

```
import java.util.*; // for Arrays.toString

/** (Note the 2 stars here. Below is a Javadoc style CLASS COMMENT.
    The REQUIRED first few lines below explain what the class/program does.
    You must remove all my parenthetical explanations from your programs.)
 * Template is used to demonstrate the following:
 *   Javadoc style comments
 *   REQUIRED information you must include to get credit for submitted work
 *   If time allows, the command line compile and execute process
 *
 * @assignment Tel - compile and run (REQUIRED: Either TE# for team exercises or SP# for
solo programs and title)
 *
 * @author      Henry Jekyll (REQUIRED)
 * @author      Edward Hyde (use only one author for solo programs, both for team work)
 * @version     3.0 Updated 31 december 2009 (REQUIRED: an accurate date)
 *
 * @course     CS&141, Winter 2010, Shoreline Community College (REQUIRED)
 *
 * @see ch. 1, p.10, ex # (REQUIRED: if assignment refers to a chapter#, page# and/or
exercise # )
 * @see Study Appendix C, p.857 (Use @see only if needed for your program)
 * @see http://java.sun.com/j2se/javadoc/writingdoccomments/
 */

// leave a blank line here and before each METHOD COMMENT
public class Template {

    /** (Javadoc style METHOD COMMENT: first few lines must briefly say what the method
does.)
 * (The main method must clearly show the overall design structure; for example, this
one will:)
 * Display the author and command line used to run this program
 *
 * @param args the command line used to start this program is just printed
 * @return (do not include this if there is no return statement)
 * @throws (do not include this if there is no exception thrown)
 */
    public static void main( String[ ] args )    {
        // print the name then the arguments ( this is a redundant comment !)
        System.out.println("Robert Louis Stevenson Shields");
        System.out.println("args=" + Arrays.toString(args) );
    } // main
} // class

// Using good identifier names will reduce the need for comments.
// You must add comments to clarify why code does something that may be confusing.
// Do not just re-state the code; explain why you do it in this particular way.
// Never add redundant comments such as the one above (and often seen in textbooks).
```