```java
import javax.mail.*;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
import java.io.*;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Properties;
import java.util.Scanner;
import java.time.LocalDate;

public class Test2 {
    static ArrayList<Email> emailList = new ArrayList<Email>();
    static LocalDate date = LocalDate.now();

    public static void main(String[] args) throws IOException,
MessagingException, ClassNotFoundException {

        //CHECK THE FOLLOWING LINES
        emailList = ByteFile.deSerialiaze("email.ser");
        // System.out.println(emailList.get(0).content);
        ArrayList<Recipients> recipientsList = new ArrayList<Recipients>();
        FileWrite.readFromFile(recipientsList, "clientList.txt");
        BirthdayWish.birthdayWish(recipientsList, date);

        Scanner scanner = new Scanner(System.in);

        boolean bool = true;
        while (bool) {
            System.out.println("Enter option type: \n"
                    + "1 - Adding a new recipient\n"
                    + "2 - Sending an email\n"
                    + "3 - Printing out all the recipients who have
birthdays\n"
                    + "4 - Printing out details of all the emails sent\n"
                    + "5 - Printing out the number of recipient objects in
the application\n"
                    + "6 - Exit");

            int option = scanner.nextInt();


            RecipientsFactory recipientsFactory = new RecipientsFactory();
            switch (option) {
                case 1:
                    // input format - Official: nimal,nimal@gmail.com,ceo
                    //Personal: sunil<nickname>,email,2000/08/10
                    //Office_friend: Kamal, kamal@gmail.com, cerlk,
2000/12/12
                    // Use a single input to get all the details of a
recipient
                    // code to add a new recipient
                    // store details in clientList.txt file
                    // Hint: use methods for reading and writing files
                    System.out.println("Enter the recipient : ");
                    Scanner scanner1 = new Scanner(System.in);
                    String inputDetails = scanner1.nextLine();
                    String[] recipientType = inputDetails.split(":");
```

```
                    recipientsFactory.list = recipientType[1].split(",");
                    Recipients recipient =
recipientsFactory.createRecipient(recipientType[0].strip());
                    FileWrite.writeToFile("clientList.txt",inputDetails);
                    recipientsList.add(recipient);
                    BirthdayWish.birthdayWish(recipientsList,date);

                    break;
                case 2:
                    // input format - email, subject, content
                    // code to send an email
                    Scanner scanner2 = new Scanner(System.in);
                    String sendEmail = scanner2.nextLine();
                    String[] emailContent = sendEmail.strip().split(",");
                    String emailAddress = emailContent[0].strip();
                    String subject = emailContent[1].strip();
                    String content = emailContent[2].strip();
                    System.out.println("Sending...");
                    EmailSending.sendMail(emailAddress, subject, content);
                    Email email = new Email(emailAddress, subject, content,
date);
                    emailList = CreateArrayList.createTheArrayList(email,
"emails.ser");
                    //serialize(emailList, "email1.ser");
                    break;
                case 3:
                    // input format - yyyy/MM/dd (ex: 2018/09/17)
                    // code to print recipients who have birthdays on the
given date
                    Scanner scanner3 = new Scanner(System.in);
                    String date_Today = scanner3.nextLine();
                    String[] Date = date_Today.split("/");
                    Print.printRecipientsWithBirthdays(recipientsList, Date);

                    break;
                case 4:
                    // input format - yyyy/MM/dd (ex: 2018/09/17)
                    // code to print the details of all the emails sent on
the input date
                    Scanner scanner4 = new Scanner(System.in);
                    String dateToday = scanner4.nextLine();
                    String[] date = dateToday.split("/");

                    EmailDetails.findEmailDetail(emailList, date);

                    break;
                case 5:
                    // code to print the number of recipient objects in the
application
                    //
System.out.println(RecipientsFactory.recipients.size());
                    System.out.println(RecipientsFactory.noOfRecipients);
                    break;
                case 6:
                    bool = false;
            }
```

```
            ByteFile.serialize(emailList,"email.ser");

            // start email client
            // code to create objects for each recipient in clientList.txt
            // use necessary variables, methods and classes

        }
    }
}

 class FileWrite{
    public static void  writeToFile(String my_file_name, String write) throws
IOException {
        FileWriter fileWriter = new FileWriter(my_file_name, true);
        BufferedWriter file = new BufferedWriter(fileWriter);
        file.write(String.format("%s\n",write));
        file.close();
        fileWriter.close();
    }

    public static void readFromFile(ArrayList<Recipients> recipients, String
my_file_name){
        try{
            FileReader fileReader = new FileReader(my_file_name);
            BufferedReader file = new BufferedReader(fileReader);
            RecipientsFactory recipientsFactory = new RecipientsFactory();
            Recipients recipient = null;
            String line = null;

            while((line = file.readLine()) != null){
