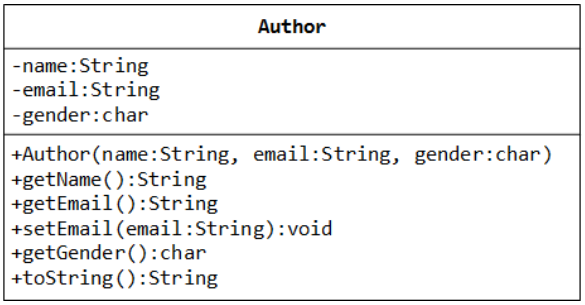
**UNIVERSITY OF RUHUNA**
**BACHELOR OF COMPUTER SCIENCE (BCS) (GENERAL) DEGREE**
**LEVEL II (SEMESTER I)**
**Laboratory Assignment 04**

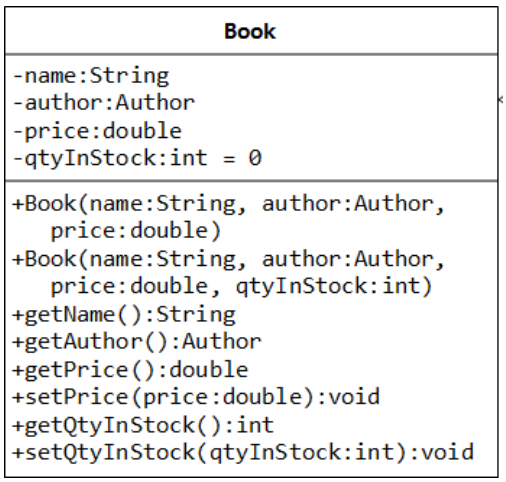
**CSC 2123 – Object Oriented Programming** **TIME:3 Hours**

1. A class called Author is designed as shown in the class diagram. 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();**
A **toString() method that returns "author-name (gender) + email"**
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




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, assume that each book has one and only one author), price (double), and 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 (gender) at email".**



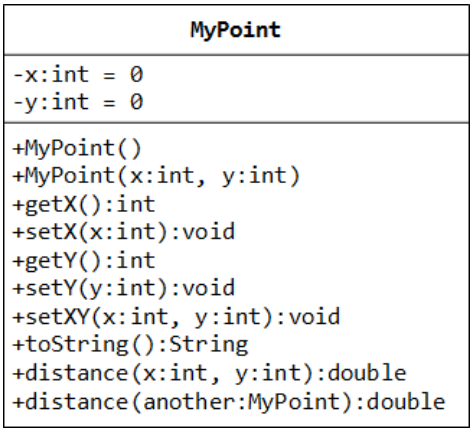
2. In the earlier exercise, a book is written by one and only one author. In reality, a book can be written by one or more author. Modify the Book class to support one or more authors by changing the instance variable authors to an Author array. Reuse the Author class written earlier.

   Notes:

   The constructors take an array of Author (i.e., Author[]), instead of an Author instance.
   The **toString()** method shall return "book-name by n authors", where n is the number of authors. A new method **printAuthors()** to print the names of all the authors.

   You are required to:
   Write the code for the Book class. You shall re-use the Author class written earlier.
   Write a test program (called TestBook) to test the Book class.

3. A class called `MyPoint`, which models a 2D point with `x` and `y` coordinates, is designed as shown in the class diagram. Implement class MyPoint.

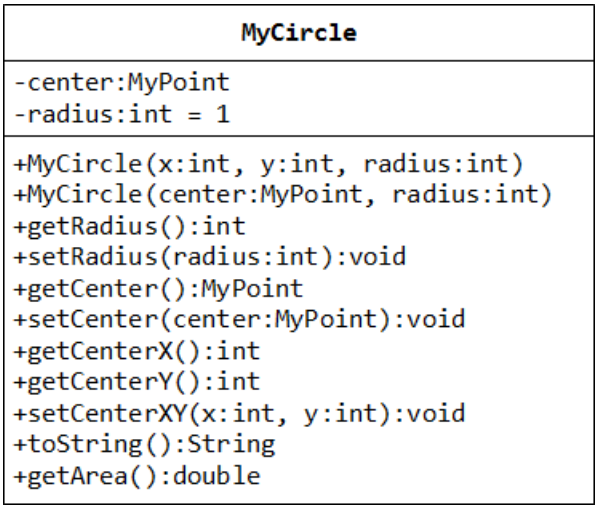



   Write a test class to check the constructors and methods of MyPoint.

4. A class called MyCircle, which models a circle with a center (x, y) and a radius, is designed as shown in the class diagram. The MyCircle class uses an instance of MyPoint class (created in the previous exercise) as its center.

   The class contains:

   - Two private instance variables: **center (an instance of MyPoint) and radius (int).**
   - A constructor constructs a circle with the given center's (x, y) and radius.
   - An overloaded constructor that constructs a MyCircle given a MyPoint instance as center, and radius. Various getters and setters.
   - A **toString()** method that returns a string description of this instance in the format "Circle @ (x, y) radius=r".
   - A **getArea()** method that returns the area of the circle in double.

