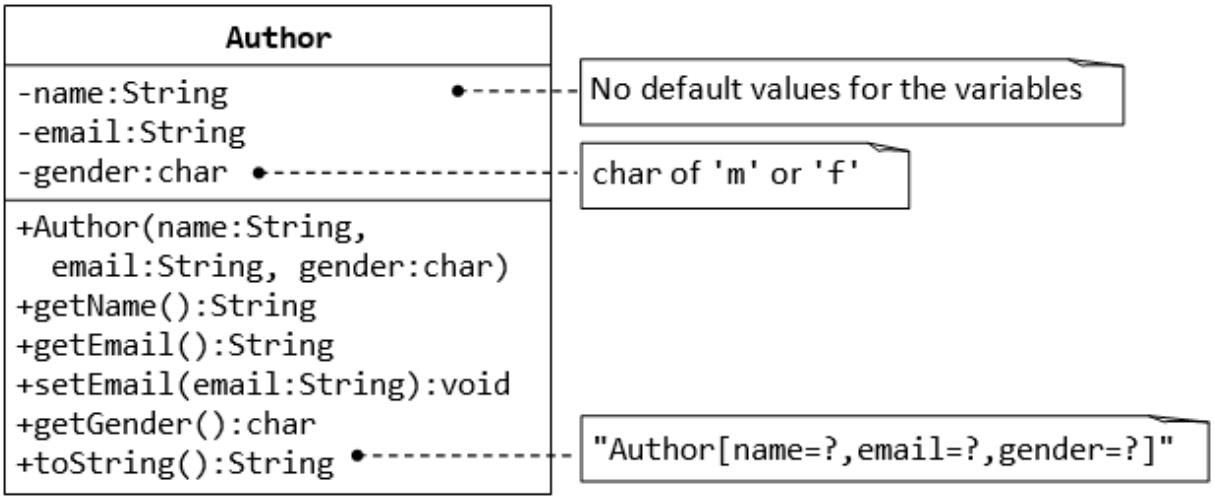
Lab 05: Object Oriented programming

Objectives:

- Building Blocks
- The student will learn how to transfer UML diagrams and their relations to Java code.

I. Exercise 1 (50 min):



A class called **Author** (as shown in the class diagram) is designed to model a book's author. It contains:

- Three private instance variables: name (String), email (String), and 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) {......}
```

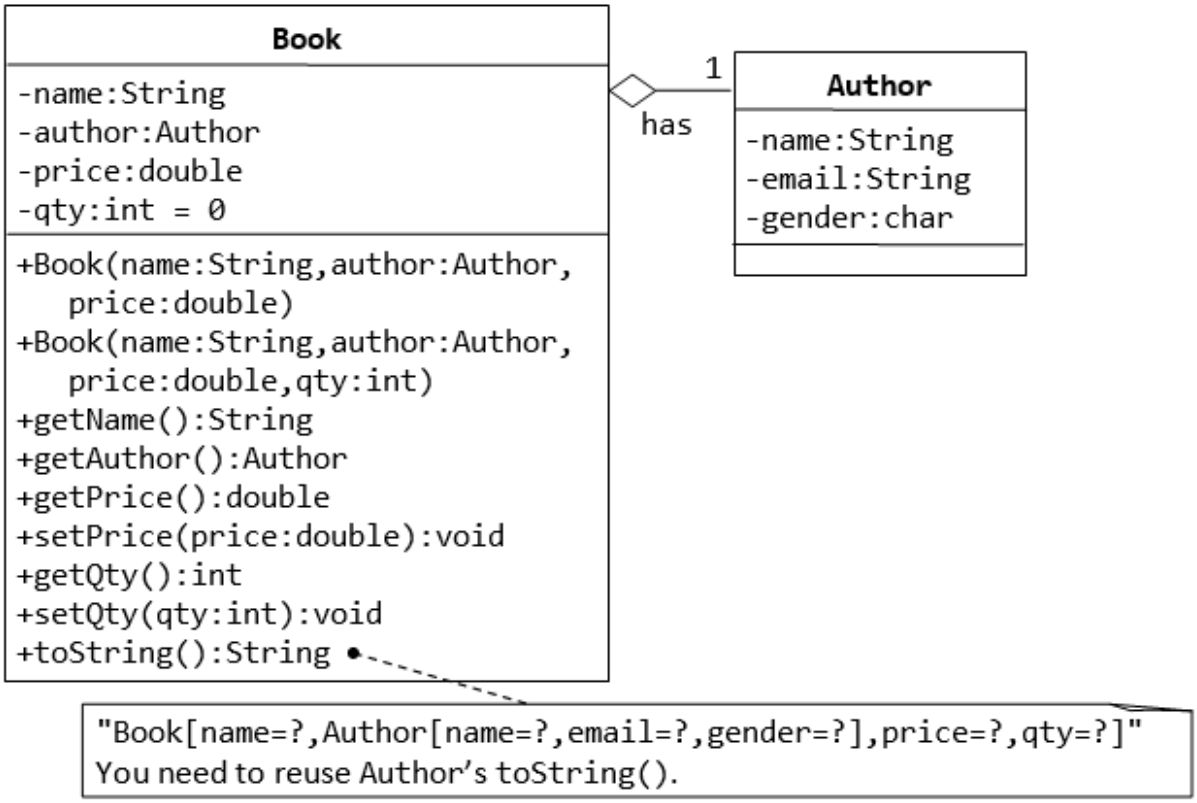
(There is no default constructor for Author, as there are no defaults for name, email and gender.)

- 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 ,email , and gender.
- Write the Author class. Also write a *test class* called TestAuthor to test all the public methods.

II. Exercice 2 (50 min):



A class called **Book** is designed (as shown in the class diagram) to model a book written by *one* author. It contains:

- Four private instance variables: name (String), author (of the class Author you have just created, assume that a book has one and only one author), price (double), and qty (int);
- Two constructors:

