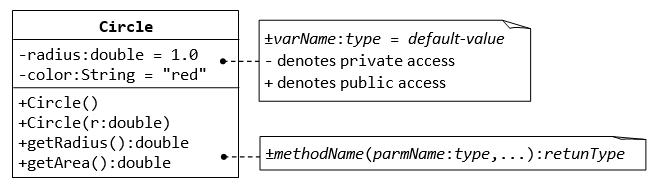
**Assignment Sheet #4**

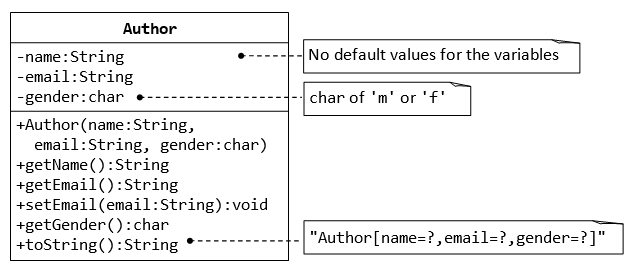
You need to write the class as well as the test class (ex, #1 you need Circle.java and TestCircle.java)

This first exercise shall lead you through all the *basic concepts* in OOP.



A class called **circle** is designed as shown in the following class diagram. It contains:

* Two private instance variables: radius (of the type double) and color (of the type String), with default value of 1.0 and "red", respectively.
* Two *overloaded* constructors - a *default* constructor with no argument, and a constructor which takes a double argument for radius.
* Two public methods: getRadius() and getArea(), which return the radius and area of this instance, respectively.



char of 'm', 'f', or 'o'

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', 'f', or 'o');
* 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=?,gender=?]", e.g., "Author[name=Tan Ah Teck,email=ahTeck@somewhere.com,gender=m]".

Write the Author class. Also write a *test driver* called TestAuthor to test all the public methods, e.g.,

Author ahTeck = new Author("Tan Ah Teck", "ahteck@nowhere.com", 'm'); // Test the constructor

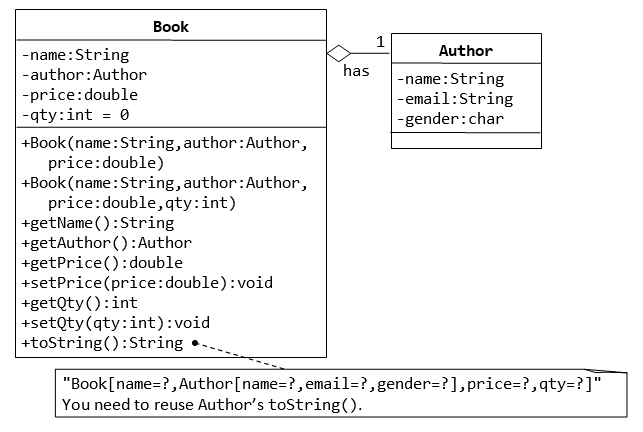
System.out.println(ahTeck); // Test toString()

ahTeck.setEmail("paulTan@nowhere.com"); // Test setter

System.out.println("name is: " + ahTeck.getName()); // Test getter

System.out.println("eamil is: " + ahTeck.getEmail()); // Test getter

System.out.println("gender is: " + ahTeck.getGender()); // Test gExerciseOOP\_MyPolynomial.pngetter



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 driver called TestBook to test all the 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.,

// Construct an author instance

Author ahTeck = new Author("Tan Ah Teck", "ahteck@nowhere.com", 'm');

System.out.println(ahTeck); // Author's toString()

Book dummyBook = new Book("Java for dummy", ahTeck, 19.95, 99); // Test Book's Constructor

System.out.println(dummyBook); // Test Book's toString()

// Test Getters and Setters

dummyBook.setPrice(29.95);

dummyBook.setQty(28);

System.out.println("name is: " + dummyBook.getName());

System.out.println("price is: " + dummyBook.getPrice());

