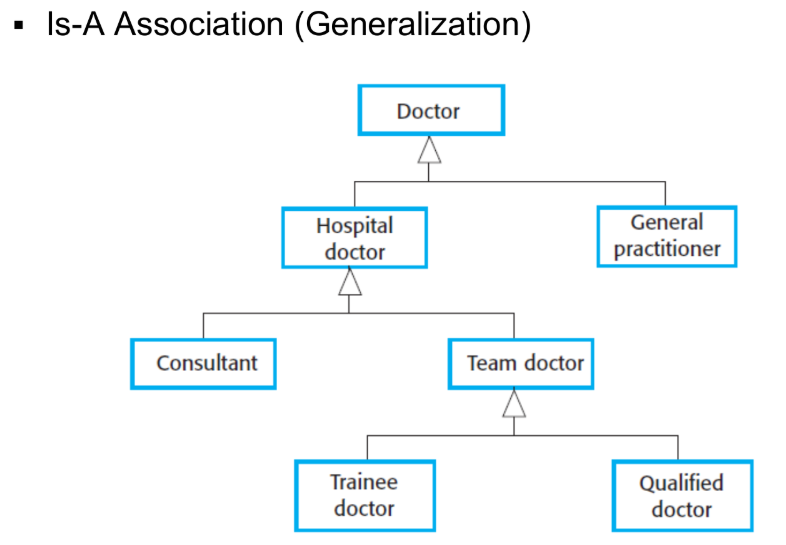
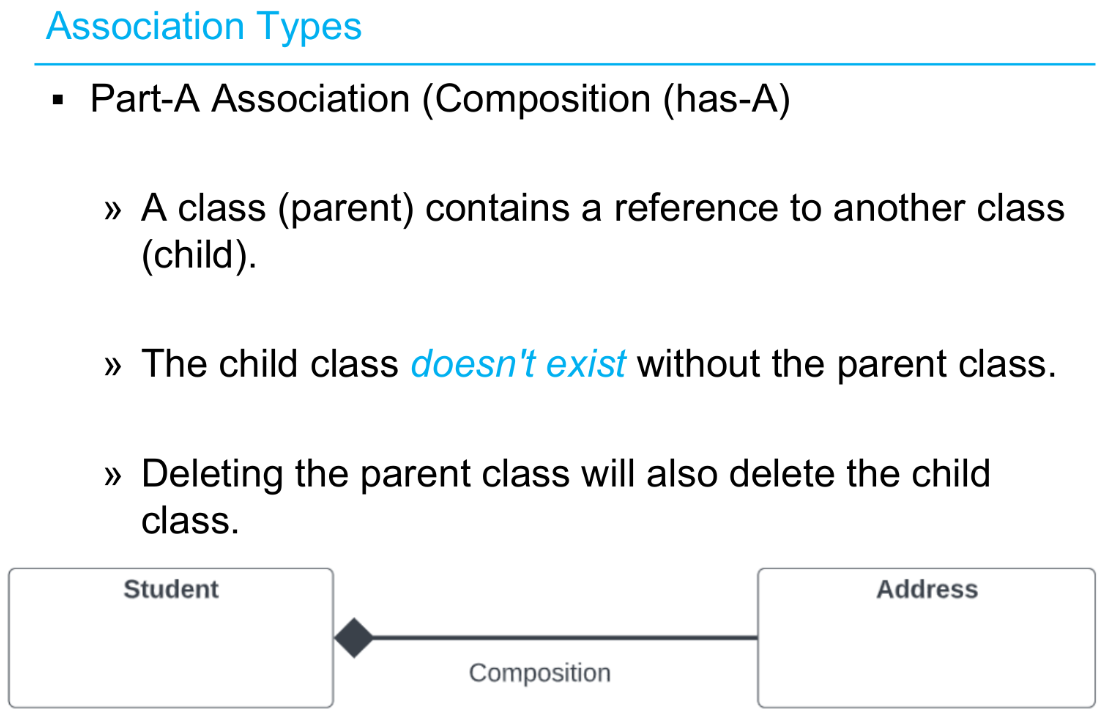
**Class Diagram**

