Object Oriented Systems

Dr. Krishnendu Guha

Assistant Professor
(On Contract)

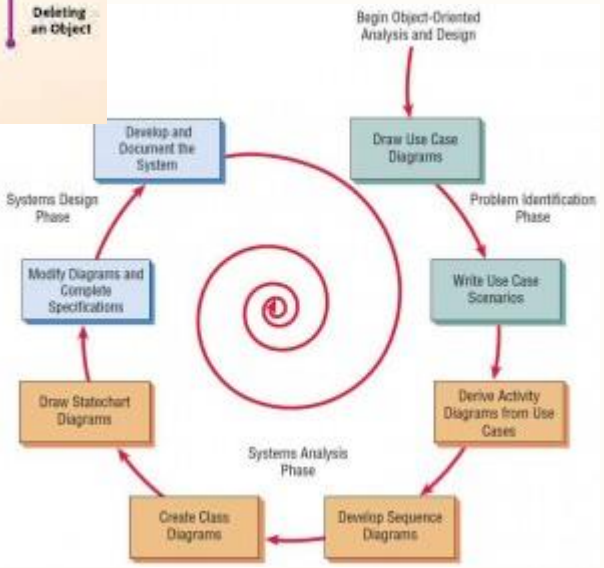
National Institute of Technology (NIT), Jamshedpur

Email: krishnendu.ca@nitjsr.ac.in

Introduction

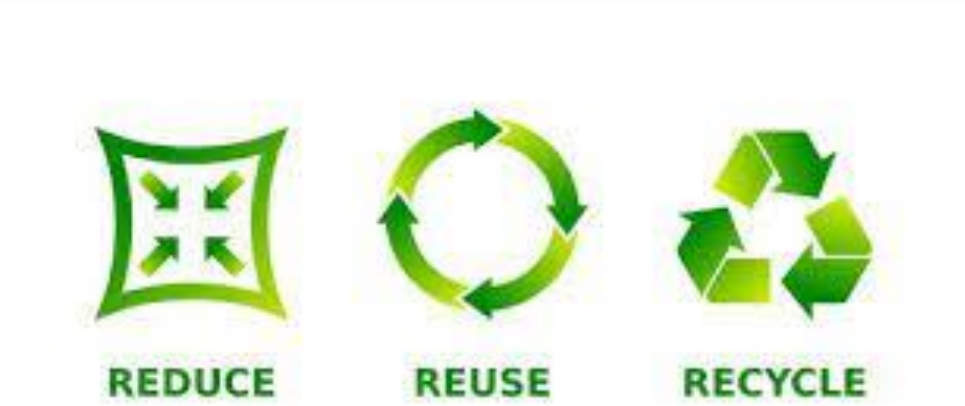
- Object-oriented analysis and design can offer an approach that facilitates logical, rapid, and thorough methods for creating new systems responsive to a changing business landscape.

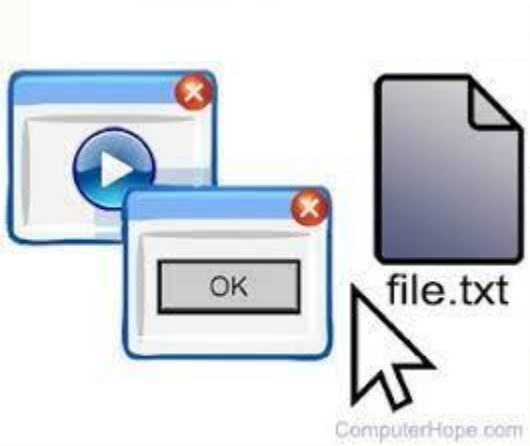
- Object-oriented techniques work well in situations in which complicated information systems are undergoing continuous maintenance, adaptation, and redesign.
- Object-oriented programming differs from traditional procedural programming by examining the objects that are part of a system.
- Each object is a computer representation of some actual thing or event.
- General descriptions of the key object-oriented concepts of objects, classes, and inheritance



Why Object Oriented Systems?

- Reusability of the objects as the main benefit of their approach.
- It makes intuitive sense that the recycling of program parts should reduce the costs of development in computer-based systems.
- It has proved to be very effective in the development of GUIs and databases.
- Maintaining systems
- As the object-oriented approach creates objects that contain both data and program code, a change in one object has a minimal impact on other objects.





Objects

- Objects are persons, places, or things that are relevant to the system we are analyzing.
- Object oriented systems describe entities as objects.

- Typical objects may be
- customers,
- items,
- orders,
- and so on.

