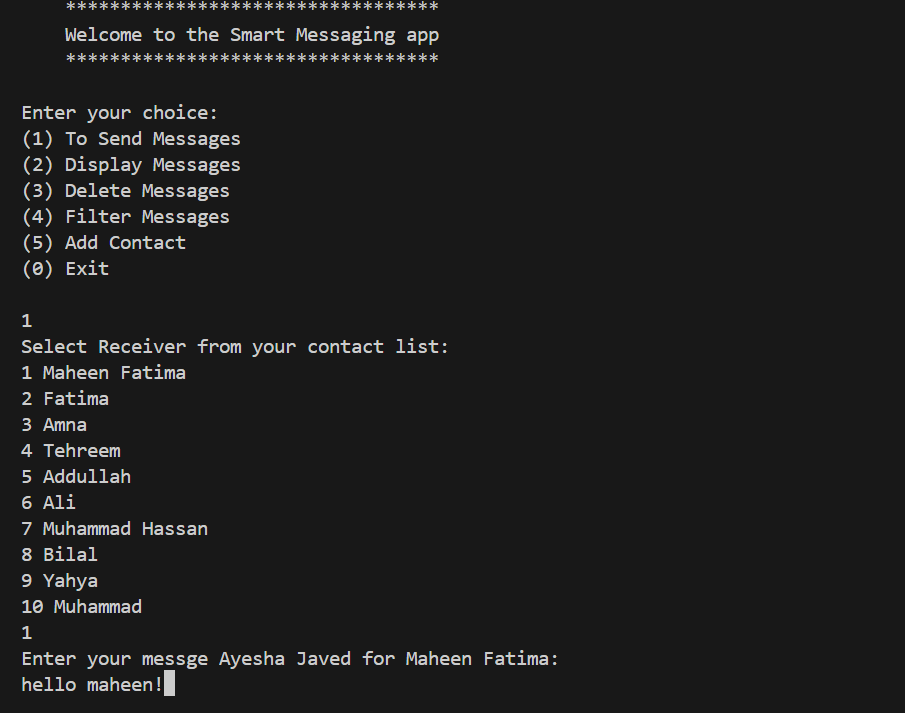
Console Working:

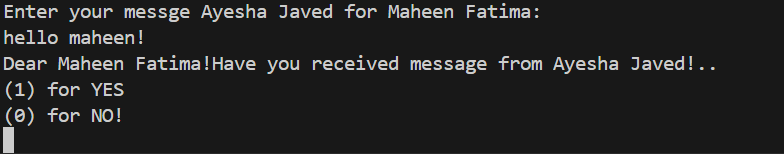
*User Interface:*



*1)Sending message*

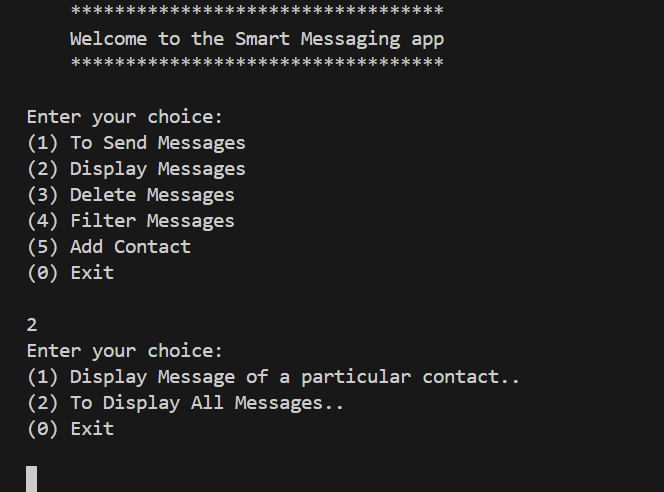
After selecting 1, the list of contacts appears, the contact is selected and the message is typed!