**->** Also known as **Inheritance**



A screenshot of a computer

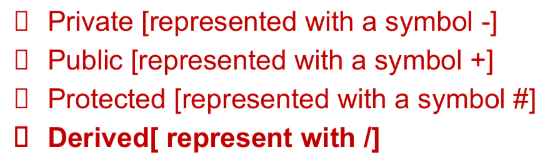
AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

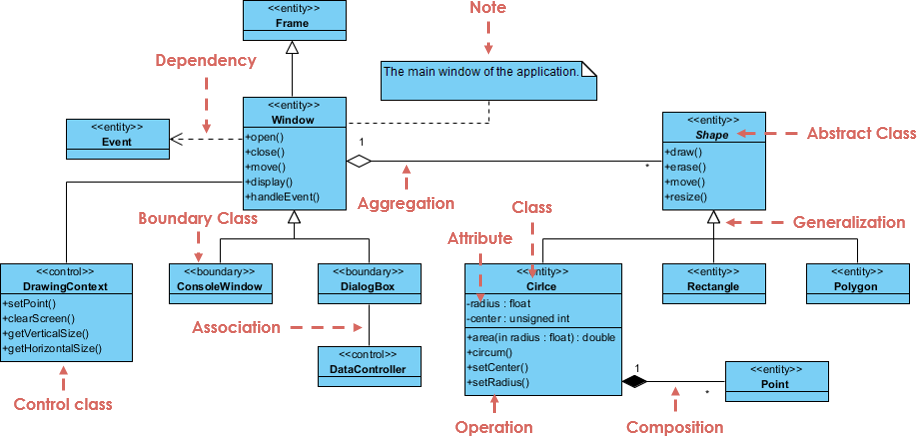
AI-generated content may be incorrect.



A black and red text

AI-generated content may be incorrect.

In a UML class diagram, **operations** (also known as methods or functions) represent the behaviors or actions a class can perform



Abstract Class name is always written in ***Italics***.

