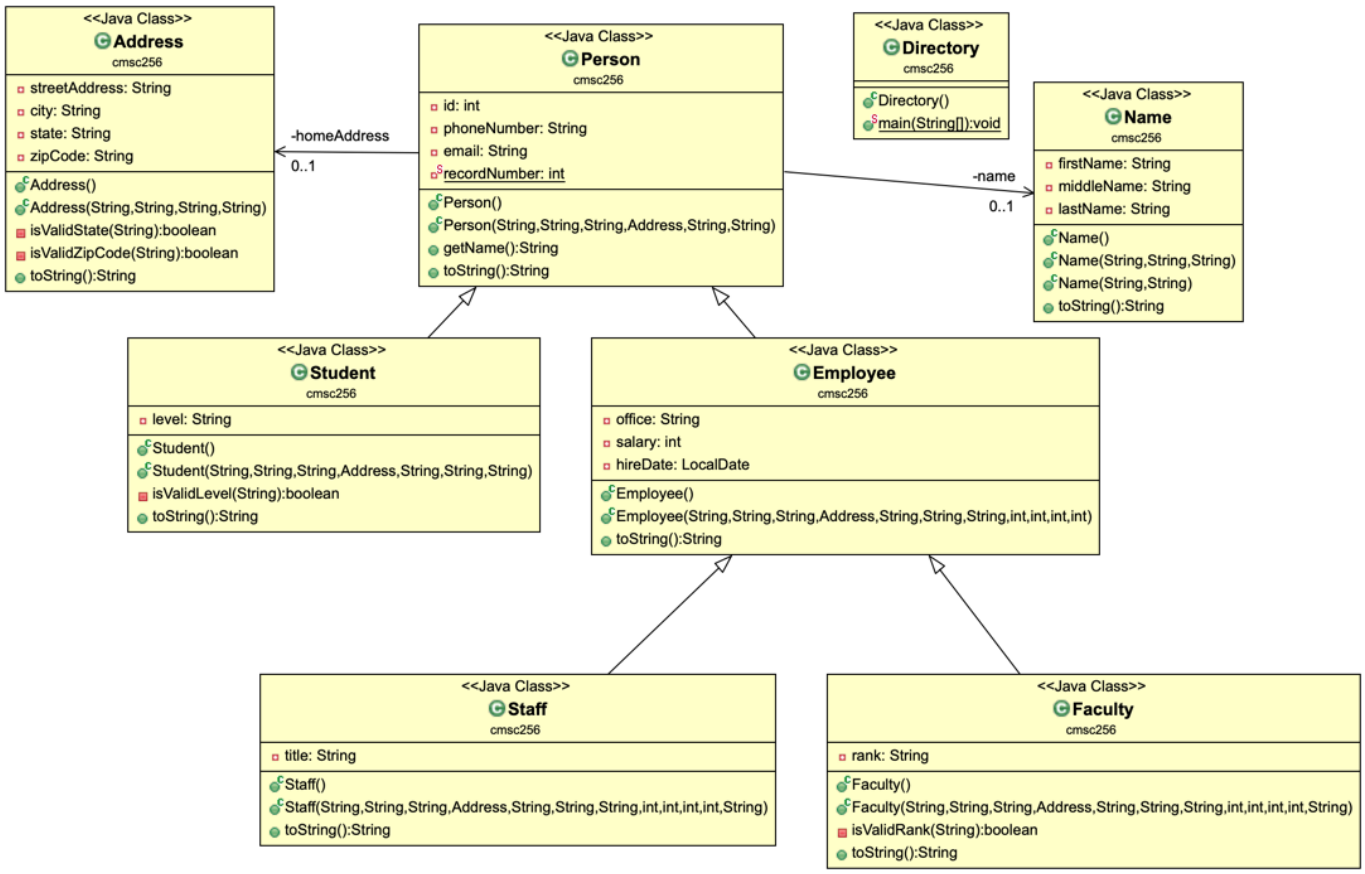
Programming Project 1 – Inheritance Point Value – 100 points

Note: When you turn in an assignment to be graded in this class, you are making the claim that you neither gave nor received assistance on the work you turned in (except, of course, assistance from the instructor or teaching assistants).

Learning Objectives:

- Implement multiple Java classes using inheritance and composition using a UML diagram to guide the program design.
- Demonstrate appropriate documentation of code using class-level and method-level comments.
- Use validation to maintain data integrity at the class level.

The classes needed are shown in the UML class diagram and listed. All classes are required to be in the same package with the package name **`cmsc256`**.



`Name`, **`Address`** and **`Directory`** classes are provided. **`Directory`** is the driving class for the program. Do not make any changes in **`Name`** and **`Address`** classes. If you make any changes to the **`Name`** or **`Address`** class, your program will not compile and execute correctly in Gradescope.

