COMP132 Assignment-2 Report

Name: İsmail Ozan Kayacan
ID: 69103

PROJECT DESIGN:

When the program is run, it prints the following in the console, (as long as user does not enter 4, it keeps asking):

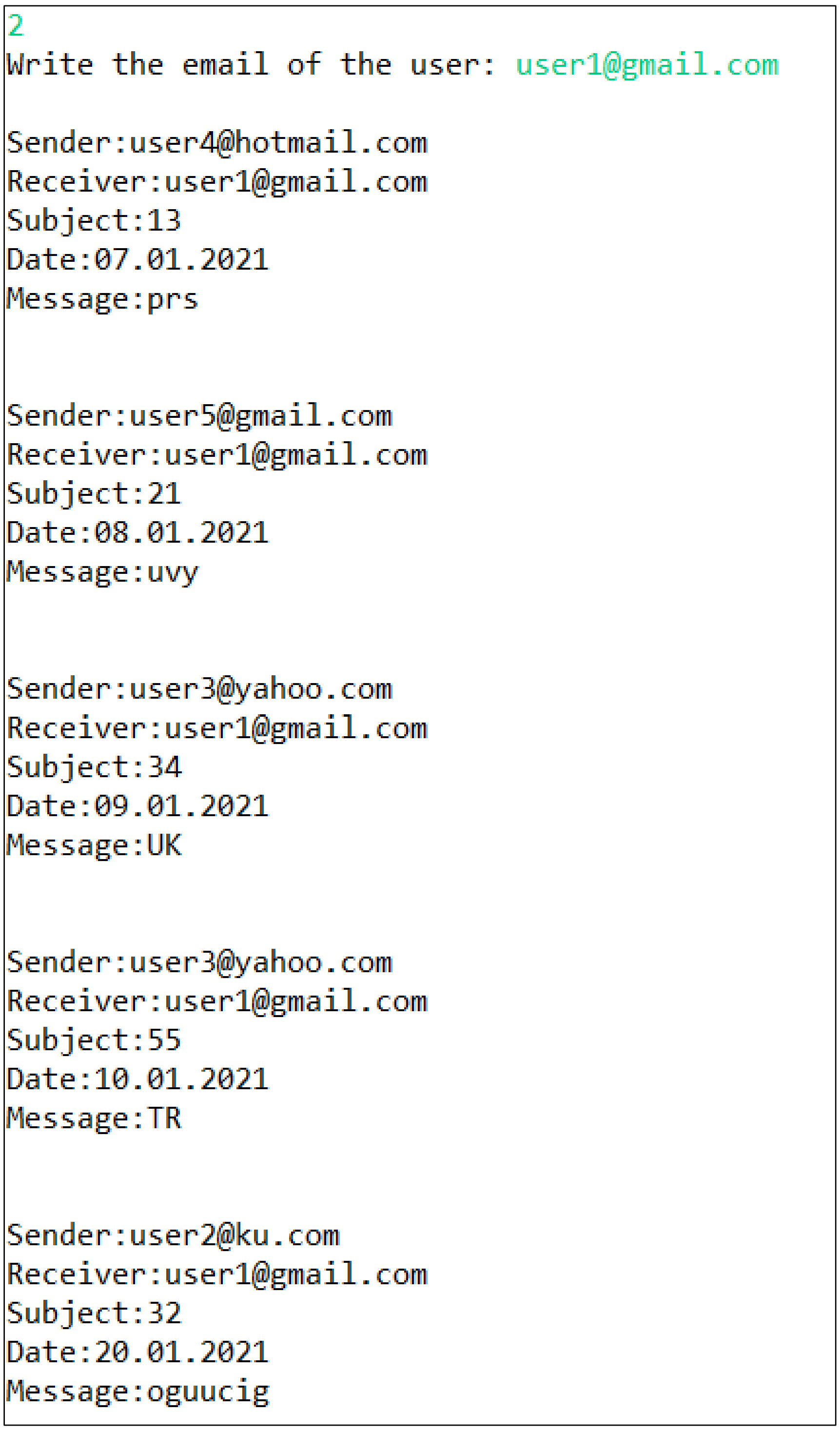
```
MENU:
1-Send message.
2-Display last 5 emails of an user.
3-Display the emails in an inbox containing a given keyword in the header.
4-Exit
```

If user enters 1, it asks the necessary information about the mail, then adds the mail to the receivers mail linkedlist. (It also shows the calculated subject and decrypted message.) *(If the user with the receiver address does not exist it prints "User not found", if subject is not in the correct format, it prints "Invalid expression!")*

Example with correct input:

```
MENU:
1-Send message.
2-Display last 5 emails of an user.
3-Display the emails in an inbox containing a given keyword in the header.
4-Exit
1
Sender: user2@ku.com
Receiver Address: user1@gmail.com
Subject: 44 - 33 + 17 + 4
Calculated Mail Subject: 32
Date: 20.01.2021
Encrypted Message Content: message
Ceaser Chiper Shift (n): 2

Mail is sent successfully:
Sender:user2@ku.com
Receiver:user1@gmail.com
Subject:32
Date:20.01.2021
Message:oguucig

-------------------------------------------------------------------

MENU:
1-Send message
2-Display last 5 mails of an user
3-Display the emails in an inbox containing a given keyword in the header.
4-Exit
```