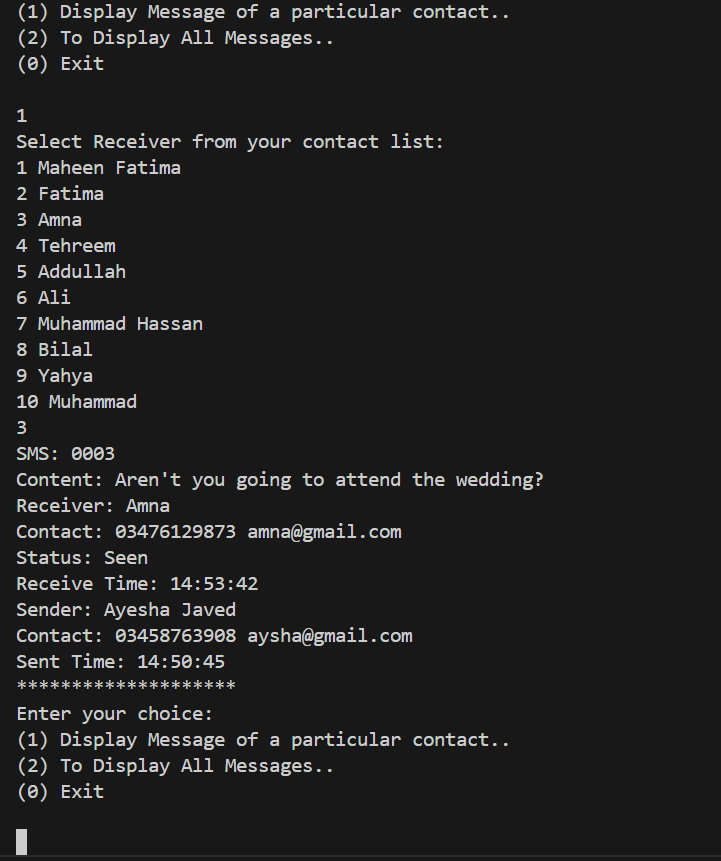


Now, the Program asks, whether the receiver has received the message from me or not!

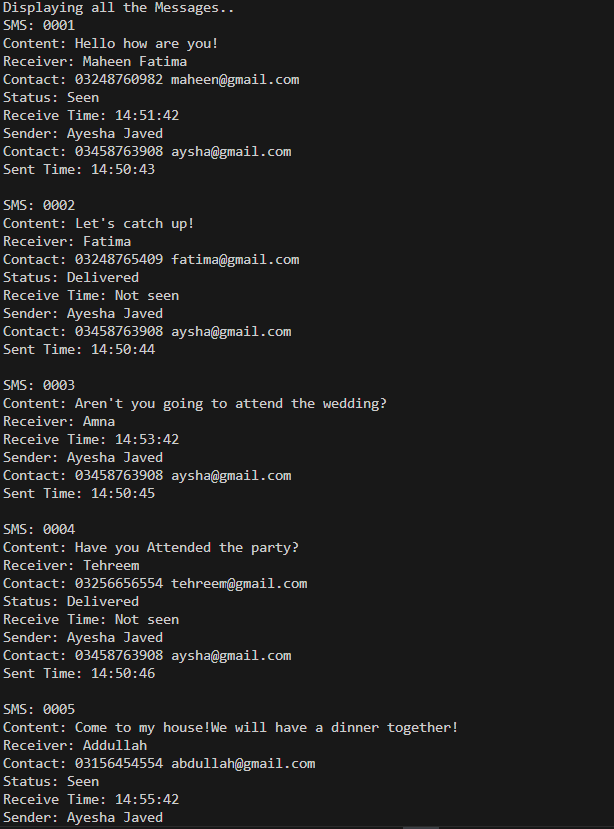
*2)Display message:*

The display function is called, now there are two choices whether to display contact wise, or to display all.