                String[] recipientType = line.split(":");
                //System.out.println(recipientType[0]);
                recipientsFactory.list = recipientType[1].split(",");
                recipient =
recipientsFactory.createRecipient(recipientType[0].strip());
                recipients.add(recipient);
            }
        }
        catch (IOException e){
            e.printStackTrace();
        }

    }
}
class BirthdayWish {

    public static void birthdayWish(ArrayList<Recipients> recipients,
LocalDate date) throws MessagingException, IOException,
ClassNotFoundException {
        //  LocalDate date = Date;
        String dateToday = date.format(DateTimeFormatter.ISO_DATE);
        String[] month_and_Date_Today = dateToday.split("-");
        for(Recipients recipient : recipients){
            if(recipient instanceof Office_friend ){
                String[] month_and_date_of_OfficeFriend = ((Office_friend)
recipient).birthday.split("/");
                boolean check_month =
```

```
(month_and_date_of_OfficeFriend[1].equals(month_and_Date_Today[1]));
                boolean check_date =
(month_and_date_of_OfficeFriend[2].equals(month_and_Date_Today[2]));
                if(check_month && check_date){
                    Office_friend office_friend = ((Office_friend)
recipient);
                    String name = office_friend.name;
                    String emailAddress = office_friend.email;
                    String subject = "";
                    String content = "Wish you a Happy Birthday, Chamil";
                    Email emailObj = new Email(emailAddress,
subject,content,date);

                    if(CheckEmailObjects.checkEmailObj(emailObj,
Test2.emailList)){
                        System.out.printf("Sending wishes to %s...\n",name);
                        EmailSending.sendMail(emailAddress,subject, content);
                        Test2.emailList =
CreateArrayList.createTheArrayList(emailObj, "emails.ser");
                    }
                }
            }
            if(recipient instanceof Personal){
                String[] month_and_date_of_Personal = ((Personal)
recipient).birthday.split("/");
                boolean check_month =
(month_and_date_of_Personal[1].equals(month_and_Date_Today[1]));
                boolean check_date =
(month_and_date_of_Personal[2].equals(month_and_Date_Today[2]));

                if(check_month && check_date){
                    Personal personal = (Personal) recipient;
                    String name = personal.name;
                    String emailAddress = personal.email;
                    String subject = "";
                    String content = "hugs and love on your birthday,
Chamil";
                    Email emailObj = new Email(emailAddress,
subject,content,date);
                    if(CheckEmailObjects.checkEmailObj(emailObj,
Test2.emailList)){
                        System.out.printf("Sending wishes to %s...\n",name);
                        EmailSending.sendMail(emailAddress,subject, content);
                        Test2.emailList =
CreateArrayList.createTheArrayList(emailObj, "emails.ser");
                    }
                }
            }
        }
    }
}
class ByteFile {
    public static void serialize(ArrayList<Email> List, String file_name)
throws IOException {

        FileOutputStream fileOutputStream = new FileOutputStream(file_name);
        ObjectOutputStream file = new ObjectOutputStream(fileOutputStream);
```

```
        file.writeObject(List);
        file.close();
        fileOutputStream.close();
    }
    public static ArrayList<Email> deSerialiaze(String file_name) throws
IOException, ClassNotFoundException {
        ArrayList<Email> List = new ArrayList<Email>();
        try{
            FileInputStream fileInputStream = new FileInputStream(file_name);
            ObjectInputStream file = new ObjectInputStream(fileInputStream);
            List = (ArrayList<Email>)file.readObject();
            file.close();
            fileInputStream.close();
        }
        catch(IOException e){
            System.out.println("First Time");
        }

        return List;
    }
}
class CheckEmailObjects {
    public static boolean checkEmailObj(Email mail, ArrayList<Email>
emailList) {
        boolean state = true;
        for(Email email:emailList){
            if(mail.emailAddress.equals(email.emailAddress) &&
mail.date.equals(email.date) && mail.subject.equals(email.subject) &&
mail.content.equals(email.content)){
                state = false;
            }
        }return state;
    }
}
class CreateArrayList {
    static ArrayList<Email> emailList = new ArrayList<Email>();
    public static ArrayList<Email> createTheArrayList(Email email, String
file_name) throws IOException, ClassNotFoundException {

        emailList.add(email);

        return emailList;
    }
}
class EmailSending {
    public static void sendMail(String recipient, String subject, String
content) throws MessagingException {

        Properties props = new Properties();

        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.starttls.enable", "true");
        props.put("mail.smtp.host", "smtp.gmail.com");
        props.put("mail.smtp.port", "587");

        String myAccountEmail = "travellerfox5@gmail.com";
```

```
        String password = "fmexbdfjccjzyxmf";
        Session session = Session.getInstance(props, new Authenticator() {
            @Override
            protected PasswordAuthentication getPasswordAuthentication() {
                return new PasswordAuthentication(myAccountEmail, password);
            }
        });
        //String emailContent = String.format("%s\n\n%s", subject, content);
        Message msg = prepareMessage(session, myAccountEmail, recipient,
subject, content);
        //assert msg != null;
        Transport.send(msg);
        System.out.println("Success :)");