If user enters 2, it asks the user’s gmail, and prints last 5 mails that user received. *(If the user with the receiver address does not exist it prints “User not found”).*
Example with correct input:

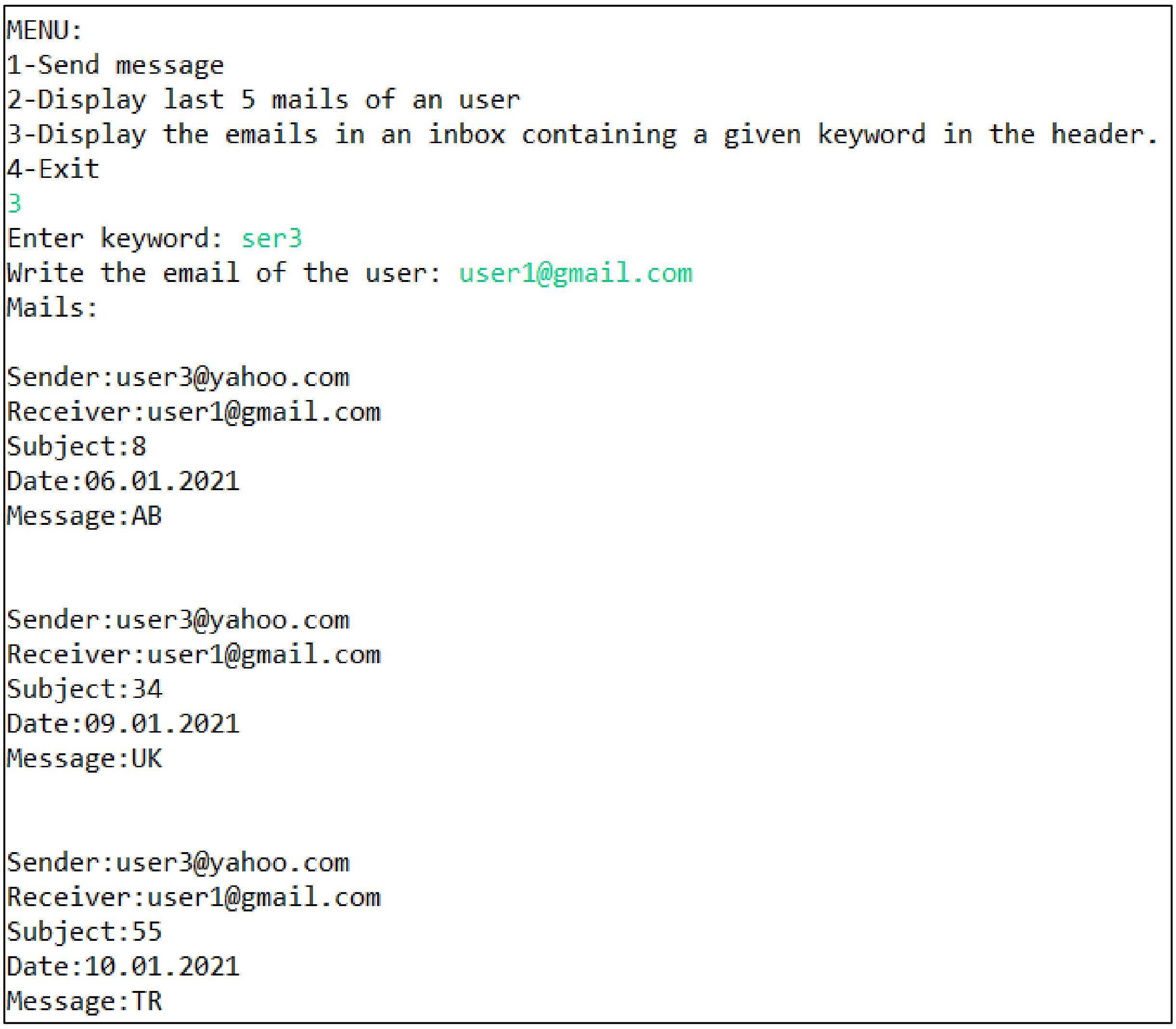


```
2
Write the email of the user: user1@gmail.com

Sender:user4@hotmail.com
Receiver:user1@gmail.com
Subject:13
Date:07.01.2021
Message:prs


Sender:user5@gmail.com
Receiver:user1@gmail.com
Subject:21
Date:08.01.2021
Message:uvy


Sender:user3@yahoo.com
Receiver:user1@gmail.com
Subject:34
Date:09.01.2021
Message:UK


Sender:user3@yahoo.com
Receiver:user1@gmail.com
Subject:55
Date:10.01.2021
Message:TR


Sender:user2@ku.com
Receiver:user1@gmail.com
Subject:32
Date:20.01.2021
Message:oguucig
```

Note: (The previous emails were sent in the main method for test.)