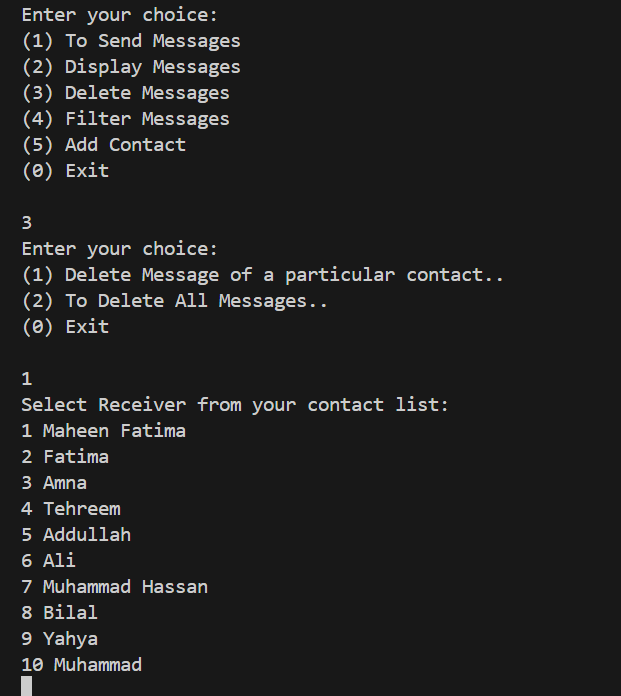


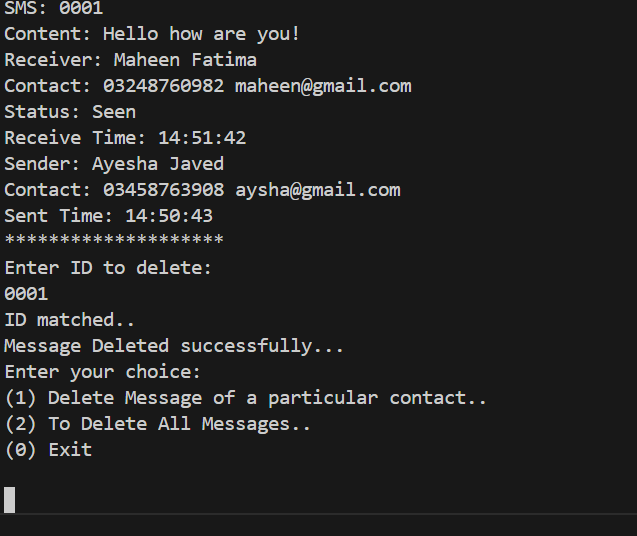


Displaying all the messages...

