Object Oriented Programming (“OOP”)

- Consists of interacting objects

- An object is a thing, both tangible and intangible, which we can imagine

- An object is comprised of
    - Data
    - Operations to manipulate data

- A class is a “mold” or “template” used to create an object

- An object is an instance of a class

- A class must be defined before an object can be created

Messages and Methods

❖ A message is sent to a class or object to instruct it to perform a task

❖ A method is a sequence of instructions that a class or an object follows to perform a task.

❖ To understand a message there must be a corresponding method (the name of the message must be the same as the name of the method)

❖ Two kinds of methods:
  - Class methods—defined for a class
  - Instance methods—defined for an object

Argument

A value or data passed along with a message to a class or object
(0, 1, or more arguments may be passed to a method)

Return Value

Data returned to the message sender
(Only one (1) return value can be retrieved)

Data Values

- Instance Data Values: information each object stores

- Class Data Values: information shared by all instances OR a representation of collective information about the instances

- ALL instances of the same class will possess the same set of data values

- Two Types:
  - constants: value cannnot change
  - variables: value can change

Inheritance

- ❖ Allows for the design of two or more entities (classes) that are different but share many common features (data values & methods).

- ❖ The “parent” class (aka “superclass” & “ancestor”) defines all of the common features.

- ❖ The “child” class (aka “subclass” & “descendent”) inherits ALL of the features defined by the superclass
  - includes ALL methods
  - includes ALL data values

- ❖ A subclass CAN override inherited methods and data values (“specializes” the subclass)

- ❖ A subclass CAN add more methods and data values

- ❖ A superclass can have one or more subclasses, but a subclass can only have one superclass

- ❖ Inheritance is NOT limited to one level; rather, a hierarchy can develop

---

Example: Applets

One creates an applet by creating a class derived from the Applet class. The programmer need only write code specific to the applet created.

Why Inheritance?

- Models the real world by modeling “...is a...” relationships

  - e.g., a student is a person

  - e.g., a car is a vehicle

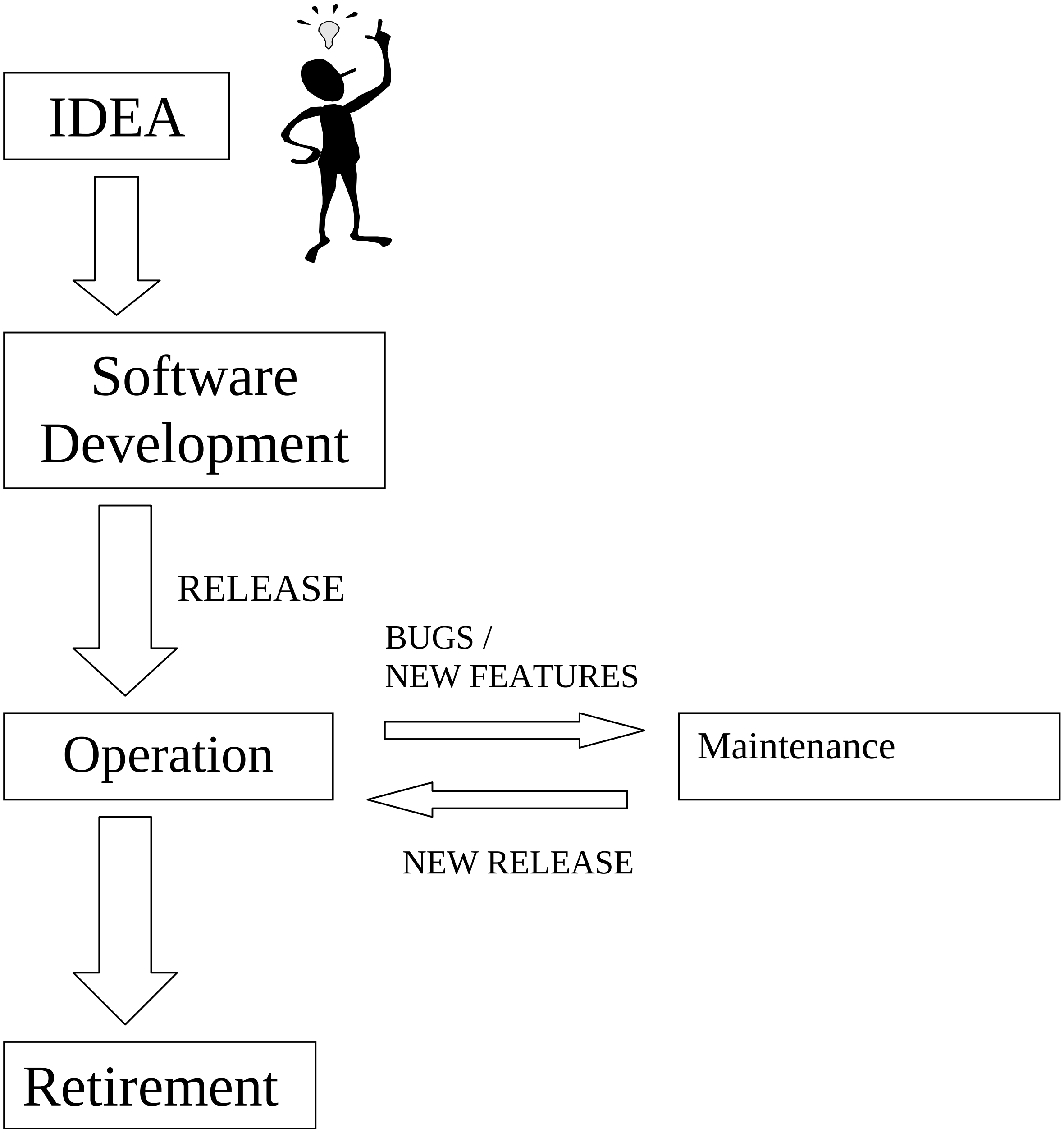
- Code Re-use: Allows one to add functionality to existing classes allowing for efficient program design