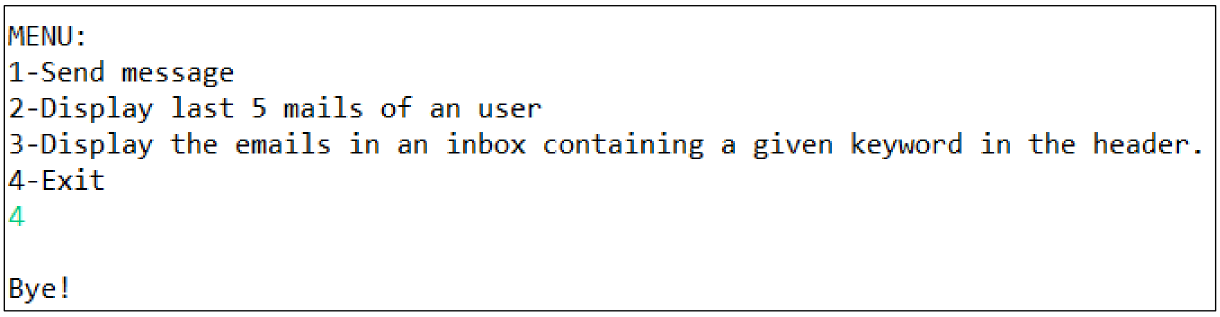
If user enters 3, it asks the keyword and the user's gmail, then prints all the mails whose header contains the keyword. *(If the user with the receiver address does not exist it prints "User not found").*

Example with correct input:



```
MENU:
1-Send message
2-Display last 5 mails of an user
3-Display the emails in an inbox containing a given keyword in the header.
4-Exit
3
Enter keyword: ser3
Write the email of the user: user1@gmail.com
Mails:

Sender:user3@yahoo.com
Receiver:user1@gmail.com
Subject:8
Date:06.01.2021
Message:AB


Sender:user3@yahoo.com
Receiver:user1@gmail.com
Subject:34
Date:09.01.2021
Message:UK


Sender:user3@yahoo.com
Receiver:user1@gmail.com
Subject:55
Date:10.01.2021
Message:TR
```

If user enters 4, it prints "Bye!" and terminates the program.



```
MENU:
1-Send message
2-Display last 5 mails of an user
3-Display the emails in an inbox containing a given keyword in the header.
4-Exit
4

Bye!
```

STRUCTURES:

1- **Mail:** It has String sender, String receiverAddress, int Subject, String date and String messageContent.
2- **ListNodeMail:** It has Mail mail and ListNodeMail *nextPtr.
3- **User:** It has String eMail and ListNodeMailPtr inbox.
4- **ListNodeUser:** It has User user and ListNodeUser *nextPtr.
5- **MailServer:** It has String domainName and ListNodeUserPtr usersList.

FUNCTIONS:

1- **char* decrypt(char *s, int n):** Takes the arguments “char *s” and “int n”. Then by shifting all alphabetical characters of “*s”, by “n”, it returns new string.
2- **int correctSubject(char *s):** Takes the argument “char *s”. If “*s” is in the desired subject format, returns 1. Otherwise returns 0.
3- **int calculateSubject(char *s):** Takes the argument “char *s”. And returns the result of the arithmetic operations as integer.
4- **void insertUser(ListNodeUserPtr *headPtr, User u):** Takes the arguments “ListNodeUserPtr *headPtr” and “User u”. Then it adds the “u” to the end of the linkedlist whose first node is pointed by “headPtr”.
5- **void insertMail(ListNodeMailPtr *headPtr, Mail m):** Takes the arguments “ListNodeMailPtr *headPtr” and “Mail m”. Then it adds the “m” to the end of the linkedlist whose first node is pointed by “headPtr”.
6- **void printUserList(ListNodeUserPtr headPtr):** Takes the argument “ListNodeUserPtr headPtr”. Then prints all the users in the linkedlist whose first node is pointed by “headPtr”.
7- **void printMailList(ListNodeMailPtr headPtr):** Takes the argument “ListNodeMailPtr headPtr”. Then prints all the mails in the linkedlist whose first node is pointed by “headPtr”.
8- **User findUser(char *eMail, ListNodeUserPtr headPtr):** Takes the arguments “char *eMail” and “ListNodeUserPtr headPtr”. Then returns the user whose eMail is equal to “*eMail”.
9- **void printWithKeyword(char *key, ListNodeMailPtr headPtr):** Takes the arguments “char *key” and “ListNodeMailPtr headPtr”. Then prints all the mails containing “*key” in the linkedlist whose first node is pointed by “headPtr”.
10- **void printLast5Mails(ListNodeMailPtr headPtr):** Takes the argument “ListNodeMailPtr headPtr”. Then prints last 5 mails in the linkedlist whose first node is pointed by “headPtr”.