Document Your Work
And make your work readable

Asst. Prof. Reginald Neil C. Recario
rcrecario@up.edu.ph
rncrecario@gmail.com
Institute of Computer Science
University of the Philippines Los Baños

Document your work...

- This might be the nth time that a teacher told you to document your work.
- This might also be the nth time you are asked to make a readable, concise and effective code.
- Writing code do not differ with writing a poem, article, novel or even a technical paper.
  - Follows a format
  - Rules of grammar, syntax and semantics
  - Clarity and effective communication

Document your work…

- Why is documentation important?

Document your work…

- Some Tips in documentation and making a readable code
- Here is a summary of the very important programming style tips from Brian Kernighan's 1994 guest CS50 lecture.

Document your work…

- Say what you mean, simply and directly.
  - Make a comment as simple and short as possible.
- Use the “telephone test” for readability.
  - What do you do with a telephone?
  - How do you verify if you were understood?
- Write clearly - don't be too clever.
  - Avoid too much technical term more than necessary.

Document your work…

- Don't use conditional expressions as a substitute for a logical expression.
  - Example: newValue = ((oldValue && constantK) || pValue)
- Parenthesize to avoid ambiguity.
  - Example: if(!k) if(k == f) if(k == p) else k+=oldValue;
- Each time you make a test, do something.
  - if(k == oldValue){ alert("Why do you hate me lab teacher?"); }

Document your work…

- Follow each decision as closely as possible with its associated action.
- Use the good features of a language; avoid the bad ones.
- Capture regularity in control flow, irregularity in data.
- Each module should do one thing well.
  - A module should do only one thing
- Make sure comments and code agree.

Document your work...

- Don't just echo the code with comments - make every comment count.
- Don't comment bad code - rewrite it.
  - Example: // k = k+54 (oldValue - pValue);
- Use symbolic constants for magic numbers.
- Watch out for side effects and order of evaluation.
- Macros are not functions.

Document your work…

- Watch out for off-by-one errors.
  - Example: for(i=0; i<10; i++)
  - Example: for(i=1; i<=10; i++)
- Test programs at their boundaries.
- Program defensively.
  - Practice defensive programming
  - Catch all possible cases of committing an error or violation
- Make sure input cannot violate the limits of the program.
  - Example: having an 8-bit sum stored in a 4-bit storage

Document your work…

- Make it right before you make it faster.
  - Make sure it works first before optimizing
- Keep it right when you make it faster.
  - Make sure the program does what it should do when optimized.
- Don't sacrifice clarity for small gains in “efficiency”.
- Don't stop with your first draft.

Document your work…

- Sample comment block:

```
/** * Short Description *
* Optional longer description or discussion that may contain
* inline tags and some html markup. See the sections below for
* more details on the possible tags and markup. Separated by
* blank lines, this is used in page-level DocBlocks and in
* element-level DockBlocks when the element merits further discussion.
*
* @since version number
* @param type argument for a function and what it takes as input
* @return type what it returns
*/
```

References:

- http://cboard.cprogramming.com/c-programming/90154-how-properly-document-program.html
- http://www.open-emr.org/wiki/index.php/How_to_Document_Your_Code_Properly