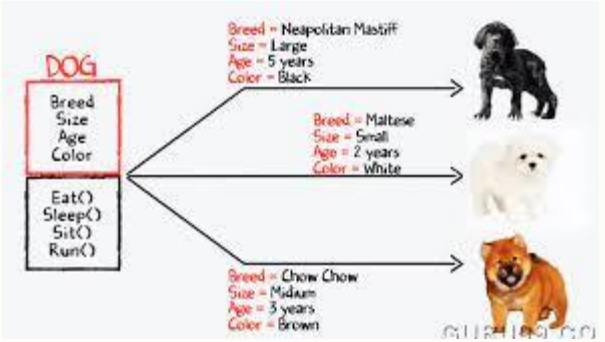
- Objects may also be GUI displays or text areas on the display.

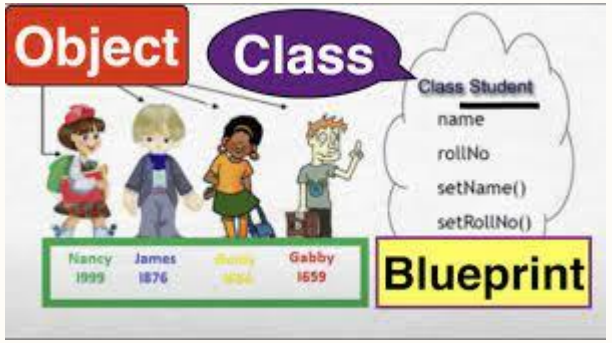
Classes

- Objects are typically part of a group of similar items called classes.
- Describing the world as being made up of animals, vegetables, and minerals is an example of classification.
- The idea behind classes is to have a reference point and describe a specific object in terms of its similarities to or differences from members of its own class.
- It is more efficient to describe characteristics, appearance, and even behavior in this way.
- When you hear the word *reusable* in the object-oriented world, it means you can be more efficient, because you do not have to start at the beginning to describe an object every time it is needed for software development.

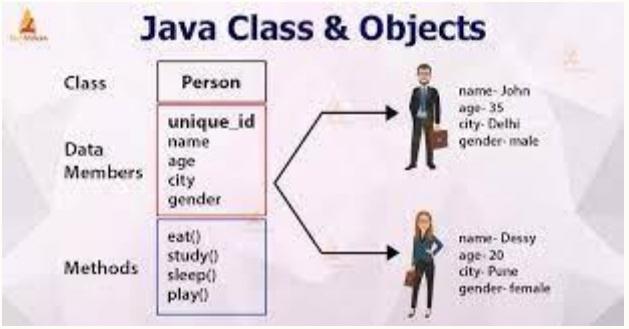
Objects and Classes

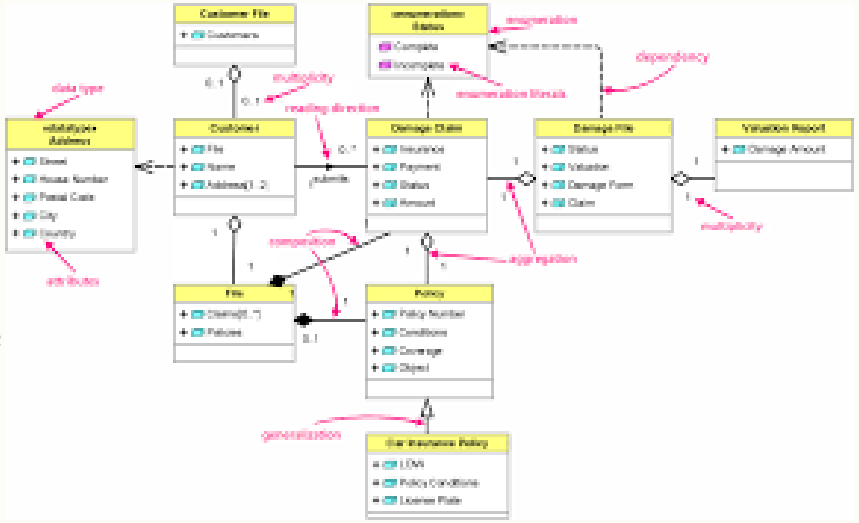
- Objects are represented by and grouped into classes that are optimal for reuse and maintainability.
- A class defines the set of shared attributes and behaviors found in each object in the class.
- For example, records for students in a course section have similar information stored for each student.
- The students could be said to make up a class (no pun intended).
- The values may be different for each student, but the type of information is the same.
- Programmers must define the various classes in the program they are writing. When the program runs, objects can be created from the established class.
- The term *instantiate* is used when an object is created from a class.
- For example, a program could instantiate a student named PeterWellington as an object from the class labeled as student.



