README.md

Final Programming Assignment: Course Listing System

`COSC 211: Computer Science II` `August 2, 2022` `Soobin Rho`

What does this program do? The main purpose of this program, which is named `CourseListingSystem.java`, is to display all courses including who is teaching the course and which students are enrolled in it. In order to run it, type this into your terminal:

```
# Compile.
javac CourseListingSystem.java

# Run.
java CourseListingSystem
```

First, the program reads `courseList.txt`, `facultlyList.txt`, and `studentList.txt`. The program then prints each course information along with the corresponding faculty and students list. Here's how the output looks like:

```
Introduction to Hot Pepper | ALCH 201 A
Lectured by: Sweet Pineapple
MWF 8:00am - 8:50am

1. Cranberries Clementine (375172)
2. Violet Raspberries (609541)
3. Sour Quince (967588)
4. Soup Huge (105227)
...



Quantitative Grapefruit Kiwifruit | ENHA 133 I
Lectured by: Lemon Orange
T 6:00pm - 9:00pm

1. Melon Salt (373394)
2. Awesome Lemon (391334)
3. Cranberries Clementine (375172)
4. Loquat Soup (295099)
5. Banana Violet (768270)
6. Soup Huge (105227)
...
```

Additional Functions: This program also has additional functions in case you want to test the program without having to input your own data. These functions create sample data for you:

```
# 1. Create 20 example students.
#    Write to `studentList.txt`
javac StudentList.java
java StudentList

# Output
Pink Wonderful | 9556 S CLEMENTINE ST 11867 | 908272936 | 2021 2 9 | 274769 | 2039 2 3 | 2043 2 5 | 101 |
101 | 13
Strawberries Carambola | 9068 S SOUR ST 92891 | 368235721 | 2011 0 23 | 842367 | 2029 0 1 | 2033 0 2 | 46
| 23 | 16
Sweet Salt | 4622 S GRAPEFRUIT ST 49263 | 510996516 | 1995 5 21 | 550459 | 2013 5 5 | 2017 5 4 | 34 | 18 |
15
...


# 2. Create 5 example faculty members.
#    Write to `facultyList.txt`
javac FacultyList.java
java FacultyList

# Output
Black Muscular | 8596 S CARAMBOLA ST 52509 | 180945707 | 1993 10 9 | 533097 | 2027 10 4 | Acerola Wizard |
WIZA | 1891825
Amazing Pickles | 3823 S APRICOTS ST 64642 | 826891857 | 2005 1 25 | 375345 | 2039 1 6 | Loquat Wizard |
DIVI | 387550
Elderberries Chicken | 8755 S LUKEWARM ST 84529 | 696870949 | 2007 9 27 | 494751 | 2041 8 30 | Happy
Wizard | WITC | 8098919
...


# 3. Create 12 example courses.
#    Write to `courseList.txt`
javac CourseList.java
java CourseList

# Output
Computational Wonderful Pineapple | WITC 188 U | MWF 1:00pm - 1:50pm
Advanced Blueberries Strawberries | PIZZ 222 B | TR 8:30am - 9:45am
Programming for Hot Green | DIVI 199 H | MWF 2:00pm - 2:50pm
...


# 4. Create 10 example student employees.
#    Write to `staffList.txt`
javac StaffList.java
java StaffList

# Output
Pineapple Muscular | 7865 S VIOLET ST 18857 | 341486164 | 1991 5 3 | 222814 | 2010 5 3 | Green Assistant
Wizard | WIZA | false | 25.42
Amazing Kiwifruit | 6049 S JACKFRUIT ST 30722 | 716540301 | 2019 4 2 | 435878 | 2038 4 5 | Blueberries
Assistant Wizard | SPRI | false | 18.63
Muscular Caramel | 8043 S SOUR ST 15867 | 533296680 | 1999 2 20 | 129121 | 2018 2 1 | Hot Assistant Wizard
| WITC | false | 27.43
...
```

By the way, notice that every class has a `...Test.java` file: `CourseTest.java`, `FacultyTest.java`, `StudentTest.java`, `PersonTest.java`, `EmployeeTest.java`, `StaffTest.java`, and `StudentEmployeeTest.java`.

End-users can ignore all `...Test.java` files becuase they are just for debugging purposes. Basically, they just test and show examples of how to use setter and getter methods. For example, `StudentTest.java` shows how to set and get the attributes of the class *Student*, such as *Date expGradDate* and *int crTaken*.

Design Approach

Everything you need to know in order to run the program was shown above. This section is for those who want to get a better understanding of what happens behind the scene each time the program is run.

```
├── class-hierarchy.png
├── Course.java
├── CourseListingSystem.java    # The main program. End users
│                               # can ignore all the rest.
├── CourseListings.txt          # This is where the main program
│                               # stores its data.
├── CourseList.java
├── CourseTest.java
├── Employee.java
├── EmployeeTest.java
├── Faculty.java
├── FacultyList.java
├── FacultyTest.java
├── Person.java
├── PersonTest.java
├── RandomData.java
├── README.md
├── Staff.java
├── StaffList.java
├── StaffTest.java
├── Student.java
├── StudentList.java
└── StudentTest.java
```

The first thing CourseListingSystem does when it runs is to check if `CourseListingSystem.txt` exists or not. If this data file doesn't already exist, the program creates sample data for you so that you can quickly test the program without having to input anything in.

What the program does with `CourseListingSystem.txt` is to first read each line and parse everything in the file and then convert them into Java variables of the class CourseListingSystem. For example, here's an example of `CourseListingSystem.txt`:

```
COSC 101 A | 100432 | 698635 385655 190274 162478
MAGI 322 A | 122805 | 753947 436291 401901
```

The first string is the course code. The program parses *COSC 101 A* into three variables. *COSC* gets assigned to `String dept`, *101* gets assigned to `int number`, and *A* gets assigned to `String section`. These three variables are then used as a reference point for finding more information about the course. This is possible because detailed course informationm such as the course schedule is all stored in `courseList.txt`. So, the program uses `dept`, `number`, and `section` to retrieve more information about the course from `courseList.txt` and then displays the results.

Likewise, the second string is for the professor. This string is the faculty ID, which is first converted into `String facultyID` and then used by the program to find more information about the professor from `facultyList.txt`.

The last string is a list of the students enrolled in the course. Just like the second string which is the faculty ID, the last string is a list of student ID's, which are internally parsed into `String studentID` and then used for retrieving detailed student information from `studentList.txt`.

How This Documentation was Made

This documentation was first written as a markdown file. Then, it was exported to a GitHub flavored html file with *grip*:

```
# Install grip.
pip3 install grip

# Install wkhtmltopdf.
sudo dnf -y install wkhtmltopdf

# Convert README.md to html and then
# host to http://localhost:6419
grip README.md 6419

# Export the html to a pdf file.
wkhtmltopdf http://localhost:6419 README.pdf
```

Thank you, Professor Steinwand! We've learned a lot about Java thanks to you, and I'm looking forward to learn C++ with you next semester. Have a great rest of the summer!

--- End of Assignment ---