Program (Source Code) Commenting Guidelines

(Version 1.2: last updated 9/28/2004)

In developing source code for this course, you should adhere a closely as possible to these specifications. Failure to do so may result in a project point deduction! (Remember, you are not just delivering a program that works, you are delivering your ideas. Be sure you know how to clearly communicate them via your source code!)

I strongly suggest modeling your commenting style after the JavaDoc style (http://java.sun.com/j2se/javadoc), though you likely do not need to produce JavaDoc documentation for my projects (unless the project specification explicitly states that you do!). I also highly recommend using a source code pretty printer to clean up your formatting (such as Jacobe or Checkstyle for Java).

The specifications are broken into sections or levels.

File level:

- At the start of each file in the project, a comment should indicate the programmer's name(s), date of submission, course the project was developed for, and your instructor's name.
- If file versioning was used, the version number of the file should also be included, along with the corresponding check-in date of the file.

Class level:

- Before the start of each class, a block of comments should clearly indicate:
  - the name of the class with a summary of the major functionality, and
  - the data the class encapsulates.
- Inside of each class, each instance or static variable should have a corresponding short comment as to their purpose.

Method level:

- Before the start of each method (not inside of it), a block of comments should clearly indicate the following:
  - the name of the method with a summary of its major functionality,
  - a listing and description of each formal parameter for the function,
  - a description of what this method returns,
  - a listing of any exceptions this method throws and under what conditions, and
  - a reference or citation of algorithms borrowed/stolen/copied from other sources (such as course textbooks, on-line references, etc.).

- Within each method there should be:
  - a brief comment for every local (method) variable declared describing what it does, and
  - a short comment should precede blocks of code which represents tasks and sub-functionality.

Example: (Does not illustrate all of the concepts mentioned above, but should give you a general idea of what I'm looking for. Please note the clean indentation style, and two blank lines between methods help with the human readability of the entire file.)

```
/**
 * Programmer: Michael DePasquale    Date Submitted: 9/10/2004
 * Instructor: Dr. DePasquale        Course: CS230
 */


/**
 * The Product class encapsulates the representation of a store product in
 * our system which we carry in our store.  The class contains simple data
 * types and Strings to store the product's category, name, cost, price, etc.
 *
 * @author  Michael DePasquale
 */
public class Product {

   /**
    * The name of the product's category.
    */
   private String category;

   /**
    * The name of the product.
    */
   private String name;

   /**
    * The quantity of the product currently in stock.
    */
   private int quantity;

   /**
    * The number of this product sold to date, excluding
    * the current number in stock.
    */
   private int numSold;


   /**
    * The cost of the product (cost we purchase it at).
    */
   private double cost;


   /**
    * The price of the product (price we sell it at).
    */
   private double price;
```

```
/**
 * Default constructor, performs no action.
 */
public Product () {
}


/**
 * Creates a Product object from the specified data.
 *
 * @param  category  String name of this product's category.
 * @param  name  String name of this product.
 * @param  quantity  Quantity in stock of this product.
 * @param  numSold  Number sold of this product to date.
 * @param  price  The price the product is sold to the consumer.
 * @param  cost  The cost at which we purchase this product.
 */
public Product (String category, String name, int quantity,
          int numSold, double price, double cost) {
   this.category = category;
   this.name = name;
   this.quantity = quantity;
   this.numSold = numSold;
   this.cost = cost;
   this.price = price;
}


/**
 * Returns the category name of the product.
 *
 * @return the category name.
 */
public String getCategory () {
   return category;
}



/**
 * Returns the name of this product.
 *
 * @return the name of the product.
 */
public String getName () {
   return name;
}


/**
 * Returns the quantity in stock of this product.
 *
 * @return the quantity in stock.
 */
public int getQuantity () {
   return quantity;
}
```

```
    /**
     * Returns the number of this product sold to date.
     *
     * @return the number sold.
     */
    public int getNumberSold () {
       return numSold;
    }


    /**
     * Returns our cost of this product.
     *
     * @return the cost of the product.
     */
    public double getCost () {
       return cost;
    }


    /**
     * Returns the price (to the consumer) of this product.
     *
     * @return the price of the product.
     */
    public double getPrice () {
       return price;
    }


    /**
     * Returns a string representation of this product.
     *
     * @return a representation of the product.
     */
    public String toString() {
       return "(" + category + ") " + name + " qoh=" + quantity +
              " numSold=" + numSold + " price=$" + price +
              " cost=$" + cost;
    }
}
```