CLASS DIAGRAM

Compiled by-

Ms.Aquilla Afonso

Assistant Professor,Department of BCA

Class diagram

A UML class diagram is made up of:

- A set of classes and
- A set of relationships between classes

What is a Class

- A description of a group of objects all with similar roles in the system, which consists of:

- **Structural features** (attributes) define what objects of the class "know"
  - Represent the state of an object of the class
  - Are descriptions of the structural or static features of a class

- **Behavioral features** (operations) define what objects of the class "can do"
  - Define the way in which objects may interact
  - Operations are descriptions of behavioral or dynamic features of a class

Class diagram representation

objects and **classes** relate to each other in a **class diagram**:

1. Class:

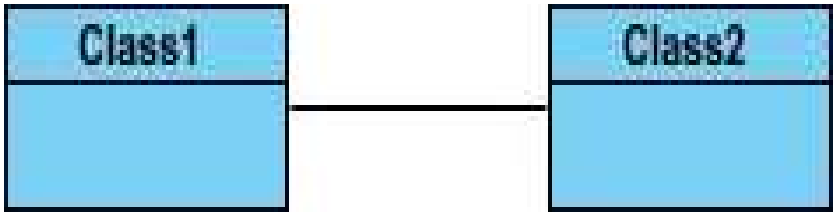
- A class represents a blueprint or template for creating objects (instances). It defines the properties (attributes) and behaviors (methods) that the objects of that class will have.
- In the class diagram, a class is usually represented by a rectangle divided into three parts:
  - **Top part**: Class name
  - **Middle part**: Attributes (variables) of the class
  - **Bottom part**: Methods (functions) of the class

Class diagram representation

- **2. Object:**
- An object is an instance of a class. It represents an actual entity in the system that has specific values for the attributes defined by its class.
- In a class diagram, objects are typically represented as instances of the classes and are shown as a specific instantiation of the class, usually outside the class box itself. Objects are often represented as labeled rectangles or as instances in a dynamic model.

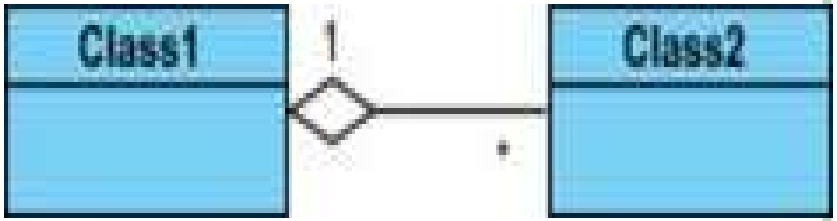
3. Relationships between Classes and Objects in a Class Diagram:

- **Association**: This represents a relationship between two classes where objects of one class are connected to objects of another class. For example, a Student class might be associated with a Course class if a student can enroll in a course.
  - Depicted as a line connecting two classes.



3. Relationships between Classes and Objects in a Class Diagram:

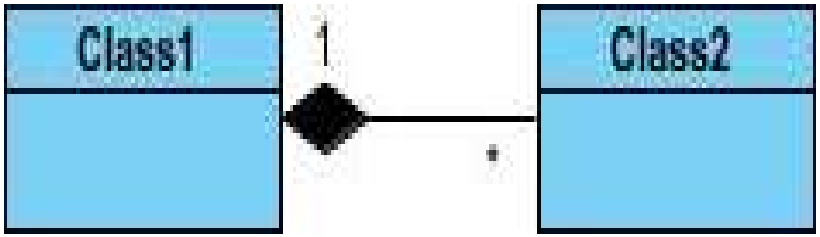
- **Aggregation/Composition**: These are special types of association that represent "whole-part" relationships.
  - Aggregation is shown with an open diamond at the whole (aggregating) class, and composition is shown with a filled diamond at the whole class.



  - **Aggregation**: A "has-a" relationship where the whole can exist without the part. For example, a Library may contain many Books, but books can exist independently of the library.

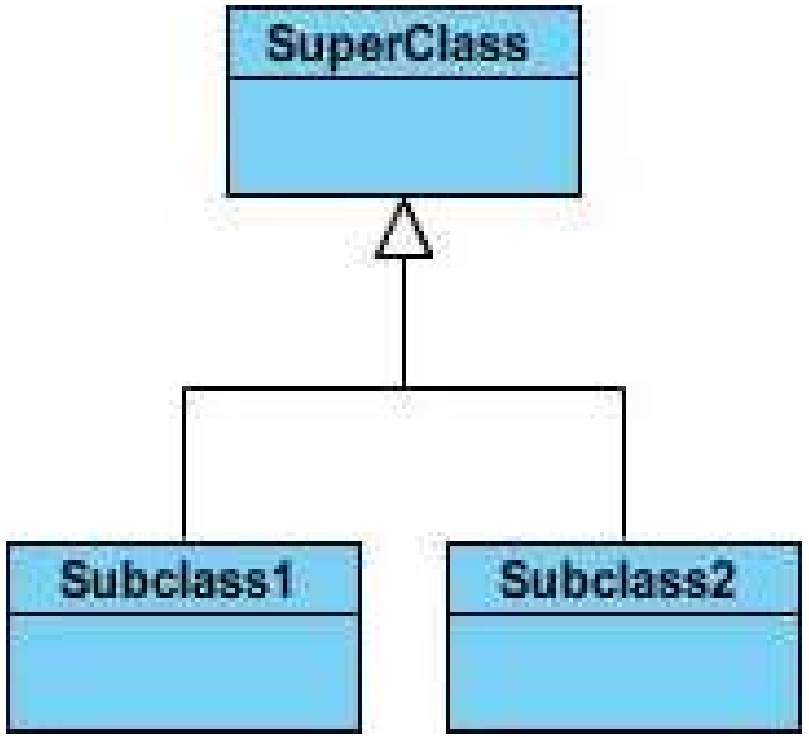
3. Relationships between Classes and Objects in a Class Diagram:

