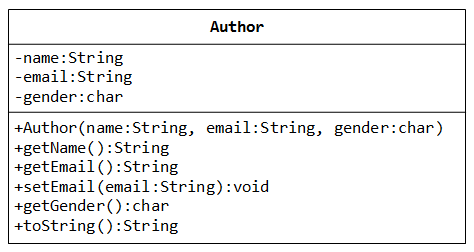
**Name:**

**Advanced Programming in Java**

**Lab Exercise 1.6.2025**

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".

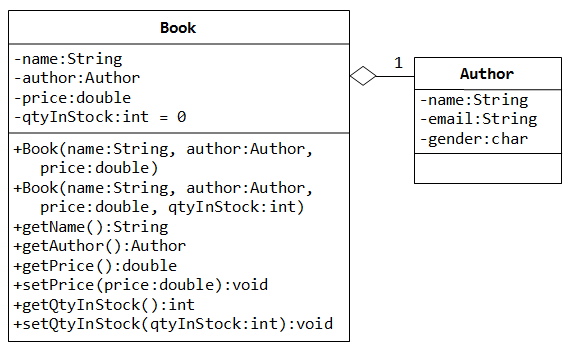
Write the Author class. Also write a test program called TestAuthor to test the constructor and public methods. Try changing the email of an author, e.g.,

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(......);

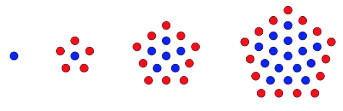
Book aBook = new Book("Java for Experts", anAuthor, 19.95, 1000);

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.

**Pentagonal Number**

Write a function that takes a positive integer num and calculates how many dots exist in a pentagonal shape around the center dot on the *Nth* iteration.

In the image below you can see the first iteration is only a single dot. On the second, there are 6 dots. On the third, there are 16 dots, and on the fourth there are 31 dots.



Return the number of dots that exist in the whole pentagon on the *Nth* iteration.

Examples

pentagonal(1) ➞ 1

pentagonal(2) ➞ 6

pentagonal(3) ➞ 16

pentagonal(8) ➞ 141

**Find the Missing Number**

Create a method that takes an array of integers between 1 and 10 (excluding one number) and returns the missing number.

Examples

missingNum([1, 2, 3, 4, 6, 7, 8, 9, 10]) ➞ 5

missingNum([7, 2, 3, 6, 5, 9, 1, 4, 8]) ➞ 10

missingNum([10, 5, 1, 2, 4, 6, 8, 3, 9]) ➞ 7

Notes

The array of numbers will be unsorted (not in order).

Only one number will be missing.