Computer Science I

Code Rubric

Department of Computer Science & Engineering
University of Nebraska–Lincoln

These are the rubric guidelines for the course. In general, code is assessed based on 4 general categories: Style, Documentation, Design, and Correctness. In addition, for your submission to be considered, you need to ensure that:

- All required soft-copy files are handed in via webhandin
- You use correct file name(s) and organization
- Programs successfully compile and execute using the webgrader

The four categories include but are not limited to the following items.

Style

- Appropriate variable and function/method identifiers
- Style and naming conventions are consistent
- Good use of whitespace; proper indentation
- Clean, readable code
- Code is well-organized

Documentation

- Well written comments that clearly explain the purpose of each non-trivial piece of code
- Comments explain the "what" and "why"
- Comments are not overly verbose or overly terse

- Code itself is “self-documenting”; it explains the “how”

Program Design

- Code is well-organized and efficient
- Code is modular; substantial pieces of it could be reused; few redundancies
- Code is easily understood and maintainable
- It is clear that sufficient testing has been performed
- Corner cases and bad input have been anticipated and appropriate error handling has been implemented

Program Correctness

- Source code compiles and executes as expected
- Program runs as specified: correctly reads any input; correctly formatted output
- Test cases successfully execute

Hack Guidelines

Each hack is worth 25 points distributed as follows.

Category	Points
Style	2.0
Documentation	2.0
Design	5.0
Correctness	16.0

Specific point deductions guidelines follow. Each of these items will result in a point deduction.

Style

- Significant improper or inconsistent use of whitespace
- Significant improper identifier naming or inconsistent naming conventions

Documentation

- Missing header documentation
- Substantial blocks (functions, complex code) are not properly documented
- Overly verbose or useless comments

Design

- Compiler warnings have not been addressed
- Dead or extraneous code remains
- Insufficient error handling regardless of webgrader behavior
- Extraneous or unnecessary output (debugging or error statements)
- Redundant code
- Improper or incorrect patterns, variable types, etc.
- Contains obvious memory leaks or misuse of data types

Correctness

- Output is not reasonably readable
- Output does not report as much information as expected

In addition, the remaining points for correctness are awarded proportionally for how many test case(s) pass.

Assignment Guidelines

In general, the same deductions will be made on assignments. Assignment points are awarded proportionally to the number of parts of the assignment.