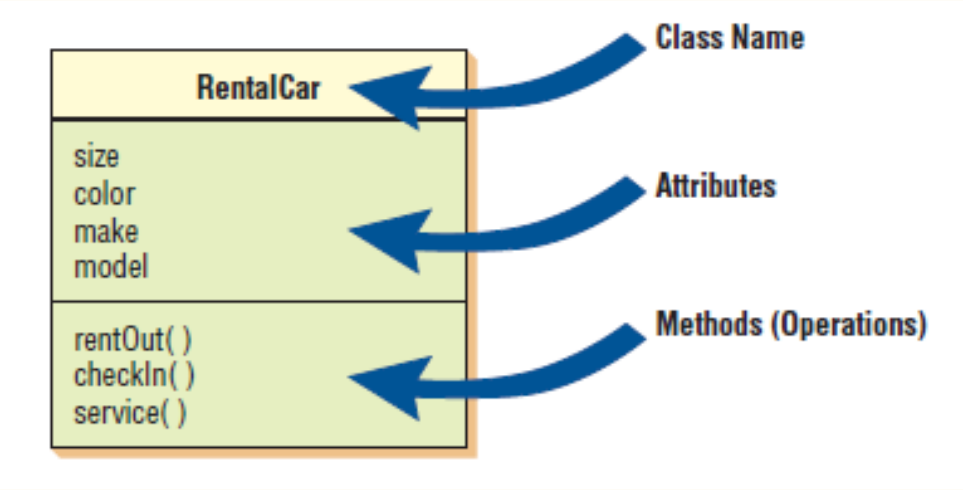


- What makes object-oriented programming, and thus object-oriented analysis and design, different from classical programming is the technique of putting all of an object's attributes and methods within one self-contained structure, the class itself.
- This is a familiar occurrence in the physical world.
- Each class should have a name that differentiates it from all other classes. Class names are usually nouns or short phrases and begin with an uppercase letter.
- In UML, a class is drawn as a rectangle. The rectangle contains two other important features: a list of attributes and a series of methods.
- These items describe a class, the unit of analysis that is a large part of what we call object-oriented analysis and design.





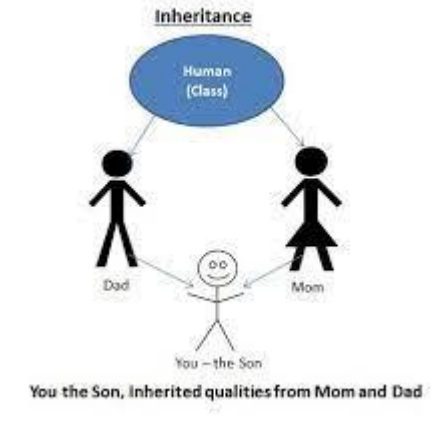
- An attribute describes some property that is possessed by all objects of the class. Notice that the **RentalCar** class possesses the attributes of size, color, make, and model. All cars possess these attributes, but each car will have different values for its attributes.
- For example, a car can be blue, white, or some other color.
- A method is an action that can be requested from any object of the class.
- Methods are the processes that a class knows to carry out.
- Methods are also called operations. For the class of **RentalCar, rentOut(), checkIn(),** and **service()** are examples of methods.
- When specifying methods, the first letter is usually lowercase.

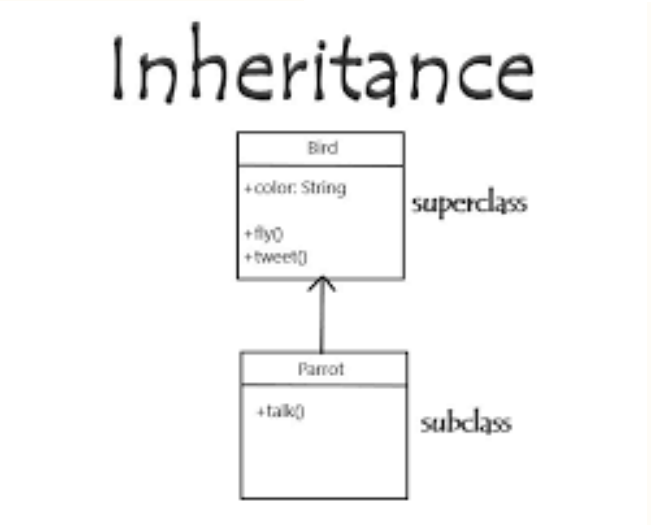


An example of a UML class. A class is depicted as a rectangle consisting of the class name, attributes, and methods.