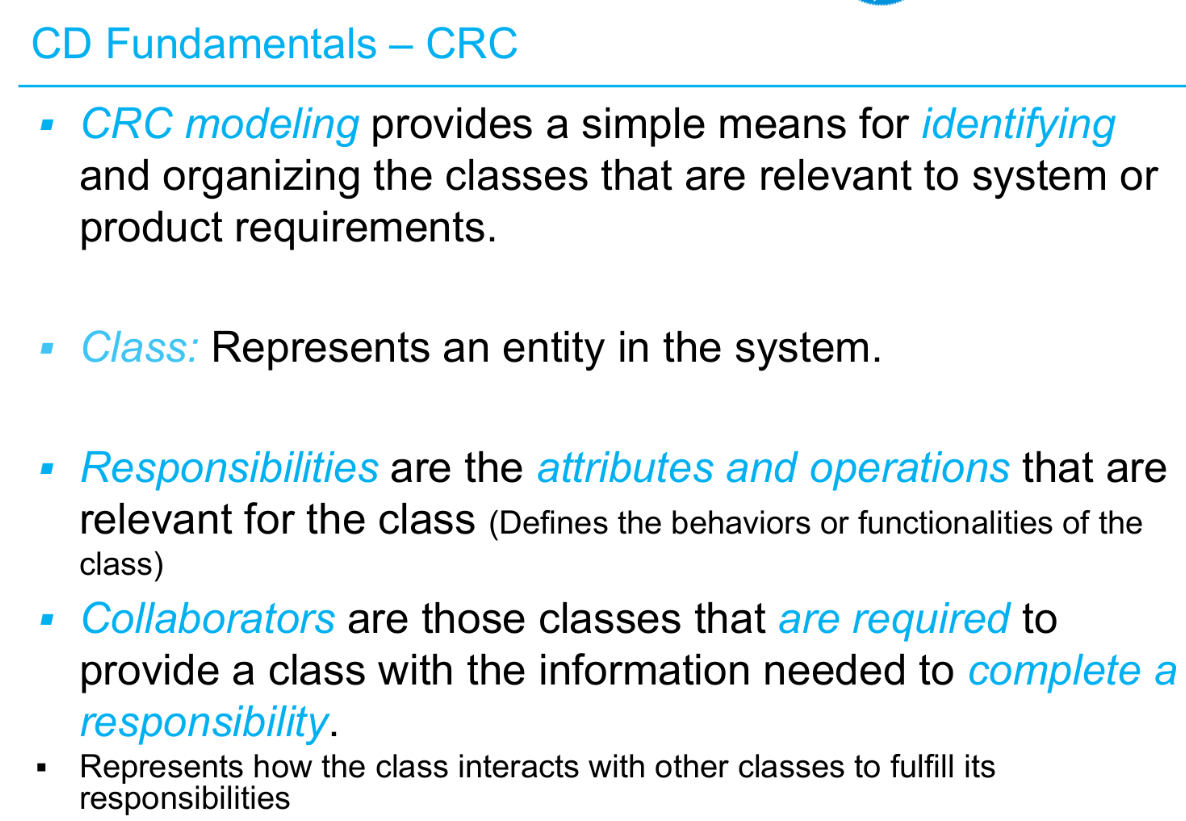
**Stereotypes**

**<<entity>>** classes model persistent data, **<<boundary>>** classes handle interactions with external actors, **<<controller>>** classes manage business logic, and **Data access** classes (often part of the Entity layer) provide persistence mechanisms.

* **Entity Classes:**
  + Represent the core data of the system, often mapping to database tables or files.
  + Contain attributes and methods related to the data they represent (e.g., a Customer entity might have attributes like name, address, and methods like getName(), getAddress()).
  + Are often considered passive, meaning they don't initiate interactions on their own.
  + Example: Customer, Product, Order.
* **Boundary Classes:**
  + Act as intermediaries between the system and external actors (users, other systems).
  + Represent user interfaces, gateways, or other interfaces that interact with the system.
  + Often handle input/output and passing data to/from the controller.
  + Example: LoginUI, OrderForm, API Gateway.
* **Controller Classes:**
  + Implement the business logic and arrange interactions between entities and boundaries.
  + Receive requests from boundary classes, perform actions on entities, and potentially send responses back to the boundary.
  + Example: LoginController, OrderController, PaymentController.
* **Data Access Classes (often part of the Entity layer):**
  + Handle the persistence of data, interacting with databases or other storage mechanisms.
  + Provide methods for creating, reading, updating, and deleting (CRUD) operations on entities.
  + Example: CustomerDAO, ProductDAO, OrderDAO.

A screenshot of a text

AI-generated content may be incorrect.

