Peer Evaluation

One of the goals of this course is to encourage good software development practices, especially when building a large software system (such as a compiler). For each project submission, you will be assigned two other random students in the course. You must review their code and offer constructive feedback. In industry, this process is known as a *code review* and is frequently used to improve software quality and to catch software defects early.

It is important to note that the goal of a code review is to improve the quality of the code, not necessarily to evaluate the quality of the developer. Accordingly, try to use "I-messages" (e.g., "I don't understand it" rather than "it's confusing"), and ask questions and make suggestions instead of making accusations. This document lists the questions about the code that you should answer for each submission, giving examples of both *non-constructive* and *constructive* feedback (aim for the latter!) for each question.

Question	**Example of Non-constructive Feedback**	**Example of Constructive Feedback**
What did you like about this submission?	"It was good." "It was clean."	"The main loop in function X was clean and well-documented." "I liked the use of extra indentation to line up the array initializations in module X."
Describe one significant difference between your own submission and this person's submission. Which approach is cleaner? Which approach is more efficient?	"My code is faster." "My function X is shorter than this author's version. Mine is better."	"I chose to calculate the maximum value on every iteration of the outer loop while this author calculated it only once and cached the result. Their approach is more efficient, but my approach works even if the list is modified during iteration."
Is the code well-formatted and well-documented? If not, suggest some specific improvements.	"Not enough documentation." "I couldn't understand the code."	"The goal of function X was unclear; the author should add some documentation regarding its inputs and outputs." "The code is inconsistently formatted; the author should consider removing the extra empty lines in functions X and Y."
Did you find any software defects? If so, briefly describe them.	"Function X doesn't work." "I couldn't get it to run."	"Function X does not produce the correct output for this input: 'ABC'" "The program crashed with an IOException when I tried to run it on 'loops.decaf'."
Do you have any other constructive comments for the author?	"This code sucks and needs to be rewritten." "This code is perfect."	"Function X has redundant if-conditions; the last two could be consolidated." "The use of recursion in function Y to avoid ugly class-level data structures is very elegant."

Checklists

As you work on the project, keep this review process in mind--your code will be reviewed by at least two people in addition to the instructor! Accordingly, here is a checklist you should use before you submit:

SUBMITTER:

- ☐ The code compiles and has been tested with unit and integration tests
- ☐ The code is clean and follows a consistent coding style
- ☐ All corner cases, workarounds, and non-obvious code have been documented
- ☐ All dead code has been removed

Correspondingly, here is a checklist you should use to help guide your efforts as you do your peer review and answer the questions on the previous page:

REVIEWER:

- ☐ Does the code compile? If so, does it pass all of your tests?
- ☐ Is the code is generally understandable and appropriate?
- ☐ Has the author included appropriate documentation?
- ☐ Are exceptions and corner cases handled appropriately?
- ☐ Has repetitive code been factored out?
- ☐ Have frameworks and existing code been used appropriately?
- ☐ Does the functionality fit the design/architecture of the overall project?

Code reviews should be at least two or three paragraphs of 3-4 sentences each. Include as much detail as possible, and if you have suggestions for improvement be explicit.

Submit your code review on Canvas, although you may wish to prepare your comments using a text editor or word processor beforehand to reduce the chance of data loss due to browser issues. Submit your code reviews on the appropriate assignment, but also make sure you send each review directly to the (correct!) author using a Canvas inbox message (CC'ing me as proof of completion).

Keep in mind that your review will also be compared with another person's review and my own assessment. In addition, your code reviews will be graded on the following scale:

Score	**Description**
4	Thorough and constructive criticism on all points.
3	Constructive feedback for most points, but not thorough.
2	OK, but lacking (usually because of a focus on cosmetic style issues over deeper issues).
1	Mostly non-constructive or inconsistent written feedback; very superficial.
0	Nothing.