CSc 1350: - Supplementary Notes

Object-oriented Programming Basics

Up to this point, we have learned programming constructs such as decision (*if*) statements and loops. These are fundamental building blocks in structured programming, an approach to programming in which we disaggregate a task in a series of subtasks and write methods that are re-usable to perform these tasks. We now make a transition to a different programming paradigm called *object-oriented programming*. This approach to programming allows us to build larger and more complex programs. We will be able to mimic both concrete and abstract real-world objects.

Definition 1. Object-oriented programming, OOP for short, is a computer programming paradigm that regards a computer program as a collection of individual units, or objects, as opposed to a traditional view in which a program is a list of instructions to the computer. Each object is capable of receiving messages, processing data, and sending messages to other objects. Object-oriented programming allows the programmer tremendous flexibility, easing changes to programs, and is widely popular in large scale software engineering.

There are four features that are often common to OOP languages:

1. **Definition 2. Abstraction**, as defined by Glady Booch, the father of Unified Modeling Language, "denotes the essential characteristics of an object that distinguish it from all other kinds of object and thus provide crisply defined conceptual boundaries, relative to the perspective of the viewer".

2. **Definition 3. Encapsulation** is a form of information hiding in which the internal representation of an object is generally hidden from view outside of the object's definition. Simply put, it is the ability to combine data and operations on that data in a simple unit. Generally, only methods (operations) that are combined with the data can directly access or manipulate the data. In Java, fields or data (called instance variables) that are defined using the *private* access specifier can only be accessed or manipulated using accessors (getters) and mutators (setters), respectively. An **accessor** is a method that gives information about an object and a **mutator** is a method that modifies an object.

3. **Definition 4. Inheritance** is the process by which one class (type) is defined by extending the behavior of an existing class (type). In Java the existing class is called the superclass and the new class is called a subclass. For example, *Quadrilateral*, four-sided plane shapes, may be a superclass and *Rectangle*, four-sided plane figures with perpendicular adjacent sides, may be a subclass of Quadrilateral.

4. **Definition 5. Polymorphism** is the ability to use the same expression to denote different operations. The word "Polymorphism" means many forms. It comes from the Greek word "Poly" which means "many" and "morphous" means "forms".