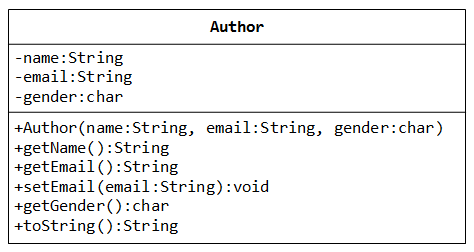
**Name:**

**Advanced Programming in Java**

**Lab Exercise 12.21.2023**

Exercise: The Author and Book Classes



Author Class UML Diagram

A class called Author is designed as shown in the class diagram. It contains:

Three private instance variables:

* name (String),
* email (String),
* gender (char of either 'm' or 'f');

One constructor to initialize the name, email and gender with the given values;

public Author (String name, String email, char gender) {......}

(There is no default constructor for Author, as there are no defaults for name, email and gender.)

public getters/setters: getName(), getEmail(), setEmail(), and getGender();  
(There are no setters for name and gender, as these attributes cannot be changed.)

A toString() method that returns "author-name at email", e.g., "Mary Jones at mjones@somewhere.com".

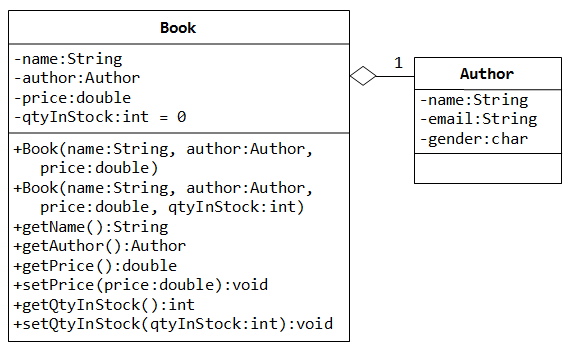
Write the Author class. Also write a test program called TestAuthor to test the constructor and public methods. Try changing the email of an author, e.g.,

Author anAuthor = new Author("Mary Jones", "mjones@somewhere.com", 'f');

System.out.println(anAuthor); // call toString()

anAuthor.setEmail("mjones@nowhere.com")

System.out.println(anAuthor);



Book Class UML Diagram

A class called Book is designed as shown in the class diagram. It contains:

Four private instance variables:

* name (String)
* author (of the class Author you have just created, each book has one and only one author)
* price (double)
* qtyInStock (int);

Two constructors:

* public Book (String name, Author author, double price) {...}
* public Book (String name, Author author, double price, int qtyInStock) {...}

public methods

getName()

getAuthor()

getPrice()

setPrice()

getQtyInStock()

setQtyInStock().

toString() that returns "'book-name' by author-name at email".  
(Take note that the Author's toString() method returns "*author-name at email*".)

Write the class Book (which uses the Author class written earlier). Also write a test program called TestBook to test the constructor and public methods in the class Book. Take Note that you have to construct an instance of Author before you can construct an instance of Book. E.g.,

Author anAuthor = new Author(......);

Book aBook = new Book("Java for Experts", anAuthor, 19.95, 1000);

Take note that both Book and Author classes have a variable called name. However, it can be differentiated via the referencing instance. For a Book instance says aBook, aBook.name refers to the name of the book; whereas for an Author's instance say anAuthor, anAuthor.name refers to the name of the author. There is no need (and not recommended) to call the variables bookName and authorName.