Daniel Vega

CS420

Homework 5

# Description

My project’s business use case is to store people in a campus directory. There are two types of people: students and professors. There are basic features available such as listing, adding, searching, and sorting. Professors have an attribute to store the classes he/she is teaching. Students have an attribute to store the classes he/she is taking along with his/her major.

A file can be read to quickly store batches of people following a specific template.

# Features

## Listing People

The program is able to list people already stored in the directory. Each person’s full name is printed out as well as their unique attributes. Each person’s number is printed to let the user know how many people are already in the directory.

## Adding Students

To add a student, the user should enter his/her first name, last name, age, major, and classes currently taking. The user will be prompted for each individual class being taken. When the user is done entering classes, the user is asked to type “done” to let the program know the user is done.

## Adding Professors

To add a professor, the user is asked to enter the professor’s first name, last name, age, major, and classes currently teaching. Similar to adding students, each individual class the professor is teaching is prompted for. The user will type “done” when done entering classes.

## Search By Keyword

When searching, the user is prompted for a keyword to search by. When it is provided, the program will start searching for they keyword in every single person’s first name and last name. People who match the keyword are printed on the console for the user to view.

## Sort By Last Name

Depending on the ADT used, the user can sort the names by last name. For the BinaryTreeADT, there is no need to sort since every person is sorting upon inserting.

## Write To File

The user is able to write the contents of the directory to a file. This will write every person’s information to a file called “output.txt”.

# Implementation

The VegaCampusDirectoryApplication is the application class. This class will take the input to handle all of the features the application provides. It may use one of two ADT classes: CS420TeamListADT and DanielBinaryTreeADT. Each ADT has a different implementation for adding and viewing people. They both use the same API so changing from one to the other requires only one line of code to change.

There are also two ItemType classes: DanielStudentPersonType and DanielProfessorPersonType. These two classes inherit from DanielAbstractPersonType, which is an abstract class. Each PersonType has their own unique attributes based on their role in the campus.

In addition to taking input from the user, at the start, the application reads a file called “input.txt” that tells the application people to store. This can be used to quickly store multiple people at once.

# Pattern

The most abundant pattern is the Adapter pattern. It is present in the ADT classes as well as the PeopleType classes. Both ADT follows the BoundedStackInterface and have the same API to handle the request from the application class.

Each PersonType inherits from DanielAbstractPersonType and thus have very much the same API except for accessing their unique attributes.

The adapter pattern allows minimal change to the application class in order to use either of the two ADT classes.

# Input File

The pattern for the input file is as follows:  
Number – (1 or 2) – 1 means student, 2 means professor

First Name – First name of student or professor

Last Name – Last name of student or professor

Number – Age of student or professor

Major – only applies to students

List of classes taking or classes teaching – Each class is separated by a new line

. – This period serves a way to let the program know the information for person ends

Below is an example for a Student:

1

Daniel

Vega

23

Computer Science

CS420

CS408

CS422

.

Below is an example for a Professor:

2

Ada

Lovelace

45

CS400

CS404

CS500

.

# Sample Output

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

Enter an option

1 - List Directory

2 - Add Student

3 - Add Professor

4 - Search by keyword

5 - Sort by last name

6 - Write Team Members to File 'output.txt'

0 - Exit

> 1

Num: 1 - Professor - Full Name: Frida Kahlo - Age: 40

Teaching: Art 101, Humanities 400, Art 300,

Num: 2 - Professor - Full Name: Ada Lovelace - Age: 45

Teaching: CS400, CS404, CS500,

Num: 3 - Student - Full Name: Peter Parker - Age: 20 - Major: Mathematics

Taking: Math300, Math330, Physics345, Humanities,

Num: 4 - Student - Full Name: Steve Rogers - Age: 19 - Major: Humanities

Taking: Humanities 100, CS100, Math144, Physics 120,

Num: 5 - Professor - Full Name: Tony Stark - Age: 35

Teaching: CS450, CS440,

Num: 6 - Student - Full Name: Daniel Vega - Age: 23 - Major: Computer Science

Taking: CS420, CS408, CS422,

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

Enter an option

1 - List Directory

2 - Add Student

3 - Add Professor

4 - Search by keyword

5 - Sort by last name

6 - Write Team Members to File 'output.txt'

0 - Exit

> 2

Enter student's first name

> Gladis

Enter student's last name

> Alvares

Enter student's age

> 20

Enter student's major

>

Computer Science

Enter the class name Gladis is taking. Type 'done' if done entering classes

> CS 420

Enter the class name Gladis is taking. Type 'done' if done entering classes

> CS408

Enter the class name Gladis is taking. Type 'done' if done entering classes

> CS422

Enter the class name Gladis is taking. Type 'done' if done entering classes

> done

Student - Full Name: Gladis Alvares - Age: 20 - Major: Computer Science

Taking: CS 420, CS408, CS422,

Student Added!-----------------------------------

Enter an option

1 - List Directory

2 - Add Student

3 - Add Professor

4 - Search by keyword

5 - Sort by last name

6 - Write Team Members to File 'output.txt'

0 - Exit

> 1

Num: 1 - Student - Full Name: Gladis Alvares - Age: 20 - Major: Computer Science

Taking: CS 420, CS408, CS422,

Num: 2 - Professor - Full Name: Frida Kahlo - Age: 40

Teaching: Art 101, Humanities 400, Art 300,

Num: 3 - Professor - Full Name: Ada Lovelace - Age: 45

Teaching: CS400, CS404, CS500,

Num: 4 - Student - Full Name: Peter Parker - Age: 20 - Major: Mathematics

Taking: Math300, Math330, Physics345, Humanities,

Num: 5 - Student - Full Name: Steve Rogers - Age: 19 - Major: Humanities

Taking: Humanities 100, CS100, Math144, Physics 120,

Num: 6 - Professor - Full Name: Tony Stark - Age: 35

Teaching: CS450, CS440,

Num: 7 - Student - Full Name: Daniel Vega - Age: 23 - Major: Computer Science

Taking: CS420, CS408, CS422,

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

Enter an option

1 - List Directory

2 - Add Student

3 - Add Professor

4 - Search by keyword

5 - Sort by last name

6 - Write Team Members to File 'output.txt'

0 - Exit

> 4

Enter keyword to search by

> gladis

Student - Full Name: Gladis Alvares - Age: 20 - Major: Computer Science

Taking: CS 420, CS408, CS422,

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

Enter an option

1 - List Directory

2 - Add Student

3 - Add Professor

4 - Search by keyword

5 - Sort by last name

6 - Write Team Members to File 'output.txt'

0 - Exit

> 0

Goodbye!

# UML

Below is a UML diagram of the entire project. It details what classes are used and their relationship to each other

