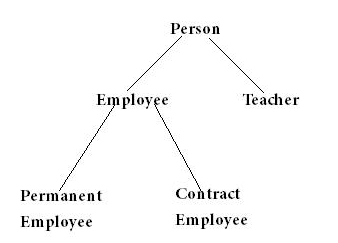
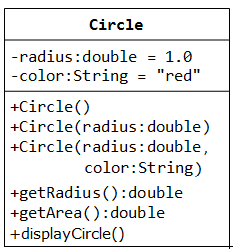
### Day 3

### 1. write java program to implement Inheritance with following example:

Person will have name and age as data members. Teacher and employee will inherit data members in the super class and create its own method myProfession() to display their profession. Then create objects of Teacher, Permanent, and Contract employee to display their profession..



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

* Two private instance variables: radius (of type double) and color (of type String), with default value of 1.0 and "red", respectively.
* Three overloaded constructors;
* **this** keyword can be used by one constructor to explicitly invoke another constructor in the same class
* Two public methods: getRadius() and getArea().
* create three circles instances by using given three constructors and display them.

3. 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,

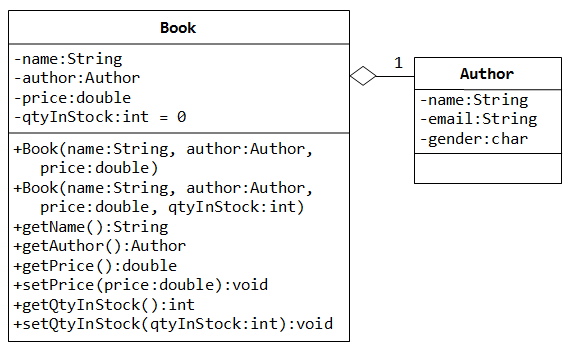
int qtyInStock) {...}

* public methods getName(), getAuthor(), getPrice(), setPrice(), getQtyInStock(),setQtyInStock().
* toString() that returns "*'book-name' by author-name (gender) at email*".  
  (Take note that the Author's toString() method returns "*author-name (gender) 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.

Note : 1. you have to construct an instance ofAuthor before you can construct an instance of Book.

2. 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.



1. create two instances of book and print 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.