**NATIONAL UNIVERSITY OF COMPUTER AND**

**EMERGING SCIENCES**

**SL-2002 – Software Design &amp; Architecture Lab**

**Lab Instructor: Zarnain Maryam Awan**

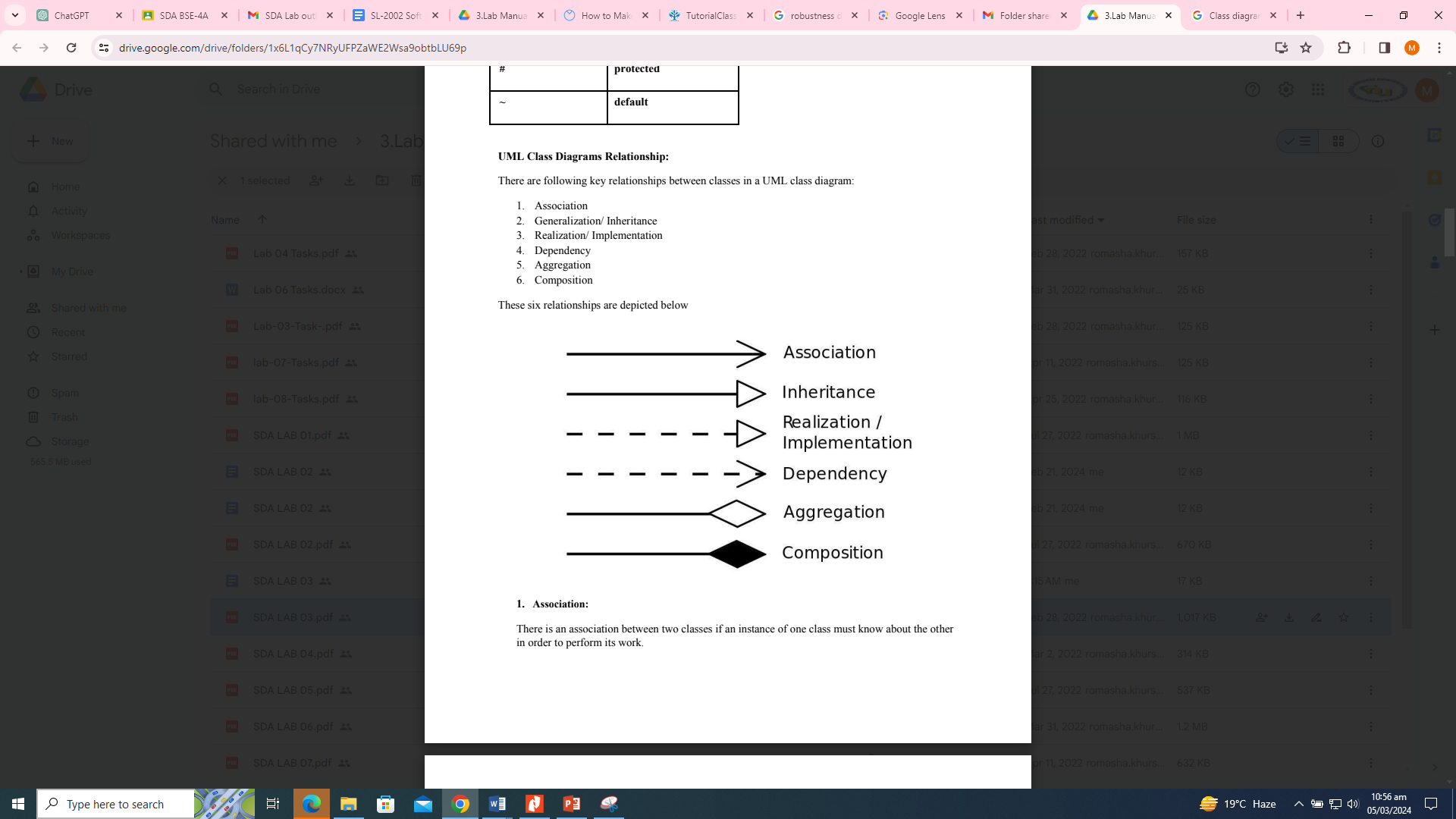
Lab 04

Creating a class diagram involves visually representing the structure and relationships of classes in an object-oriented system. Here's a step-by-step guide to creating a basic class diagram:

1. **Identify classes**: Begin by identifying the classes in your system. A class represents a group of objects with similar properties, behaviors, and relationships.
2. **List attributes and methods**: For each class, list its attributes (data members) and methods (functions or operations). Attributes describe the state of objects, while methods define their behavior.
3. **Define relationships**: Determine the relationships between classes. Common relationships include association, aggregation, and inheritance.
   * **Association**: Represents a relationship between two or more classes, where objects of one class are connected to objects of another class.