Software Engineering

The application of a systematic and disciplined approach to the development, testing, and maintenance of a program

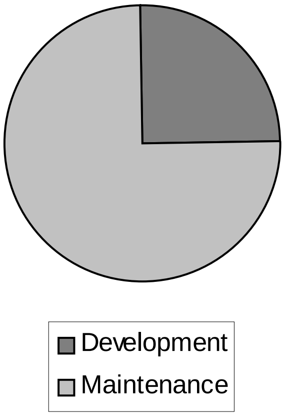
Software Life Cycle

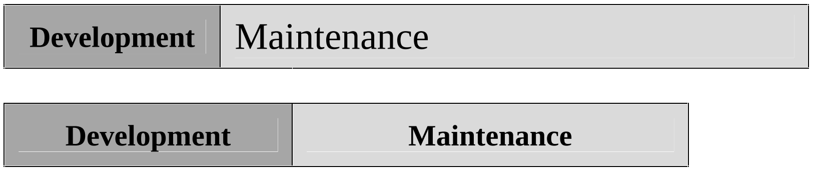
The sequence of stages from conception to operation of a program



Maintenance

Process of modifying program to enhance its features or fix its problems

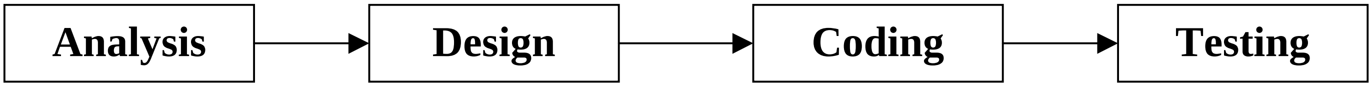




Those maintaining a program are most likely NOT those who created it. Therefore, it is essential for programs to be understandable by others.

The degree to which a program can be understood by others is directly related to how well it was designed, implemented, and documented.

Software Development



Analysis: Determine Feasibility (Is a solution possible?)
Specify Requirements (Describe features of the program)

Design: Establish a set of classes/objects to fulfill the requirements

Coding: Implement design into an actual program.
(Much easier with a well constructed design.)

Testing: Verify that the code meets the requirements

Two Types of Testing:

1. Unit Testing: verify each class works
2. Integration Testing: verify that the classes work together

Two Approaches to Testing:

1. Build & Fix: Coding a program, then modifying it until it reaches some level of acceptance.
2. Iterative: A mode of development where the stages can be revisited and new information is uncovered that affects the development.

Debugging: Eliminating errors from code.