Student Name: _____ **Roll No:** _____ **Section:** _____

Lab No. 02

Lab 02 – Class diagram with their notation and relationships

Class Diagram

- Class diagrams are visual representations of the static structure and composition of a particular system using the conventions set by the Unified Modeling Language (UML).
- System designers use class diagrams as a way of simplifying how objects in a system interact with each other.
- Using class diagrams, it is easier to describe all the classes, packages, and interfaces that constitute a system and how these components are interrelated.
- Since class diagrams are used for many different purposes, such as making stakeholders aware of requirements to highlighting your detailed design, you need to apply a different style in each circumstance

Example

- Simple class diagram may be used to show how an organization such as a convenient store chain is
set up.
- Precisely detailed class diagrams can readily be used as the primary reference for translating the
designed system into a programming code.

Notation of Class Diagram

1. **Class**

- An object is any person, place, thing, concept, event, screen, or report applicable to your system.
Objects both know things (they have attributes) and they do things (they have methods).
- A class is a representation of an object and, in many ways, it is simply a template from which objects are created.
- Classes form the main building blocks of an object-oriented application.

Example
Although thousands of students attend the university, you would only model one class, called Student, which would represent the entire collection of students.

Student Name: _____ Roll No: _____ Section: _____

2. Attributes

An attribute of a class represents a characteristic of a class that is of interest for the user.

The full format of the attribute text notation is:
Visibility name: type multiplicity = default [property-string]

3. Operations

A UML operation is a declaration, with a name, parameters, return type, exceptions list, and possibly a set of constraints of pre and post conditions. But, it isn't an implementation – rather, methods are implementation.

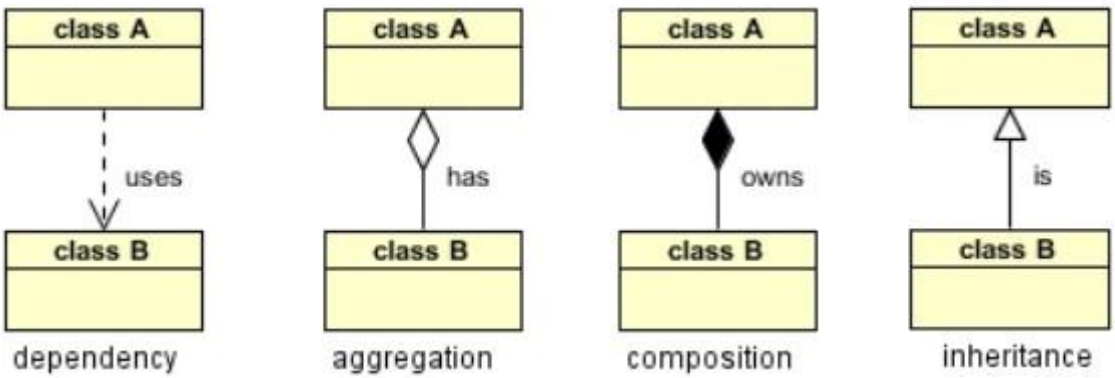
4. Visibility:

Use visibility markers to signify who can access the information contained within a class.

- Public +
- Private -
- Protected #
- Package ~

Relationship

Dependency: class A uses class B
Aggregation: class A has a class B
Composition: class A owns a class B
Inheritance: class B is a Class A (or class A is extended by class B)



Student Name: _____ **Roll No:** _____ **Section:** _____

1. Association

An association is a "using" relationship between two or more objects in which the objects have their own life time and there is no owner.

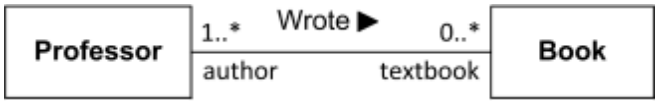
For **Example**: A patient may visit one or many doctors and same way, a doctor can be associated with multiple patients. If a patient dies, existence of doctor will not be vanished and similarly if doctor dies patient will remain patient.

Association is represented as thin line connecting two classes. Association can be unidirectional (shown by arrow at one end) or bidirectional (shown by arrow at both end) or without arrow.

Multiplicity defines how many instances can be associated at any given moment**.**

0..1	No instances or one instance	A flight seat can have no or one passenger only
1	Exactly one instance	A class can have zero or more students
0..* or *	Zero or more instances	A class can have zero or more students.
1..*	One or more instances (at least one)	A flight can have one or more passenger

Example:



Association Wrote between Professor and Book with association ends author and textbook.