    }

    public static Message prepareMessage(Session session, String
myAccountEmail, String recepient, String subject, String Content){
        try {
            MimeMessage msg = new MimeMessage(session);
            msg.setFrom(new InternetAddress(myAccountEmail));
            msg.setRecipient(Message.RecipientType.TO, new
InternetAddress(recepient));
            msg.setSubject(subject);
            //  msg.setSentDate(new Date());
            msg.setText(Content);
            return msg;
        } catch (MessagingException e) {
            System.out.println("send failed, exception: " + e);
        }
        return null;
    }
}

// making of Recipients
interface Recipients {

}
 class Office_friend implements Recipients {
     String name, email, designation, birthday;

     Office_friend(String name, String email, String designation, String
birthday) throws IOException {
         this.name = name;
         this.email = email;
         this.designation = designation;
         this.birthday = birthday;

     }
 }
 class Official implements Recipients {
     String name, email, designation;

     Official(String name, String email, String designation) throws
IOException {
         this.name = name;
         this.email = email;
         this.designation = designation;
```

```
     }
 }
class Personal implements Recipients {
    String name, nickname, email, birthday;

    Personal(String name, String nickname, String email, String birthday)
throws IOException {
        this.name = name;
        this.nickname = nickname;
        this.email = email;
        this.birthday = birthday;
    }
}

class RecipientsFactory {
    String[] list;
    public static int noOfRecipients;
    public static ArrayList<Recipients> recipients = new
ArrayList<Recipients>();
    public Recipients createRecipient(String type)  {
        if(type == null){
            return null;
        }
        if(type.equalsIgnoreCase("Official")) {
            noOfRecipients += 1;
            String name = list[0].strip();
            String email = list[1].strip();
            String designation = list[2].strip();
            try{
                Recipients recipient = new Official(name, email,
designation);
                recipients.add(recipient);
                return recipient;
            }
            catch(IOException e){
                e.printStackTrace();
            }
        } else if(type.equalsIgnoreCase("Office_friend")){
            noOfRecipients += 1;
            String name = list[0].strip();
            String email = list[1].strip();
            String designation = list[2].strip();
            String birthday = list[3].strip();
            try{
                Recipients recipient = new Office_friend(name, email,
designation, birthday);
                recipients.add(recipient);
                return recipient;
            }
            catch (IOException e){
                e.printStackTrace();
            }


        } else if(type.equalsIgnoreCase("Personal")){
            noOfRecipients += 1;
            String name = list[0].strip();
```

```
            String nickname = list[1].strip();
            String email = list[2].strip();
            String birthday = list[3].strip();
            try{
                Recipients recipient = new Personal(name, nickname, email,
birthday);
                recipients.add(recipient);
                return recipient;
            }
            catch(IOException e){
                e.printStackTrace();
            }
        }
        return null;
    }
}
class Print {
    public static void printDetails(Email email){
        System.out.printf("You have send an email to %s,\n subject : %s,\n
content : %s\n on this day\n",email.emailAddress, email.subject,
email.content);
    }

    public static void printRecipientsWithBirthdays(ArrayList<Recipients>
recipientList, String[] date){
        String  month = date[1], day = date[2];
        System.out.printf("on %s/%s/%s\n", date[0],date[1], date[2]);
        boolean haveBirthday = false;
        for(Recipients recipients:recipientList){
            if(!(recipients instanceof Official)){
                if(recipients instanceof Office_friend){
                    Office_friend office_friend = (Office_friend) recipients;
                    String[] Birthday = office_friend.birthday.split("/");
                    if(Birthday[1].equals(date[1]) &&
Birthday[2].equals(date[2])){
                        System.out.printf("%s has
birthday\n",office_friend.name);
                        haveBirthday = true;
                    }
                }
                else {
                    Personal personal = (Personal) recipients;
                    String[] Birthday = personal.birthday.split("/");
                    if(Birthday[1].equals(date[1]) &&
Birthday[2].equals(date[2])){
                        System.out.printf("%s has birthday\n",personal.name);
                        haveBirthday = true;
                    }
                }
            }
        }
        if(!haveBirthday){
            System.out.println("No birthdays");
        }

    }
}
```

```
class EmailDetails {
    public static void findEmailDetail(ArrayList<Email> emailList, String[]
date) {
        String year = date[0], month = date[1], day = date[2];
        boolean noMail = true;
        for (Email email : emailList) {
            String dateSend = email.date.format(DateTimeFormatter.ISO_DATE);
            String[] month_and_Date_dateSend = dateSend.split("-");
            if (month_and_Date_dateSend[0].equals(year) &&
month_and_Date_dateSend[1].equals(month) &&
month_and_Date_dateSend[2].equals(day)) {
                Print.printDetails(email);
                noMail = false;
            }
        }
        if (noMail) {
            System.out.printf("No Mails on %s/%s/%s\n", year, month, day);
        }
    }
}
```