CMPUT 174 REVIEW

Andrew Li

Fall 2019

1 Abstract

This is a review document along with some tips and tricks for CMPUT 174 at U of Alberta. First, I'd like to claim that this review may not be relevant to any future classes or teachers. This will be a review of classes, labs, and exams under Dr. Geoff Hollis. Briefly, this class will go through the basics of programming and the pygame module.

2 Background

This review may be a bias review of 174 because I have been a python programmer, however, if you have programmed in other languages, i.e. Java, C++, JavaScript, Shell, etc. this review should be the same. However, if you have not programmed, it is hard for me to gauge the difficulty of this class and its contents. If you are reading this before taking this class, get familiar with python and experiment (more resources in Appendix) with python. If you do not know python, but you are familiar with other programming languages, take some time to familiarize yourself with the syntax because python is not a conventional C-styled language like Java, C or JavaScript (again Appendix). Also, welcome to python from the python community.

3 Classes

These are optional. I attended every class only to work on homework or other homework not relating to the class. If you know python, the content will seem ~~stupid~~ trivial. The class is intended for beginner with no understanding of python or any other programming languages.

4 Homework

The only 'homework' is Vignettes 1-10, with are a online assignment with videos as guides. These will go over flowcharts, tokens, syntax, semantics, the python interpreter, functions, logic, data structures, (i.e. lists), and classes. This does take a lot of time (mostly to watch the videos) but there are not too challenging. If it appears too easy, take the videos at 2x speed or read the transcript or simply jump into the quiz (not recommended).

5 Labs

These are times (3 hours weekly) allocated in a computer lab to test your application. It will start with a design document (only for the start of each project), followed by preparation questions, the actual coding, then the review questions (optional). Usually labs projects are separated into multiple steps. I recommend doing the labs at home so you can zip out of lab after showing the TAs. The design document as you will learn is simply an outline that may not be necessary for the first few projects if you are a non-noob programmer, however, you still have to do them. The coding is what I usually do first, then I work backwards to finish the design document and the preparation questions. The review is optional however, I suggest you at least look at it.

Time breakdown for projects: 10:20:70 =>design document: prep questions: coding

5.1 Lab 1: Guess the Number

Time: <0:35 min. Simple program to get you into python

5.2 Lab 2: Word Puzzle

Time: <1:25 min. Display a bunch of words and select a word so the player can guess a word

5.3 Lab 3: Pong

Time: <3:15 min. Recreation of pong

5.4 Lab 4: Memory

Time: <6:15 min. A match tile game. Requires operator overloading

** Hint use tostring comparisons

6 Exams

Midterm and final are multiple choice and are similar in style to the vignettes; i.e. looks of application question and fill in the blanks.

- The midterm is 90 minutes (1.5 hours) and it averaged (mean) around 31/45. There where 45 multiple questions.
- The final is 180 minutes (3 hours) and it averaged (mean) around ??/85. There where 85 multiple questions. This final seemed to be easier than the midterm.

A Appendix

Helpful guides

- Pygame
- Lynda.com - Available if you have a library card
- YouTube - You can look around for more videos on YouTube