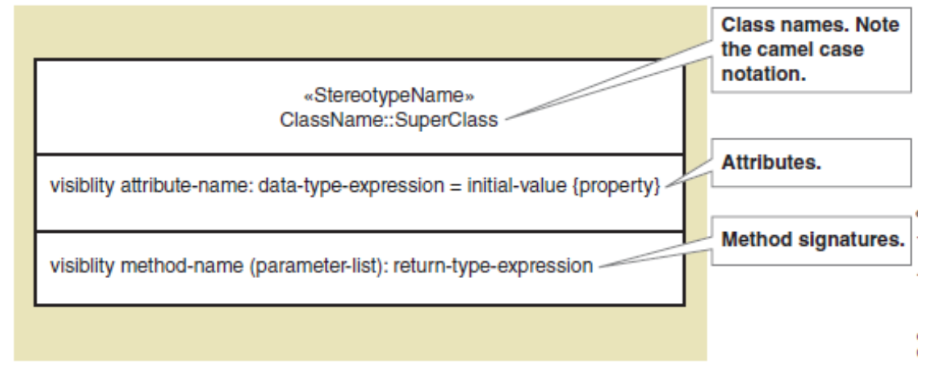


A black text on a white background

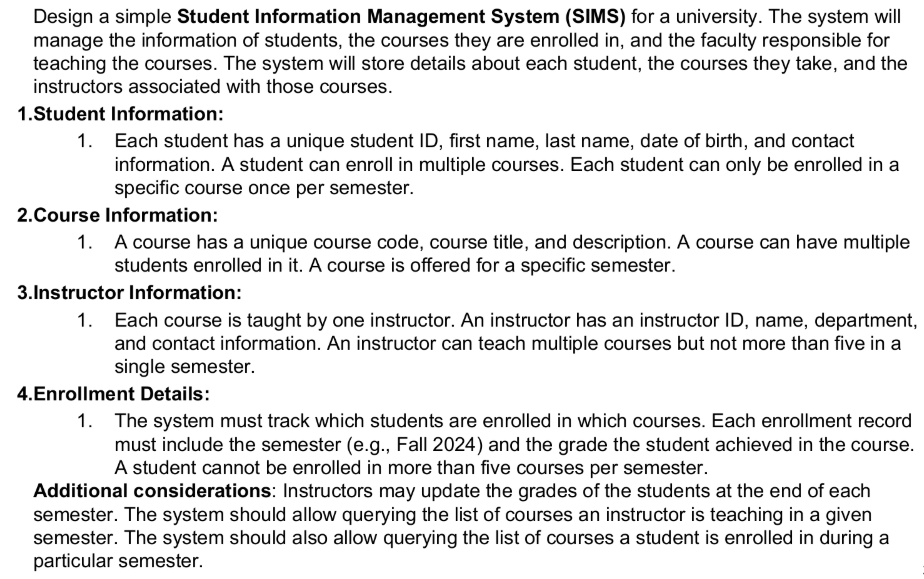
AI-generated content may be incorrect.

A list of items on a white background

AI-generated content may be incorrect.



* Methods/Operations
  + The name, parameter list and return type of an operation/method are collectively known as its Signature.
  + The operation signatures provide a clear specification for the collaboration between operations.



**1. Identify Classes (from the problem description):**

* Student
* Course
* Instructor
* Enrollment
* Semester

**2. Create CRC Cards for Each Class:**

Here's a breakdown of the responsibilities and potential collaborators for each class:

**CRC Card: Student**

* **Class:** Student
* **Responsibilities:**
  + Maintain personal information (ID, name, DOB, contact info).
  + Know which courses the student is enrolled in for a specific semester.
  + Enroll in a course for a specific semester.
  + View their enrollment history and grades.
* **Collaborators:**
  + Enrollment
  + Course
  + Semester

**CRC Card: Course**

* **Class:** Course
* **Responsibilities:**
  + Maintain course information (code, title, description).
  + Know which students are enrolled in the course for a specific semester.
  + Know which instructor is teaching the course for a specific semester.
  + Know the semester the course is offered in.
* **Collaborators:**
  + Enrollment
  + Student
  + Instructor
  + Semester

**CRC Card: Instructor**

* **Class:** Instructor
* **Responsibilities:**
  + Maintain personal information (ID, name, department, contact info).
  + Know which courses the instructor is teaching in a specific semester.
  + Update student grades for a specific course and semester.
* **Collaborators:**
  + Course
  + Enrollment
  + Semester

**CRC Card: Enrollment**

* **Class:** Enrollment
* **Responsibilities:**
  + Record which student is enrolled in which course.
  + Record the semester of the enrollment.
  + Record the grade achieved by the student in the course.
  + Ensure a student is not enrolled in the same course twice in a semester.
  + Ensure a student is not enrolled in more than five courses per semester.
* **Collaborators:**
  + Student
  + Course
  + Semester

**CRC Card: Semester**

* **Class:** Semester
* **Responsibilities:**
  + Maintain the name or identifier of the semester (e.g., Fall 2024).
  + Know which courses are offered in this semester.
  + Know which students are enrolled in courses during this semester (through Enrollment).
  + Know which instructors are teaching courses during this semester (through Enrollment and Course).
* **Collaborators:**
  + Course
  + Student
  + Instructor
  + Enrollment