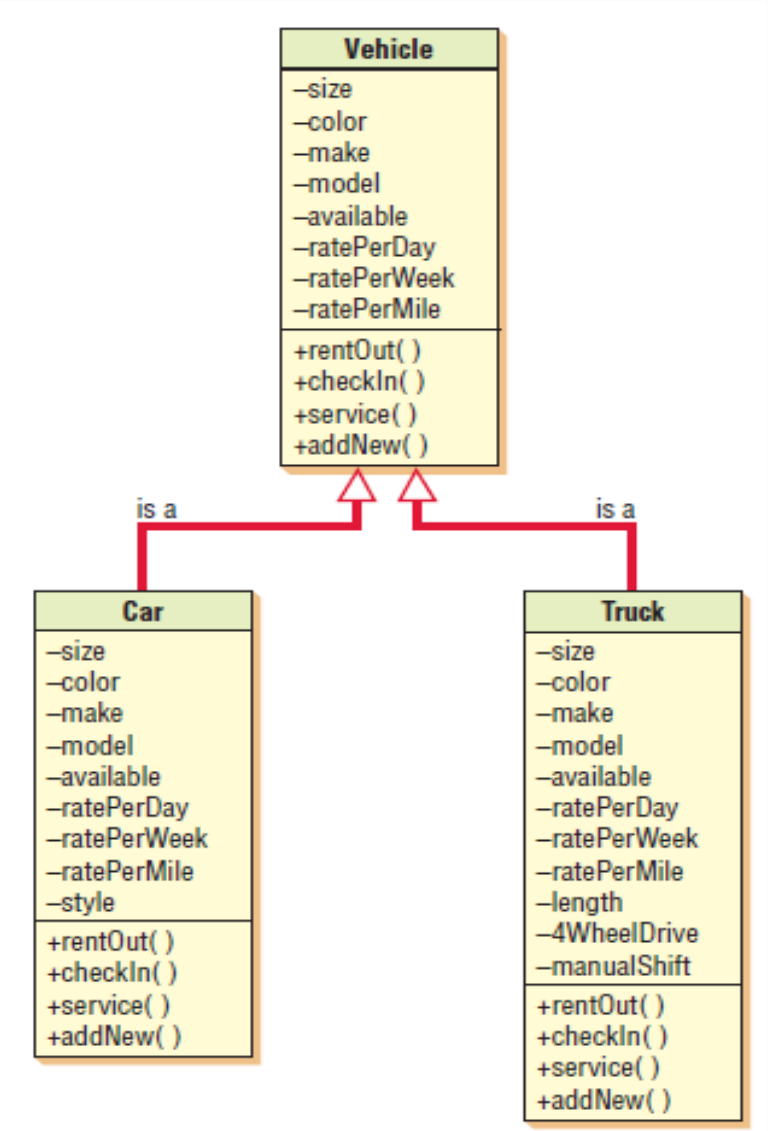
Inheritance

- Classes can have children; that is, one class can be created out of another class.
- In UML, the original—or parent—class is known as a base class.
- The child class is called a derived class.
- A derived class can be created in such a way that it will inherit all the attributes and behaviors of the base class.
- A derived class, however, may have additional attributes and behaviors.
- For example, there might be a **Vehicle** class for a car rental company that contains attributes such as **size, color,** and **make.**
- Inheritance reduces programming labor by using common objects easily.
- The programmer only needs to declare that the **Car** class inherits from the **Vehicle** class, and then provide any additional details about new attributes or behaviors that are unique to a car.
- All the attributes and behaviors of the **Vehicle** class are automatically and implicitly part of the **Car** class and require no additional programming. This enables the analyst to define once but use many times





- The derived classes shown in Figure **Car** or **Truck**.
- Here the attributes are preceded by minus signs and methods are preceded by plus signs.
- the minus signs mean that these attributes are private (not shared with other classes) and these methods are public (may be invoked by other classes).
- Program code reuse has been a part of structured systems development and programming languages (such as COBOL) for many years, and there have been subprograms that encapsulate data.
- Inheritance, however, is a feature that is only found in object-oriented systems.



A class diagram showing inheritance. Car and Truck are specific examples of vehicles and inherit the characteristics of the more general class, **Vehicle.**


THANK YOU
THANKS
THANKS
THANK YOU
THANK YOU
THANK YOU
THANK YOU
THANK YOU
GRAZIE
THANK YOU
THANK YOU
THANKS
GRAZIE
MERCI
GRACIAS
Merci
THANK YOU
THANKS
GRACIAS
THANK YOU
THANK YOU!
Thanks!
THANK YOU