You are to implement the following classes:

- Person Class: The class variables and types are shown in the class diagram. Specifically, **`RecordNumber`** is a static variable. It will increase each time you instantiate a new **`Person`**

object and will be assigned to the **id** variable of the currently created instance. Here is the signature for the constructor of **Person** class:

```
public Person(String first, String middle, String last, Address homeAddress,
String phoneNumber, String email)
```

- Student Class: **Student** class extends Person class. It has extra variable, `level`, which indicates the student level as one of the following, Freshman, Sophomore, Junior, Senior, Graduate. Constructors of the **Student** class initialize the **level** variable in addition to all variables of super class.
- Employee Class: **Employee** class extends **Person** class. It has extra **office**, **salary** and **dateHired** variables. Constructors of the **Employee** class initialize all its variables.
- Faculty Class: **Faculty** class extends **Employee** class. It has extra variable, `rank`, which can either be Adjunct, Instructor, Assistant Professor, Associate Professor, or Professor. Constructors of the Faculty class initialize all its variables.
- Staff Class: **Staff** class extends **Employee** class. It has extra variable which is a String called `title`. Constructors of the **Staff** class initialize all its variables.

You are provided with the driver program called `Directory` to help guide the development of your classes. The correct output from executing the `Directory` program should look like this:

```
Student:
---------------------------------------
Peter Luong
---------------------------------------
Home Address: 10 FakeHome Street
Staged City, VA  22222
Phone Number: 858-789-8575
Email Address: fake0@ mymail.vcu.edu
ID: 1
Student Level:  Freshman

Student:
---------------------------------------
Alice Myers
---------------------------------------
Home Address: 10 FakeHome Street
Staged City, VA  22222
Phone Number: 858-789-8576
Email Address: fake1@ mymail.vcu.edu
ID: 2
Student Level:  Senior

Faculty:
---------------------------------------
Maria L Garcia
---------------------------------------
Home Address: 401 West Main Street
Richmond, VA  23284-3019
```

```
Phone Number: 804-828-8577
Email Address: fake2@vcu.edu
ID: 3
Office:  323KL
Salary:  $60000
Date Hired:  10/10/2010
Rank:  Assistant Professor

Faculty:
---------------------------------------
Michael Junior Cook
---------------------------------------
Home Address: 401 West Main Street
Richmond, VA  23284-3019
Phone Number: 804-789-8578
Email Address: fake3@vcu.edu
ID: 4
Office:  356JL
Salary:  $50000
Date Hired:  5/5/2015
Rank:  Instructor

Staff:
---------------------------------------
Shane Dave
---------------------------------------
Home Address: 20 Staged Home Road
Dummy, VA  33333-1234
Phone Number: 858-789-8579
Email Address: fake4@vcu.edu
ID: 5
Office:  356JL
Salary:  $40000
Date Hired:  8/8/2008
Title:  Department Secretary
```

Make sure your classes are well documented - including a comment block header in all source code files with your name, the course and section numbers, the project name, the program purpose. Be sure to include appropriate comments throughout your code, choose meaningful identifiers, and use indentation as shown in your textbook and in class.

Submit your source code files, **`Person.java`**, **`Student.java`**, **`Employee.java`**, **`Staff.java`**, and **`Faculty.java`** to the assignment link in Gradescope by midnight on the due date. You have a maximum of three submission attempts for this project. Additional attempts will not be allowed, so test your files thoroughly before uploading to Gradescope.

Grading Rubric

Person class:

Instance data members are correctly declared (2 pts.) _____

Default constructor included with appropriate assignment (2 pts.) _____

Parameterized constructor correctly written (5 pts.) _____
Correct use of a static class variable to generate ID numbers (5 pts) _____
getName() method written correctly (1 pts.) _____
toString() method written correctly (5 pts.) _____
Total (20 pts.) _____

Student class:

Instance data member is correctly declared (2 pts.) _____
Default constructor included with appropriate assignment (3 pts.) _____
Parameterized constructor correctly written (5 pts.) _____
toString() method written correctly (5 pts.) _____
Class design includes appropriate code to uphold encapsulation (5 pts.) _____
Total (20 pts.) _____

Employee class:

Instance data member is correctly declared (2 pts.) _____
Default constructor included with appropriate assignment (3 pts.) _____
Parameterized constructor correctly written (5 pts.) _____
toString() method written correctly (5 pts.) _____
Total (15 pts.) _____

Faculty class:

Instance data member is correctly declared (2 pts.) _____
Default constructor included with appropriate assignment (3 pts.) _____
Parameterized constructor correctly written (5 pts.) _____
toString() method written correctly (5 pts.) _____
Class design includes appropriate code to uphold encapsulation (5 pts.) _____
Total (20 pts.) _____

Staff class:

Instance data member is correctly declared (2 pts.) _____
Default constructor included with appropriate assignment (3 pts.) _____
Parameterized constructor correctly written (5 pts.) _____
toString() method written correctly (5 pts.) _____
Total (15 pts.) _____

All classes:

Files named, commented appropriately, and submitted as specified (5 pts.) _____
Java syntax is correct (class names starting with capital letter, variable names starting with small letter, indentations, correct packages etc) (5 pts.) _____

Project Total (100 pts.) _____