- **Composition**: A "strong" form of aggregation where the part cannot exist without the whole. For example, a House consists of Rooms, and rooms cannot exist independently of a house.



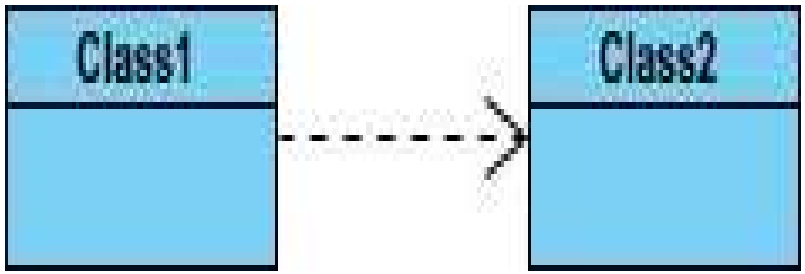
3. Relationships between Classes and Objects in a Class Diagram:

- **Generalization (Inheritance)**: This relationship shows that one class is a specialized version of another class. For example, a Dog class might inherit from an Animal class.
  - Depicted as a line with a hollow triangle pointing toward the parent class.



3. Relationships between Classes and Objects in a Class Diagram:

- **Dependency**: This shows that one class depends on another class in some way, often for invoking its methods or using its attributes. For example, a Car class may depend on a Wheel class.
  - Represented as a dashed line with an arrow pointing from the dependent class to the class it depends on.



Example:

- Consider a class diagram with the following classes:
- **Person** (with attributes like name, age, and methods like speak())
- **Student** (inherits from Person, with an additional attribute studentId)
- **Course** (with attributes like courseName and methods like enroll())
- **Enrollment** (represents the relationship between a Student and a Course, showing that students can enroll in courses)
- The **Student** class would be a subclass (child class) of the **Person** class, representing inheritance (generalization).
- The **Student** and **Course** classes would have an association relationship, shown with a line connecting them, with the Enrollment class acting as a bridge to manage the relationship.

- In this example:
- The **Student** class inherits from the **Person** class (represented by the arrow).
- The **Student** and **Course** classes are associated through the **Enrollment** class, showing the relationship between students and courses.

Examples of Aggregation:

- **Aggregation (A "Has-A" relationship where the whole can exist without the part)**

1. University and Department

- **Aggregation**: A **University** "has" multiple **Departments**, but a department can exist independently of the university. A department can be transferred to another university or even exist independently.
- **Example**:
  - **University** class has a collection of **Department** objects.
  - If a university is closed, the departments may still exist in another context.

2. Library and Books

- **Aggregation**: A **Library** "has" **Books**, but if the library is closed or moved, the books can exist independently in other libraries or locations. Books can exist without the library system itself.
- **Example**:
  - **Library** class contains multiple **Book** objects.
  - Books may be borrowed by readers, and they can be moved to another library or stand alone.

Examples of Composition:

- **Composition (A strong "Has-A" relationship where the part cannot exist without the whole)**

1. House and Rooms

- **Composition**: A **House** "has" **Rooms**, but if the house is destroyed, the rooms cannot exist independently. The rooms are part of the house, and without the house, the rooms would not make sense.
- **Example**:
  - A **House** class is composed of **Room** objects.
  - When the house is demolished, the rooms are gone as well.

2. Body and Organs

- **Composition**: A **Body** "has" **Organs** (e.g., heart, liver), and the organs cannot exist without the body. If the body is dead, the organs would also cease to function and cannot exist independently.
- **Example**:
  - A **Body** class contains **Organ** objects.
  - If the body dies, the organs are no longer functional.

Examples of Composition:

3. Computer and CPU

- **Composition**: A **Computer** "has" a **CPU**. If the computer is shut down or removed, the CPU is no longer functional or independent. The CPU is an essential part of the computer, and without the computer, the CPU cannot perform its purpose.
- **Example**:
  - A **Computer** class contains a **CPU** object.
  - The CPU cannot operate outside of the computer.

MULTIPLICITY

- Multiplicity is an association relationship and indicates that at least one of the two classes refers to the other when creating instances of itself.

- In other words, it shows the cardinality of the relationship.

Multiplicities examples:

1	Exactly one, no more and no less
0..1	Zero or one
*	Many
0..*	Zero or many
1..*	One or many

THANK YOU!!!!