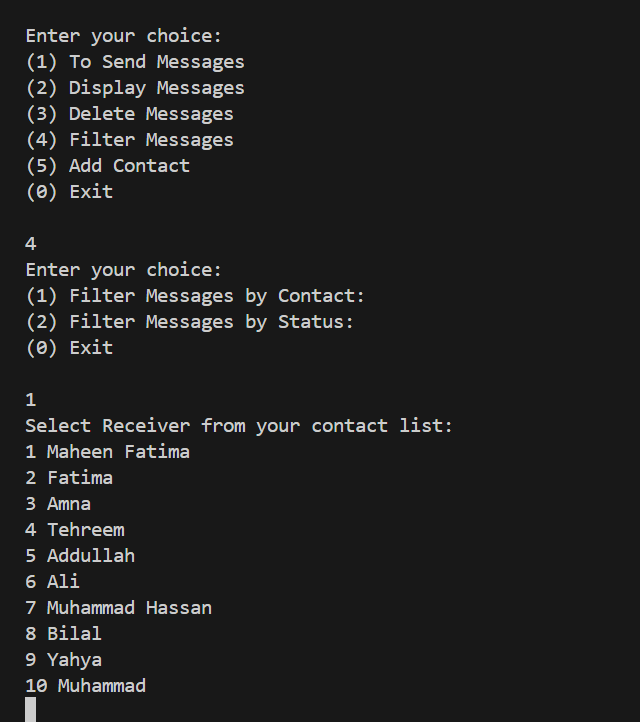


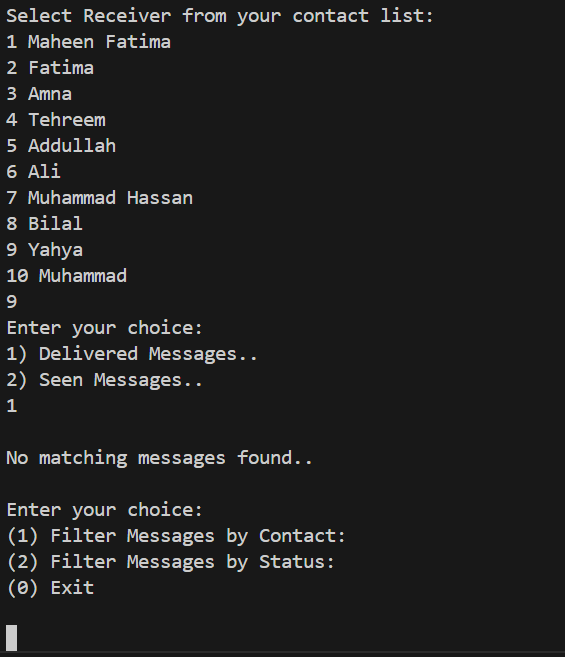
*3)Delete message*

now I am deleting message of maheen of id 0001

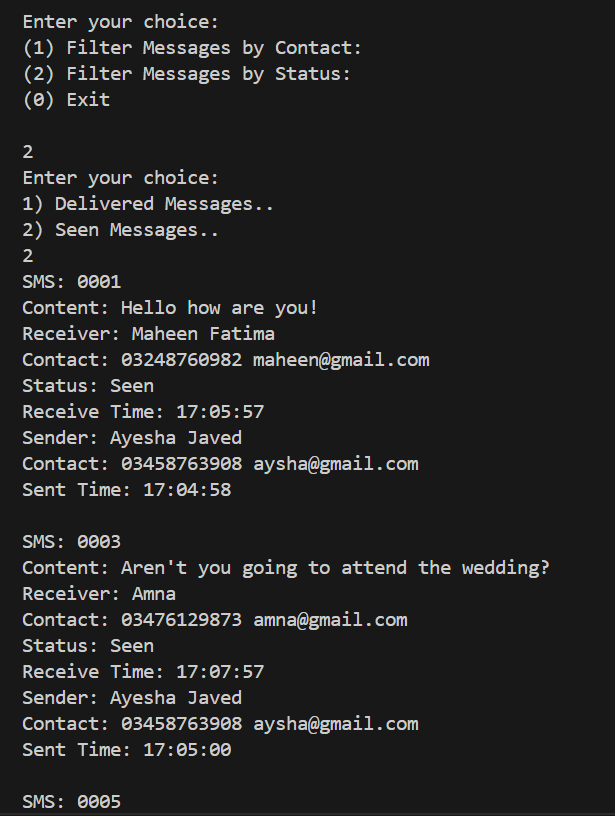
*4)Filter messages*

**By contact**



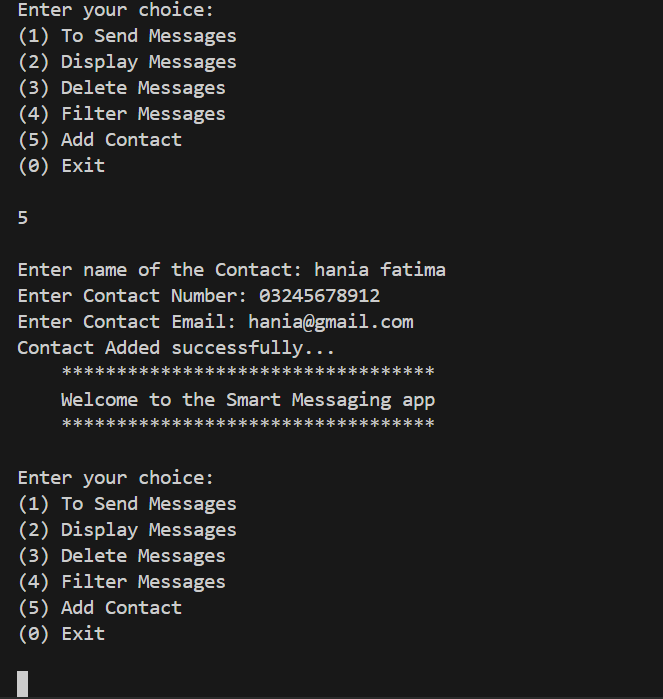


Now, as the contact 9 (Yahya), there are no delivered messages so, it displays that there are no delivered messages.

**By status:**

*5)Adding Contact:*

Now, I am adding the contact named, Hania her contact number and her email. In the code, the array is increased in size and a duplicate array is formed and again its assigned to the previous array.



*6) Sorting Messages:*

Entering 6, sorts the messages in the 2d array.

***Description of Sorting Using Comparator and Arrays.sort()***

In this code, we utilize the Arrays.sort() method to sort an array of SMS objects based on their sent time. The Arrays.sort() method is called with three parameters: the chats[i] array, the starting index (0), and the ending index (chat Index[i]). This sorts only the portion of the array that contains actual messages.

A custom Comparator<SMS> is provided to define the sorting criteria. In this case, the messages are sorted based on the sent Time attribute of each SMS object.

The compare method of the Comparator<SMS> takes two SMS objects (sms1 and sms2) as parameters.

To achieve a descending order sort (i.e., newer messages appear first), the compare method compares the sent Time of the two SMS objects using sms2.getSentTime().compare To(sms1.getSentTime()).

* If sms2 has a later sent time than sms1, this will return a positive value, indicating that sms2 should come before sms1 in the sorted order.

