CMSC 143: Introduction to Object-Oriented Programming with Robots

Lab 9: Code Review & Critique
Due November 9, 2009

In this lab, we will practice our code reviewing and critiquing skills. Specifically, we will review our special effects programs from the latest assignment. One person from each team should each submit a copy of your reviews as text files named: `codeX_review.txt`. You should start with the original python file and add the review as a set of comments at the end and throughout the python file.

Learning Objectives

- Critically Evaluate Programs
- Provide Feedback on Design and Implementation
- Apply Feedback to Improve your Programs

Ground Rules

- **Don't be mean**
- **Be constructive.**
- **Don't take the reviews personally.**

Part One – Critique

Each group should review five special effects programs (they are labeled codeX.py). You should not review your own programs. You can download the zip files of all the special effects on moodle. As a group, you should perform a code review and critique. Each team should complete the following steps for each program:

1. Become familiar with the other group's program. What are the primary functions? What are they intended to do? At this point, don't dive into the code, only use functions names and comments. You are trying to uncover what the program is **intended** to do, not what it actually does.
2. Use the team's program with their test images, and then with your own test images. What are the results, does the program run effectively for both sets of test images?
3. Carefully review the program. In particular:
   (a) Does the program work correctly? Under what conditions does it it fail?
   (b) Why do you think the authors implemented the special effect in the manner they did?
   (c) Comment on each of the following points. Cite a specific example or two from the program.

Clarity	How clear is the program? Is the program well-structured? Does it use clear variable and function names?
Modularity	Is the code written in a modular fashion using functions and modules? Or is there a lot of repetition?
Succinctness	Is the code short and to the point, "Is the code as simple as possible, but no simpler?"
Requirements	Does the program accomplish what it is supposed to?
Efficiency	Does the program do just the work that is necessary, or many unneeded operations?
Error Handling	Does the program handle errors in a a reasonable way?
Documentation	Is the code effectively commented?
Tests	Does the code include useful tests?

Part Two – Applying Feedback

Read the review comments from your evaluators and write a short paragraph reflecting on their review. Next, apply at least two of their recommendations toward your program. Write a description of your improvements.