# CPSC 386 — Game Dsgn — Final Project

**Final Project: New Video Game** [after Prof. Michael Shafae's version]

**Introduction**

This is the final project for the course. The objective of this assignment is to create a non-trivial video game of your own choosing. The final game must be a well conceived and implemented game that aims to be challenging and engaging for the target audience.

A premium is placed on having a well documented and well conceived design. According to the syllabus, the final project will account for 10% of your grade. The final project is judged not only on the completeness of the solution but also on the quality of the student's understanding of the project as demonstrated by a written project summary, and an in class presentation and demonstration of a working video game program.

**Teams**

The final project may be built either individually or in a team of two class students. If it is a team of two, then itemize what each member was tasked with doing/building in your project report.

**Weighting of Project Issues**

- 50% for a playable game
- 25% for a project summary, design report (with screen shots), and a README.txt
- 20% an in class presentation and demonstration
- 5% for following Submission Rules

Your code is expected to be well organized and well documented. Points will be deducted for poorly structured or undocumented code.

**Prerequisites**

In order to complete the exercise successfully, at a minimum, you will need to have the following:

- Python interpreter
- A text editor
- Pygame module
- Distutils

Students are encouraged to use the pygame module for graphics rendering and mouse/joystick input. If you are familiar with another GUI toolkit and would like to use it for this assignment, please seek the instructor's consent. Students with very little programming experience can seek permission to create an interactive text/terminal shell game.

Although not required, students are encouraged to create an installer such that the game program and its resource files can be used by a non-technical computer user. For the MS Windows platform, the python program can be built into a binary using py2exe (www.py2exe.org). In turn, the binary and its resource files can be built into an installer using freely available tools such as WIX (wix.sourceforge.net) and NSIS (nsis.sourceforge.net). Alternatively, using Python's **Distutils**, an MS Windows MSI file can be created.

Students using Apple Mac OS X are strongly encouraged to create an application bundle, otherwise known as a .app, and a Mac OS X metapackage, a .mpkg file. Mac OS X metapackages are similar in behavior to MS Windows MSI file. A Mac OS X metapackage can be built using the bdist_mpkg package (pypi.python.org/pypi/bdist_mpkg). Using py2app (svn.pythonmac.org/py2app/py2app/trunk/doc/index.html), the program and its resource files can built into an application bundle.

At a minimum, the game must have a user interface and be interactive. Students are strongly encouraged to incorporate sound, graphics, and human input devices other than keyboards.

**Readme File**

The README text file must describe the project's scope, instructions for building, instructions for use and any known bugs. Be clear in your instruction on how to build and use the project by providing instructions a novice would understand. The README must also list any external dependencies and how to find these dependencies.

A README would cover the following:

- The Program
  - Your Name
  - Contact info
  - Class and Assignment
- Intro
- External Requirements
- Setup and Installation
- Rules
- Features
- Bugs

**Project Summary and Design Report**

A project summary and design report is a short document that is an extension of your original game design document. A softcopy included in the zip archive of your submission is sufficient; no hardcopy needs to be provided.

The final project is judged not only on the completeness of the solution but on the quality of the students understanding of the project as demonstrated by a written project summary. The project summary, at a minimum, must be cleanly typeset, easy to read and properly presented.

A good summary and design report would cover the following topics:

- **Introduction**

  What is your project? Describe in detail what your project is so a layperson can understand the objective, as well as the utility.

- **Design**

  Detail the rules of the game, how it is played, example scenarios. Provide enough detail such that a savvy person can create a software architecture from this information. Factors to consider while writing your game design are:

  - Rules
  - Sources of uncertainty
  - Win state, loose state
  - Expected skills the player must have prior to starting the game
  - Controls
  - Expected duration of a game
  - Scoring
  - Visual representation of game state

- **Software Architecture Detail**

  The software design you used for your program. Provide details such as algorithms used, organization of the software, the classes/objects (each with title, responsibility, and their collaborating classes), and their major functions/methods, etc.

- **Game Demonstration**

  Detail what was implemented and how. Use screen shots to illustrate your results.

- **Bibliography**

  Write a bibliography which details the resources you used to understand and accomplish this project. Give credit where credit is due.

**Project Presentations and Demonstrations**

On the last week of class, you will present your Final project to the class. You should come prepared to talk about your final project and demonstrate it. Your presentation should be no more than 5 minutes long.

Be prepared to answer questions during and after your presentation. During your presentation, it is suggested that the following topics be covered (the timing is a suggestion, only).

- Title of your game and Who you are – as shown on your opening splash(AKA entry)-screen
- Brief overview of the game (~ 1/2 minute) – including the game type
- Algorithm, implementation, display (~ 1/2 - 1 minute)
- Project demonstration (~ 2 - 3 minutes)
- Thank the audience

Your project must be demonstrated in class during your presentation. Should you require specific technology or materials to accomplish your demonstration please see the instructor to make arrangements.

Feel free to invite people to attend the last week of class to watch your presentation as well as others. Other members of the CS community may be invited to attend the final project presentation.

**Academic Rules**

Correctly and properly attribute all third party material and references, lest points be taken off.

**New Submission Rules**

Your submission must, at a minimum, include a plain ASCII text file called **`README.txt`** (e.g., title, contact info, files list, installation/run info, bugs remaining, features added) all necessary source files to allow the submission to be built and run independently by the instructor. [For this project, no unusual files are expected.] Note, the instructor doesn't necessarily use your IDE.

All source code files must include a comment header identifying the author, author's contact info (please, no phone numbers), and a brief description of the file.

Do not include any IDE-specific files, object files, binary executables, or other superfluous files.

Place your submission files in a **folder named** `XXX-pY_lastname-firstinitial`. Where `XXX` is the class course number (e.g., 123 for course CS-123) and Y is the project number (eg, 9 for Project #9) For example in CS-123 for Project #9, if your name were Tim Cruise, then you would use

```
123-p9_Cruise-T
```

Then zip up this folder. Name the .zip file the **same as the folder name**.

The project is due by the time and on the due date specified in the class's bulletin-board project assignment post. Turn in by **sending me email** (see the Syllabus for the correct email address) with the zip file attached. The email subject title should also include **the folder name**. [NB, If your emailer will not email a .zip file, then change the file extension from .zip to .zap, attach that, and tell me so in the email.] Please include your name and campus ID at the end of the email (because some email addresses don't make this clear) – and if it's a team effort then include everybody on the team.. If there is a problem with your project, don't put it in the email body – put it in the README.txt file. Do not provide a link to Dropbox, Gdrive, or other cloud storage.