Final project grade breakdown and comments

1 message

From: Erik Talvitie <erik.talvitie@fandm.edu>
To: colin.poindexter <colin.poindexter@fandm.edu>, guwanjith.tennekoon <guwanjith.tennekoon@fandm.edu>
Date: Wednesday, December 15, 2010 11:16:24 AM
Subject: Final project grade breakdown and comments

Here is the grade breakdown for your final project, as well as some brief, high-level comments. If you have any questions, or would like more detailed feedback, don't hesitate to ask.

Erik

P.S. Note that on Blackboard your project grade is spread across two columns, one for the proposal (worth 1 point) and one for the rest (worth 9 points)

Colin & Guwanjith

Total: 9.05/10

Proposal & Meeting: 1/1

Presentation: 1.35/1.5
+ Engaging presence
+ Fun and interesting demo

- Not enough on programming issues
- A few key features missing in action
- Detailed answers to questions, but did not clarify for audience members unfamiliar with your project.

Write-up: 1.4/1.5
+ Detailed and clear

- No real discussion of challenges faced, testing methodology, or external sources

Project Code: 5.3/6
+ Menu mode options work well
+ Output is nicely formatted, prompts are logical and sensible
+ Conversation mode responds to many reasonable queries
+ A good start on learning to interpret unrecognized queries

- Input handling is not very robust to user error ('y' interpreted as 'no', errors if non-numbers given in menu mode, etc.)
- If you use menu mode once, you end up in conversation mode with no easy way back
- While learning works well with one-word queries, it behaves strangely on unrecognized multi-word queries. It seems to each word as a query and deals with each in turn, which is confusing to the user (me).
- Various functions asking for user input all call each other (menu calls secondary which calls convo which calls menu which calls menu which calls secondary which calls…). This makes the code difficult to follow, and causes problems like the one above where a user gets multiple queries in a row, even after they've said they're done. A better design would have been to have one centralized interaction loop that calls the various functions to get user input, which then *return* the relevant information, so the central loop can decide what to do next.
- A good deal of code duplication. If you write two functions that are identical except for one line, you should think seriously about how to consolidate them. Similarly, if you ever write the same code in both the "if" and the "else" part of a conditional block, there's probably a way to write that code just once outside the conditional.
- A class may not have been the right design tool here. If once you've thought it through you think there really only needs to be one object of one class in your whole program, maybe there shouldn't be a class at all: and instead just a collection of functions.