System.out.println("qty is: " + dummyBook.getQty());

System.out.println("Author is: " + dummyBook.getAuthor()); // Author's toString()

System.out.println("Author's name is: " + dummyBook.getAuthor().getName());

System.out.println("Author's email is: " + dummyBook.getAuthor().getEmail());

// Use an anonymous instance of Author to construct a Book instance

Book anotherBook = new Book("more Java",

new Author("Paul Tan", "paul@somewhere.com", 'm'), 29.95);

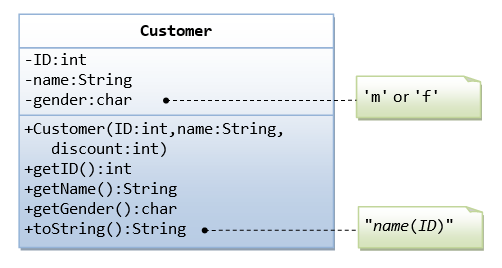
System.out.println(anotherBook); // toString()

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 say 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.

TRY:

1. Printing the name and email of the author from a Book instance. (Hint: aBook.getAuthor().getName(), aBook.getAuthor().getEmail()).

**The Customer and Account classes**



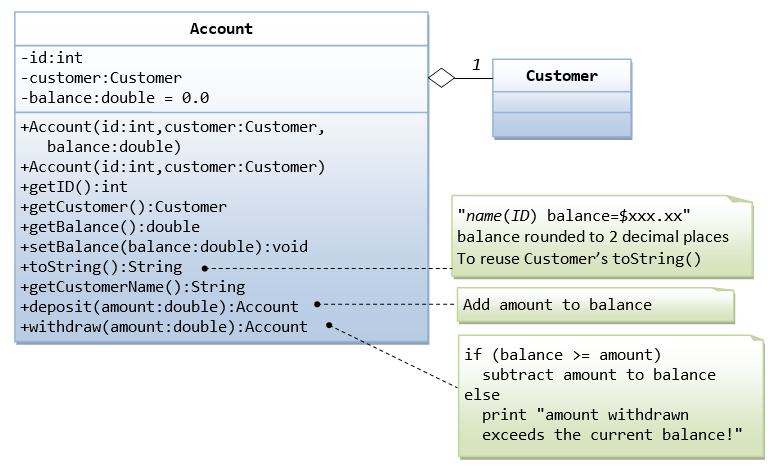
The Customer class models a customer is design as shown

in the class diagram. Write the codes for the Customer

 class and a test driver to test all the public methods.

‘m’ or ‘f’ or ‘o’

Gender:char)



double

double

The Account class models a bank account, design as shown in the class diagram, composes a Customer instance (written earlier) as its member. Write the codes for the Account class and a test driver to test all the public methods.

For deposit and withdraw functions set the return type to double.

**Array List Question**

Complete the classes shown in the following diagrams.

|  |
| --- |
| **Course** |
| -name: String  -grade: int  -totalEnrolled: int = 0 |
| +Course(name:String, grade:int, totalEnrolled:int)  +Course(name:String, grade:int)  +getName():String  +getGrade():int  +getTotalEnrolled():int  +setGrade(grade: int):void  +setTotalEnrolled(totalEnrolled: int):void  +toString():String |

|  |
| --- |
| **Student** |
| -name:String  -phoneNumber:String  -email:String  -enrolledList:ArrayList<Course> |
| +Student(name:String, phoneNumber:String, email:String)  +getName():String  +setName(name:String):void  +getPhoneNumber():String  +setPhoneNumber(phoneNumber:String):void  +getEmail():String  +setEmail(email:String):void  +enrollStudent(courseIn:Course):void  +printCourses():void  +toString():String |

The enrolledList variable will store all the courses the person is currently enrolled in and the printCourses() method will print a list of all the courses the person is currently enrolled in. Think of your school timetable showing your courses for the semester.

You will need to create a TestStudent class that will test all of the methods in both the Student and Course class.