```
public Book (String name, Author author, double price) { .... }

public Book (String name, Author author, double price, int qty)
{ ..... }
```

- Public methods getName(), getAuthor(), getPrice(), setPrice(), getQty(), setQty().
- A toString() that returns Book name ,Author name ,email ,gender ,price ,qty . You should reuse Author's toString().

Write the Book class (**which uses the Author class written earlier**). Also write a test class called TestBook to test all the public methods in the class Book.

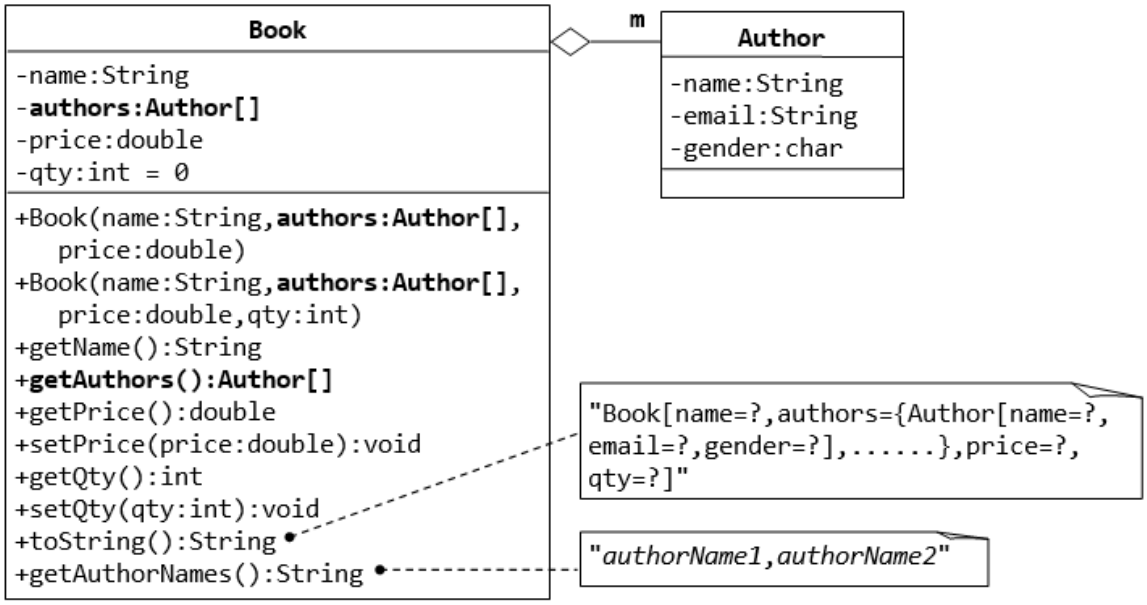
Hints:

- Take Note that you have to construct an instance of Author before you can construct an instance of Book.
- 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 auAuthor, anAuthor.name refers to the name of the author. There is no need (and not recommended) to call the variables bookName and authorName.

TRY:

1. Printing the name and email of the author from a Book instance. (**Hint:** aBook.getAuthor().getName(), aBook.getAuthor().getEmail()).
2. Introduce new methods called getAuthorName(), getAuthorEmail(), getAuthorGender() in the Book class to return the name, email and gender of the author of the book.

III. Exercise 3:



In the earlier exercise, a book is written by one and only one author. A book can be written by one or more author. Modify the Book class to support one or more authors by changing the instance variable authors to an Author array.

Notes:

- The constructors take an array of Author (i.e., Author[]), instead of an Author instance. In this design, once a Book instance is constructor, you cannot add or remove author.
- The toString() method shall return Book name ,Author name , Author email ,Author gender ,Book price ,Book qty .

You are required to:

1. Modify the code for the Book class to support multiple Authors . You shall re-use the Author class written earlier.
2. Write a test driver (called TestBook) to test the Book class.