Project Report

Spring 2025

Name: Rakshith
Project: Contact List

Note: As you write each section, try to be as clear and detailed as possible. Your goal is to communicate your thought process and work clearly. Don't worry if you faced challenges or made mistakes; discussing these is a valuable part of learning and shows your problem-solving skills! Remember, there's no single 'right' way to do these tasks, so be creative and honest in your responses.

Problem Statement (2-3 Paragraphs):

This project builds a console based contact list manager. The goal is to store contacts with first name, last name, and birthday, then let a user add, view, delete, import, and export contacts. The program keeps the list readable by sorting contacts by last name and first name.

The user inputs menu choices, names, and birthdays. The program outputs confirmation messages, a formatted contact table, and file import or export results. It also outputs each contact’s age based on the birthday.

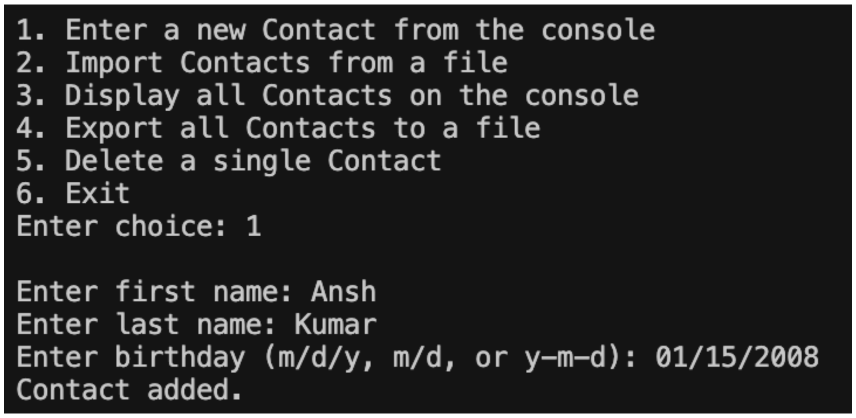
Several things can go wrong. The user can enter blank names, invalid dates, or a file can be missing or malformed. I handled name validation by rejecting empty first or last names. I also handled date errors by looping until parse_date returns a valid date.

Design (1-3 Paragraphs):

The program is divided into a Contact class file and a main driver file to keep responsibilities organized. A list stores Contact objects, and sorting by last name and first name maintains consistent order after adding or importing contacts. The csv module manages file operations, and datetime handles date parsing in multiple formats. The overall structure remains clear and efficient for the scope of the assignment, though age calculation could be improved by checking whether the birthday has occurred in the current year.

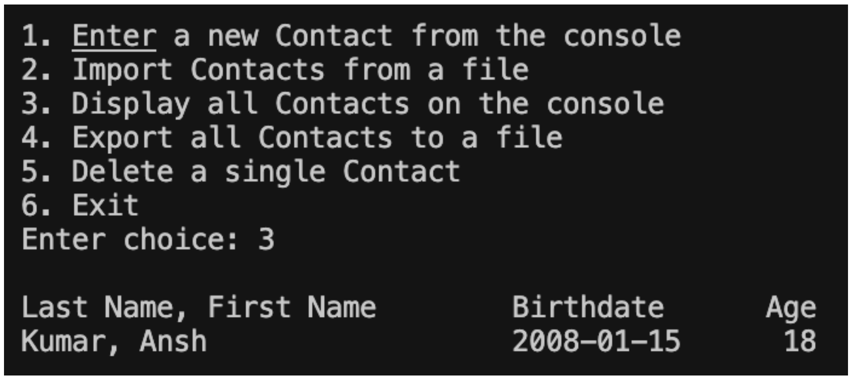
Testing (1-2 Paragraphs + screenshots of 3 test cases):

Testing included normal inputs, invalid date entries, missing files, and delete operations. Regular tests involved adding valid contacts and verifying correct sorted display. Edge cases included entering invalid dates and attempting to import when the file did not exist. In all tested cases, the program responded appropriately without crashing. Screenshots provided show successful add and display operations, date validation behavior, and both failed and successful import attempts.



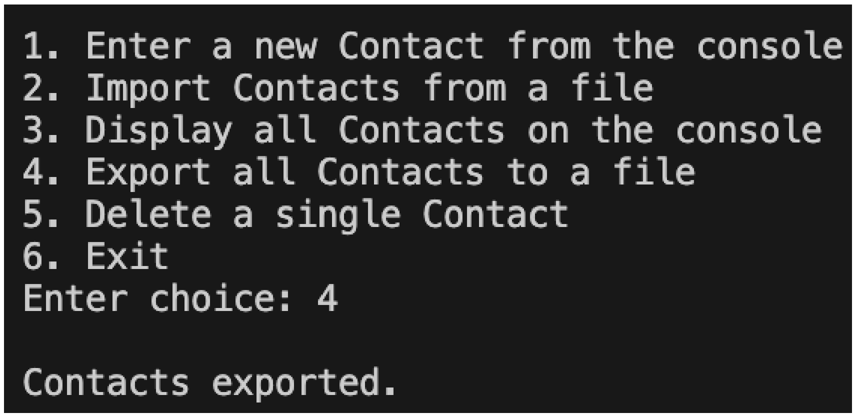
```
1. Enter a new Contact from the console
2. Import Contacts from a file
3. Display all Contacts on the console
4. Export all Contacts to a file
5. Delete a single Contact
6. Exit
Enter choice: 1

Enter first name: Ansh
Enter last name: Kumar
Enter birthday (m/d/y, m/d, or y-m-d): 01/15/2008
Contact added.
```



```
1. Enter a new Contact from the console
2. Import Contacts from a file
3. Display all Contacts on the console
4. Export all Contacts to a file
5. Delete a single Contact
6. Exit
Enter choice: 3

Last Name, First Name        Birthdate      Age
Kumar, Ansh                  2008-01-15      18
```



```
1. Enter a new Contact from the console
2. Import Contacts from a file
3. Display all Contacts on the console
4. Export all Contacts to a file
5. Delete a single Contact
6. Exit
Enter choice: 4

Contacts exported.
```

Conclusion (1 paragraph)

The project achieved its goal of creating a functional contact management system with validation and file support. This assignment strengthened understanding of classes, file handling, sorting, and exception handling. In future projects, improvements would include refining age calculation accuracy and providing more specific error messages, while maintaining the same organized structure.