OOD/OOP

Object Oriented Design

Object-Oriented Design Principles

- Abstraction
- Encapsulation
- Modularity

Abstraction

- Defines function but not implementation
- Abstract Data Type (ADT)
  - Data and operations on that data
  - Operations are specified with a defined interface
  - Specify what and **not how**
- You will use classes to design ADTs this quarter

Encapsulation

- Information hiding
  - Classes and functions
- User does not have access to inner-workings, can only use public interface to change state of data
- Objects encapsulate data and operations
- Functions encapsulate actions

Abstraction/Encapsulation

Honda Civic - Is this the gas or the hybrid version?

Modularity

- Design program as interacting objects or modules - classes and functions
- Closely related to reusability
  - Design a module that you can reuse later
  - Creating and selling physics and graphics engines

Benefits of Modularity

- Program construction
  - Team work
  - Easier to manage
- Debugging
  - Isolates errors
  - Debug each module as you go
    - Easier to locate bugs
  - Do not write entire program and then debug
    - Difficult to locate bugs

Benefits of Modularity

- Reading the program
  - Easier to read
  - Helps designer construct complicated project just as it helps a reader understand
  - Functions
    - Easy to read if they have a good description and descriptive names and parameters
    - Reader should not have to actually read the code in the function unless they require details

Benefits of Modularity

- Modifying the program
  - Isolates modifications
  - Update just a module as it is changed
    - Game and application patches
- Eliminating redundant code