Computer Science I

Code Rubric

Department of Computer Science & Engineering
University of Nebraska–Lincoln

Fall 2018

,

This is a draft of items/elements that will be included in rubrics for assignments and hacks.

General Guidelines

Following Instructions

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

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. Specific point deductions guidelines follow. For each category, once all points are lost, no more points should be deducted for that category (the minimum grade per category is zero). However, all errors should be noted in the rubric comments.

Category	Points
Style	2.5
Documentation	2.5
Design	5.0
Correctness	15.0

Style

- Proper use of whitespace
  - Some (at least 2-3 instances or lines) of inconsistent whitespace style: -0.5
  - Frequent misuse of inconsistent use of whitespace: -1
  - Extremely poor use of whitespace leading to unreadable code -2
- Variable naming
  - Inconsistent naming convention(s) of variables, functions, etc.: -1
  - Non-descriptive, cryptic, or inappropriate variable naming: -1

Documentation

- Missing “header” comment with author/date/etc. in all user-authored files
- Overly verbose or unnecessary comments: -1
- Some function(s) lack proper documentation: -1

Design

- Compiler issues warnings that have not been taken care of -1 to -5 depending on severity
- Dead or extraneous code has not been removed, large sections of code are commented out but not deleted -1 to -5 depending on the severity

- Sufficiently reasonable error handling has been omitted regardless of how the webgrader runs -1 to -3 depending on severity

Correctness

Generally, points are awarded proportionally according to the number of test cases. Points are lost for each test case such that:

- The reported output (numbers) differ significantly from the expected output
- Formatting details need not match exactly, but must report *at least* as much information as the expected output (but *may* give more or may give alternative formatting).
- Output must be well-formatted and readable.
- Proportional credit given in the case of automated test suites
- If no substantial code is present but (some) test cases pass, credit should still not be awarded.

Assignment Guidelines