General Guidelines

- Put your documentation in a plain text (ASCII) file called README and submit it in class and as part of your electronic submission.
- Graphics should be attached to the hardcopy submission, but should be omitted from the electronic submission. Use whatever tool you want for graphics.
- Be brief. I should be able to read and understand the main part of the documentation in 10 to 15 minutes.
- Put only the primary details in the body of the documentation. Put additional information in appendices.
- The sections (and subsections) of the documentation are shown in boldface below. When possible, the approximate length of the sections are given next to the section name.

Organization

1) Status (1 paragraph)
   - State explicitly if your program is bug free or not. If not bug free, list the problem(s). If you have clear conjectures as to the causes of the problems, list them.
   - List the major tests that were successfully conducted (i.e., what do you think really works?).
   - Do not describe the details of the tests (and failures) here. Describe them in the Program Testing section.
   - e.g.:  
     *This program has no known bugs. The most demanding test (called XXX below) demonstrates that all of the major features are fully functional. Furthermore, I have added and tested the capability to XXX.*

2) Overview
   - Shar Inventory: A list of files grouped by type (C/C++ code, test scripts/code, input files, output files, ....
   - Build: Where did you build your binary and what command did you use? e.g.,
     *I built the binaries on hilton.cec using g++. 'make all' will build the binary B and the master test script XXX.*
   - Usage: What is the syntax required to run the binaries and major shell scripts and what is the purpose of each binary and script?

3) Program Description
– **Overview (1 page)**: Briefly describe the overall control structure of your program and the key data structures/classes employed.

– **Key Data Structures and Classes**: Provide additional details especially the relationship between the key structures (data and control).

– **Usage Examples**: Give usage examples for the major functions (methods).

4) **Program Testing**

– **Overview**: What was your testing procedure and what major tests did you run?

– Why should I be confident that your program runs correctly?

– **Test Results**: Include the output from key test runs and describe what features are exercised.

5) **Experimental Results**

– **Experiments**:
  * Where did you run the experiments (e.g., system, OS)?
  * What experiments did you run?
  * How did you measure time?

– **Results (Presentation/Discussion)**:
  * A picture (graph) is worth a thousand words.
  * Do your results make sense? Did you get the results you expected? If not, present possible explanations?
  * Are your results reproducible? Are they stable? For example, a common error is to run the program for too short of a period.

6) **Extra Credit and/or Enhancements**

7) **Appendix: Source Code Listing**

– **Table of Contents**: Provide a list of the attached files

– **Listings**: Attach the printouts in the order specified in the Table of Contents.

8) **Appendix: XXX**

– Put other details in additional appendices as needed.