2. Aggregation

Aggregation is a special form of association. It is also a relationship between two classes like association, however, it's a **directional** association, which means it is strictly a **one way association, means unidirectional association.** It represents a **Has-A** relationship.

For Example: Consider two classes Student class and Address class. Each student must have an address so the relationship between student and address is a Has-A relationship. But if you consider its vice versa then it would not make sense as an Address doesn't need to have a Student necessarily.
NOTE: Unarguably, Address is an attribute of a student, but here in this example I am breaking address into
several fields i.e city, province and country. This is the reason for making address a class**.**

Student Name: _____ **Roll No:** _____ **Section:** _____

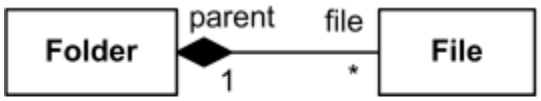
3. Composition

Composition is a special case of aggregation. In a more specific manner, a restricted aggregation is called composition. When an object contains the other object, if the contained object cannot exist without the existence of container object, then it is called composition.

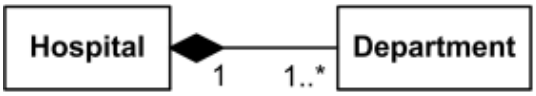
For **Example**: Consider the same scenario with some modifications. In this scenario, a student has address and each student has different address (Please keep sibling relationship argument apart). So, when a student record is added his house number and street number will be entered. And if I delete the record of a particular student, then his/her record will be of no use.

filled black diamond at the aggregate (whole) end.

Example:



Folder could contain many files, while each File has exactly one Folder parent.
If Folder is deleted, all contained Files are deleted as well.



Hospital has 1 or more Departments, and
each Department belongs to exactly one Hospital.
If Hospital is closed, so are all of its Departments.

4. Generalization

In object oriented programming, the concept of IS-A is a totally based on Inheritance, which can be of two types Class Inheritance or Interface Inheritance. It is just like saying "A is a B type of thing". For example, Apple is a Fruit, Car is a Vehicle etc. Inheritance is uni-directional. For example House is a Building. But Building is not a House.

It is key point to note that you can easily identify the IS-A relationship. Wherever you see an extends keyword or implements keyword in a class declaration, then this class is said to have IS-A relationship.

Example: Refer your theory lectures.

Student Name: _____ **Roll No:** _____ **Section:** _____

Case Study1

Library Management System

Problem Statement:

The case study titled Library Management System is library management software for the purpose of monitoring and controlling the transactions in a library. This case study on the library management system gives us the complete information about the library and the daily transactions done in a Library. We need to maintain the record of new s and retrieve the details of books available in the library which mainly focuses on basic operations in a library like adding new member, new books, and up new information, searching books and members and facility to borrow and return books. It features a familiar and well thought-out, an attractive user interface, combined with strong searching, insertion and reporting capabilities. The report generation facility of library system helps to get a good idea of which are ths borrowed by the members, makes users possible to generate hard copy.

The following are the brief description on the functions achieved through this case study:

End-Users:

•Librarian: To maintain and update the records and also to cater the needs of the users.

•Reader: Need books to read and also places various requests to the librarian.

Vendor: To provide and meet the requirement of the prescribed books.

Student Name: _____ **Roll No:** _____ **Section:** _____

Use-case Diagram

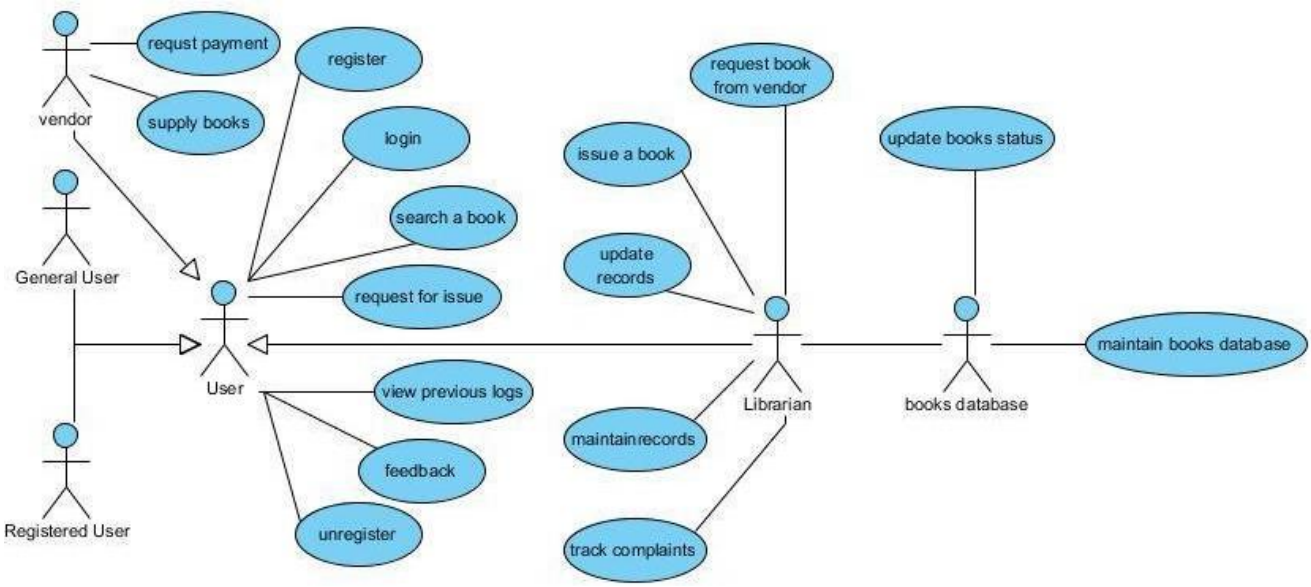
Actor's vs Use Cases:

Librarian
•Issue a book
•Update and maintain records
•Request the vendor for a book
•Track complaints.

User
•Register
•Login
•Search a book
•Request for issue
•View history
•Request to the Librarian
•Unregister

Books Database
•Update records
•Show books status

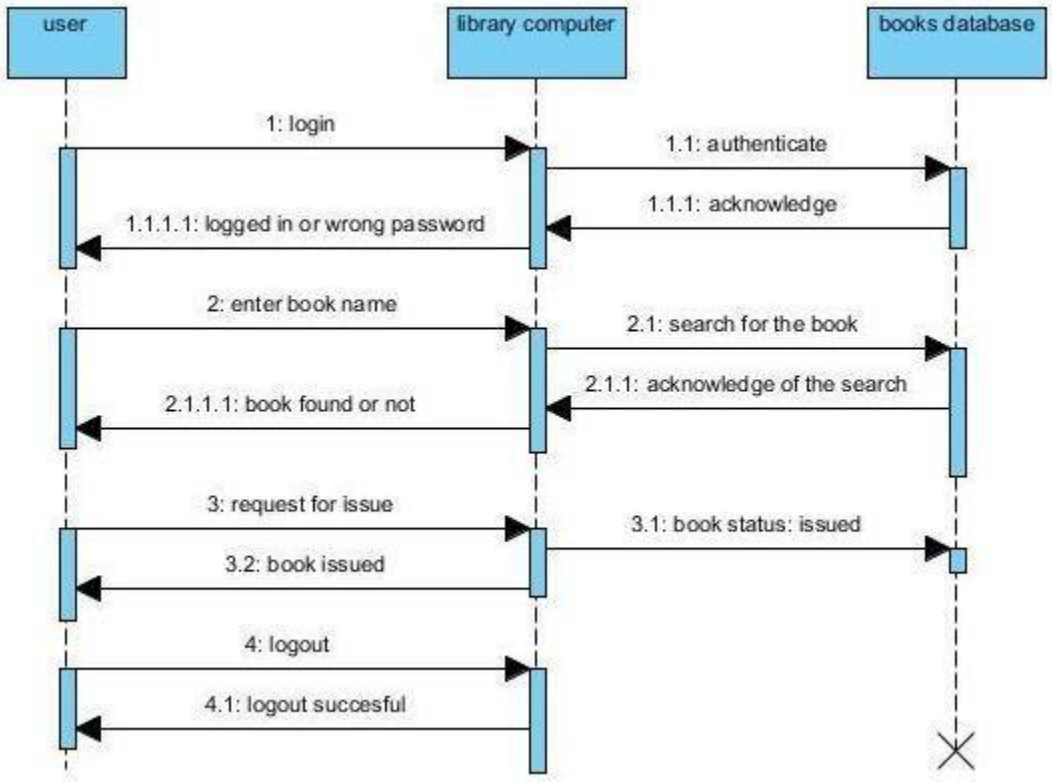
Vendors
•Provide books to the library
•Payment acknowledgement



Student Name: _____ **Roll No:** _____ **Section:** _____

Sequence Diagram

Sequence diagram for searching a book and issuing it as per the request by the user from the librarian:

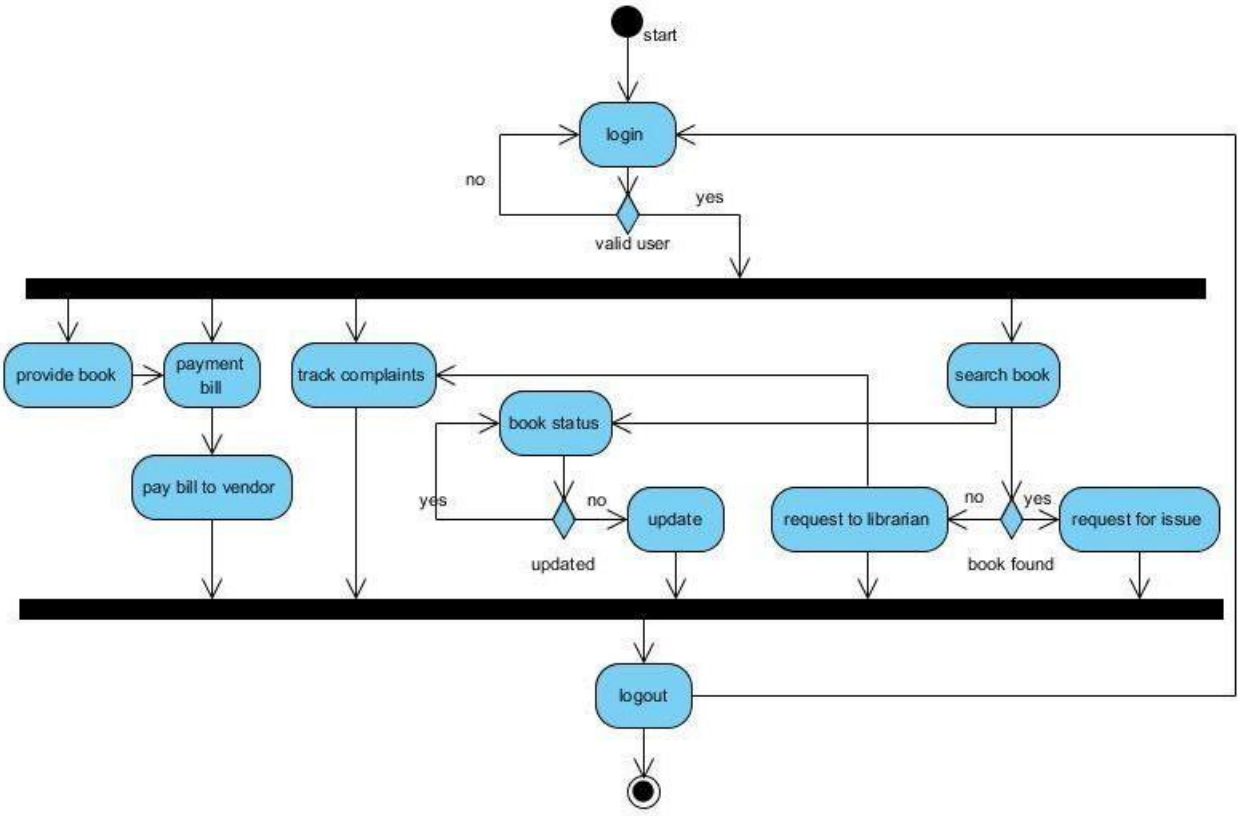


Student Name: _____ **Roll No:** _____ **Section:** _____

Activity Diagram

Activities:

User Login and Authentication
Search book operation for Reader
Acknowledge and Issue books to the users by the Librarian
Provide books requested by the Librarian from the Vendor
Bill payment from the Librarian to the Vendor
Status of the books updated in the Books Database



Student Name: _____ **Roll No:** _____ **Section:** _____

Programming Exercise (OOAD)

Task 1: Create a UML Class diagram of the below scenario also include all possible relations

1. An **Order** is ordered by a **Customer**.
2. An **Order** is fulfilled by an **Employee**.
3. An **Order** is paid via a **Payment Method**.
4. An **Order** is shipped via an **Address** belonging to the **Customer** who is the buyer.
5. An **Order** is composed of **Order Items**.

Note: Add appropriate attributes and methods according to the order management system

Task 2 Create a UML Class diagram of the given School Management System.

- A school has one or more departments
- A department can only belong to one school
- A school has one or more students.
- A student can attend one or more courses
- A courses can have one or more students
- A course is taught by one teacher only
- A teacher can teach one or more courses
- A teacher is assigned one or more departments
- A department can have one or more teachers.
- A teacher can be the chairman of zero or one department.
- A department can have zero or one chairman (teacher)
- A course belongs to one or more departments.