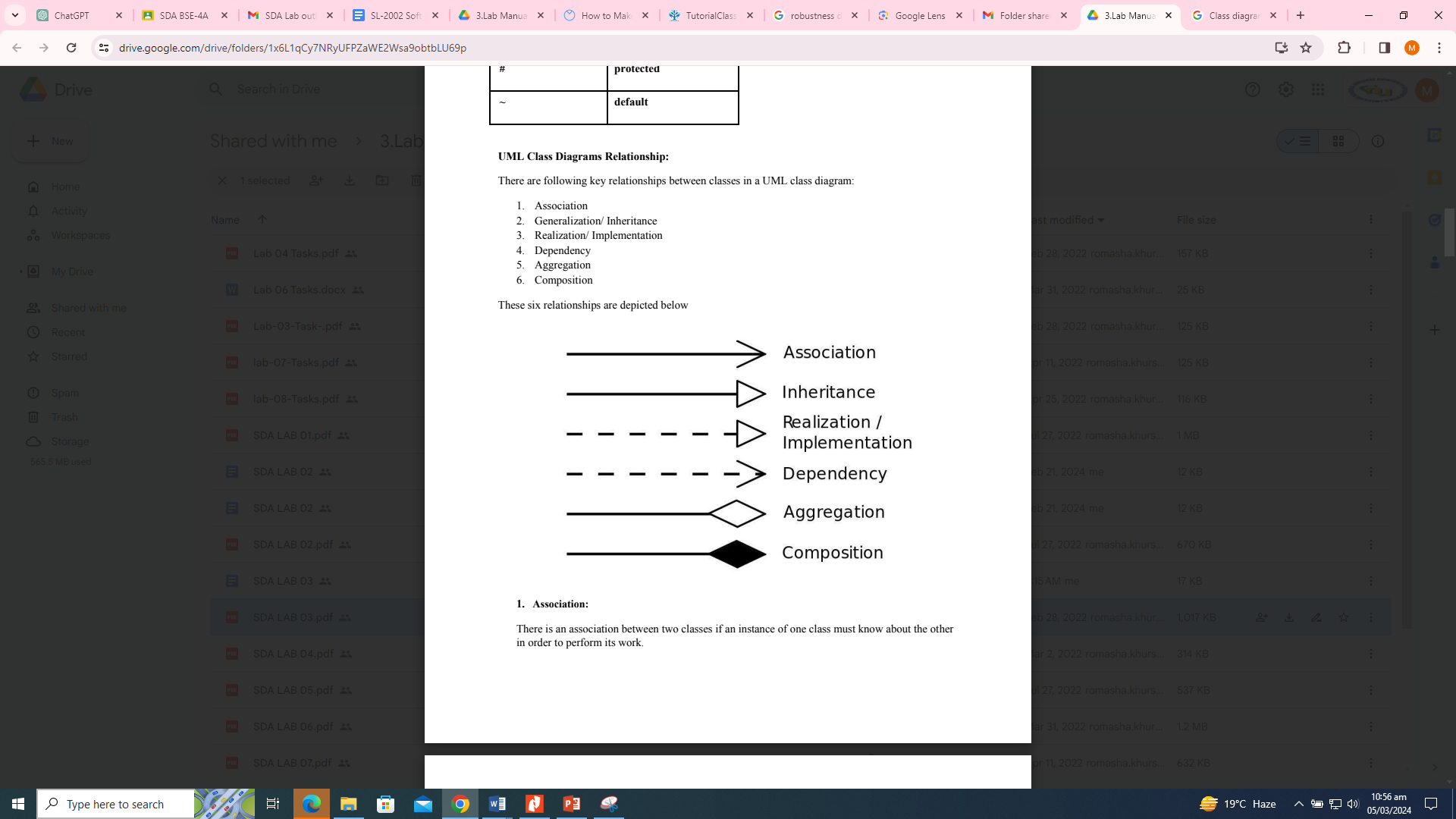


* + **Aggregation**: Indicates a "whole-part" relationship, where a whole object contains or is composed of part objects. It's represented by a hollow diamond at the containing end. 
  + **Composition**: A stronger form of aggregation where the part objects are exclusively owned by the whole object and have a lifecycle dependency on it. It's represented by a filled diamond at the containing end.



* + **Inheritance**: Depicts an "is-a" relationship, where one class (subclass or child class) inherits attributes and methods from another class

(superclass or parent class). It's represented by a solid line with a closed arrowhead pointing to the superclass.



1. **Draw the diagram**: Use a modeling tool or software to draw the class diagram.
2. **Add class boxes**: Draw rectangles to represent each class. Write the class name at the top of the rectangle.
3. **Add attributes and methods**: Inside each class box, list its attributes and methods. You can use notation like "+" for public, "-" for private, and "#" for protected visibility.
4. **Connect classes with relationships**: Use lines with appropriate notations to connect classes and represent their relationships. Label the relationships to indicate their nature (e.g., association, aggregation, composition, inheritance).
5. **Review and refine**: Check the diagram to ensure it accurately represents the system. Make adjustments if necessary.

**Example scenario:**

The University of Toronto has several departments. Each department is managed by a chair, and at least one professor. Professors must be assigned to one, but possibly more departments. At least one professor teaches each course, but a professor may be on sabbatical and not teach any course. Each course may be taught more than once by different professors. We know of the department name, the professor name, the professor employee id, the course names, the course schedule, the term/year that the course is taught, the departments the professor is assigned to, the department that offers the course.

**Identify classes:**

The class candidates are departments, chair, professor, course, and course section. Since there is only one instance of the University of Toronto, we exclude it from our consideration.

**List attributes and methods**:

We have the department name, the professor name, the professor employee id, the course names, the course schedule, the term/year that the course is taught

Specify the operations that are required for each class. (assume getter and setter methods for each attribute.) In this example - we are not given any behaviors, so we will have to make them up. What are some behaviors of these classes?

**Define relationships:**

Now find the verbs that join the nouns. e.g., The professor (noun) teaches (verb) students (noun). The verb in this case, defines an association between the two nouns. Identify the type of association. Use a matrix to define the associations between classes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | department | chair | professor | course |
| department |  | managed by | is assigned  (aggregate) | offers |
| chair | manages |  | is a (Generalization) |  |
| professor | assigned to  (aggregate) |  |  | teaches |
| course | offered by |  | taught by |  |

