Information Hiding

- **Black Boxes**
  - **Your Education**
    - Depends on your level: (and “Who’s asking?”)
- **Encapsulation**
- **Abstraction**
- **Object Oriented (OO)**
  - *Object* is often the *Black Box*
More on Classes

- Syntax of a Class
  
  \[ \text{accessSpecifier} \ \text{class} \ \text{ClassName} \ {\ }
  \text{fields} \\
  \text{constructors} \\
  \text{methods} \]

  - Have seen Chicken example in lab

- Syntax Fields (define the state) /Instance Variables

  \[ \text{accessSpecifier} \ \text{fieldType} \ \text{fieldName}; \]

  - Examples: (accessSpecifier should be private: WHY???)
    
    \[ \text{private} \ \text{double} \ \text{weight}; \]
    \[ \text{private} \ \text{String} \ \text{address}; \]
More on Classes

- Syntax of a Constructor
  
  ```
  accessSpecifier ClassName (parameterList)
  {
    constructor body
  }
  ```

  Example: (ObGynCase)
  ```
  public ObGynCase(String aName) {
    sex = 'F';
    name = aName;
  }
  ```

  ❏ May have multiple constructors.

- How do I know it’s a Constructor?
  ❏ Same name as class
  ❏ No return type (not even `void`!)
More on Classes

- Syntax of a Method

\[
\text{accessSpecifier \ returnType \ methodName(\text{parameterList})\{} \\
\quad \text{method body} \\
\\}
\]

- Examples:

```java
public int oldOdometer(int milesTraveled) {
    int milesDisplayed = milesTraveled\%100000;
    return milesDisplayed;
}
```

```java
public void resetTimer() {
    hours = 0;
}
```
Method Features (repeat)

- **Return values**
  - Methods can return information
    - Accessor methods require that
      - Have return type in header specifying type of info
    - Use: \( w = \text{chick}.\text{getWeight}(); \)

- **Parameters**
  - Methods may receive information thru parameters
    - Mutator methods usually require that
      - Method header includes parameter definition in parentheses
    - Use: \( \text{chick}.\text{newName}("Elsa"); \)

- **May have both parameters and return values**
Testing Classes

- **Use a `main` method and include test code**
  - Can have a main in every class definition for testing purposes
  - Must then specify *which* main to be used when compiling

- **Can use existing special test environments**
  - Will discuss APT system later.

- **Create your own Test Platform(s)**
  - Test platforms can be quite elaborate in large projects
Variable Categories

- **Instance Variables (fields)**
  - Belong to an Object
  - Are “known” throughout the class (scope)
  - Life is same as of the object

- **Local Variables and Parameters**
  - Belong to a Method
  - Scope is only that method (“know” only locally)
  - Life is while method is active (dies on return)
Primitive Types

- Primitive Types (base types)
  - Built-in data types; native to most hardware
  - Note: not objects (will use mostly first four)

  - boolean
  - int
  - double
  - char

- Constants (by example):

  boolean f = false;
  int i = 32769;
  double d = 0.333333;
  char c = 'x';

  byte b = 33;
  short s = 21;
  long l = 289L;
  float = 3.141592F;