You must complete the following task by 11am on 11/30/12.

Your solutions to the homework must be committed to your Subversion repository in a sub-directory HW07.

You will commit your files as follows:

i. Use \LaTeX to format your homework.

ii. Submit the following files HW07/hw07.pdf, HW07/hw07.tex.

iii. Your source code files, resources, and Makefile for the game should be placed in a sub-directory HW07/game

iv. Submit the documentation directory created by Doxygen as HW07/game/man

You must be ready to demo your game on Friday 11/30 at 11am and class attendance is mandatory on that day. All students will rate each game. Up to two Raspberry PI [web] will be awarded [although presentation may be delayed].

You may work on your own or with one other student. You must manage your source code in Subversion (collaborative or individual). We will check the logs to make sure that you really did do this.

You may consult the internet, gaming, and programming texts. You are encouraged to do extensive research playing your favorite computer, mobile, and console games.

---

1But do remember to cite all contributing sources.
Q1 Purpose: combine your creativity with the skills covered in CAAM 420. It is an opportunity for you to demonstrate mastery of project collaboration and source code management (svn), C++ (g++), debugging (gdb and valgrind), profiling (gprof), building C++ objects, container classes (stacks, lists, vectors, maps etc from STL or hand rolled), graphics (OpenGL), interactivity (GLUT/GLU), documenting your project (LATEX), citing literature (bibtex), and documenting your code (doxygen).

Build a game. The specifics of the game are up to you (think “Angry Birds 3D”, “PacMan”, “Tron”, “Cubert”, “Driver”, “Elite”...). You should exercise all of the tools mentioned above. You may use the class examples and small amounts of code from the web, but it is your responsibility to perform the bulk of the game implementation.

i. [20 points] We will review your game implementation for readability, style, playability, robustness, and to make sure you are indeed using C++ classes.

ii. [10 points] Use OpenGL for graphics.

iii. [10 points] Run your code with valgrind and include the output in a verbatim environment or use \verbatiminput. Explain any errors that you were not able to resolve.

iv. [10 points] Profile your code with gprof and include the output. Discuss how you used the profiling information to identify bottlenecks, and whether you could use the profiling information to help refactor your code to improve performance.

v. [10 points] Include your C++ source code and header file(s) in the LATEX report using the lstlisting environment provided in the listings package.

vi. [10 points] Make sure your game is well commented with Doxygen style comments. Use Doxygen to generate a manual for your symbolic class. Add and commit the Doxygen manual to your HW06/Q1 svn directory.

vii. [10 points] Include screen shots from your game in your report.

viii. [10 points] Other students can try your game during class 11/30/12.

ix. [10 points] Use STL container class objects to manage your game objects.

x. [Extra credit (max 150 points)]: You can gain serious extra credit for this project. We will give credit for elastic deforming objects, non-static object geometries, “physics”, electrostatics, gravity, heat, lasers (because everything is better with lasers), clever collision detection, neat graphics, neat polyhedral objects, using OpenGL textures, clever game idea, network based multi-player features, help screens, maps, ...

“Tic-tac-toe” and “The Oregon Trail” are not an option.