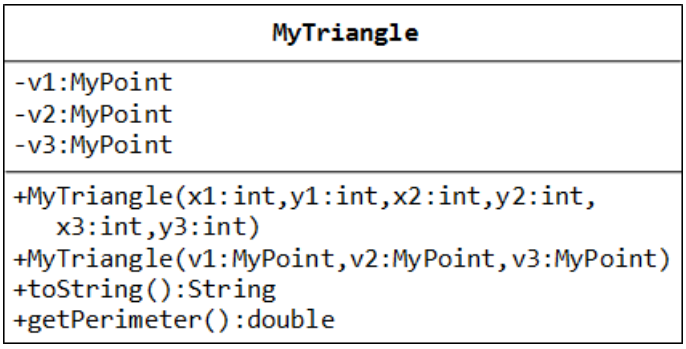



Write the MyCircle class. Also write a test program (called TestMyCircle) to test all the methods defined in the class.

5. A class called MyTriangle, which models a triangle with 3 vertices, is designed as follows. The MyTriangle class uses three MyPoint instances (created in the earlier exercise) as the three vertices.

   The class contains:

   - Three private instance variables v1, v2, v3 (instances of MyPoint), for the three vertices.
   - A constructor that constructs a MyTriangle with three points v1=(x1, y1), v2=(x2, y2), v3=(x3, y3).
   - An overloaded constructor that constructs a MyTriangle given three instances of MyPoint.
   - A **toString()** method that returns a string description of the instance in the format "Triangle @ (x1, y1), (x2, y2), (x3, y3)".
   - A **getPerimeter()** method that returns the length of the perimeter in double. You should use the distance() method of MyPoint to compute the perimeter.
   - A method printType(), which prints "equilateral" if all the three sides are equal, "isosceles" if any two of the three sides are equal, or "scalene" if the three sides are different.



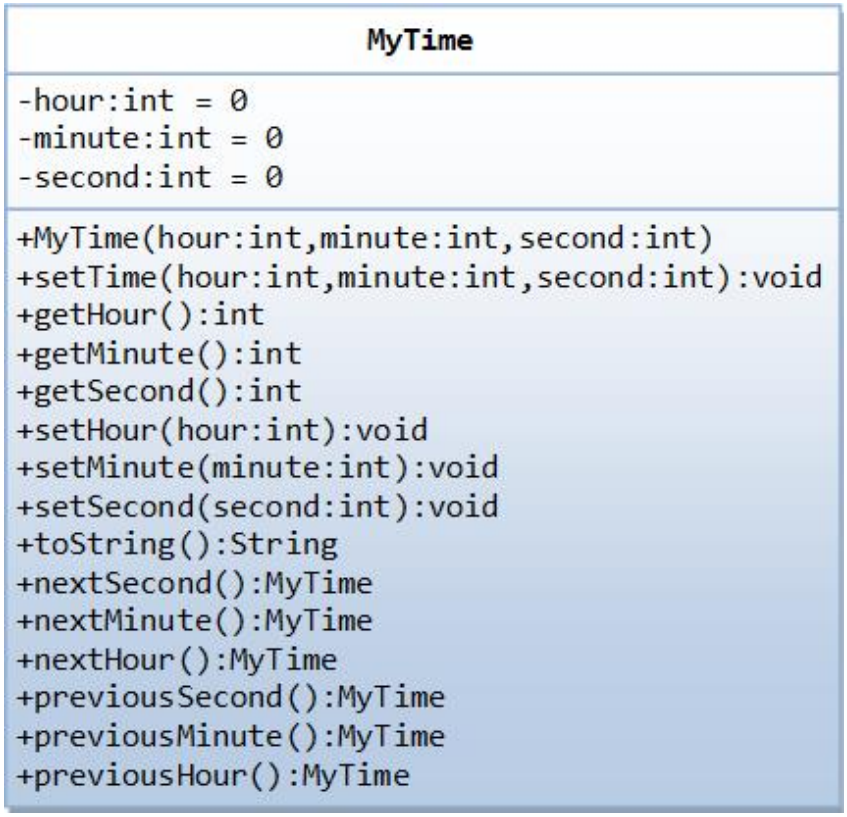

   Write the MyTriangle class. Also write a test program (called TestMyTriangle) to test all the methods defined in the class.

6. A class called MyTime, which models a time instance, is designed as shown in the class diagram.

   It contains the following private instance variables:

   **hour: between 0 to 23.**
   **minute: between 0 to 59.**
   **Second: between 0 to 59.**

   The constructor shall invoke the setTime() method (to be described later) to set the instance variable.




It contains the following public methods:

- **setTime(int hour, int minute, int second):** It shall check if the given hour, minute and second are valid before setting the instance variables. (Advanced: Otherwise, it shall print the message "Invalid hour, minute, or second!" and stop the execution)
- Setters **setHour(int hour), setMinute(int minute), setSecond(int second):** It shall check if the parameters are valid, similar to the above.
- Getters **getHour(), getMinute(), getSecond(). toString(): returns "HH:MM:SS".**
- **nextSecond():** Update this instance to the next second and return this instance. Take note that the nextSecond() of 23:59:59 is 00:00:00.
- **nextMinute(), nextHour(), previousSecond(), previousMinute(), previousHour():** similar to the above.

Write the code for the MyTime class. Also write a test program (called TestMyTime) to test all the methods defined in the MyTime class.

**********