Project 2: Group Game

The aim of the group project is build a second, somewhat more complex arcade style game from the ground up, in a team setting. Groups of 2-4 members work best, with 3 often being an optimal number.

For this project, you will have roughly 5-6 weeks to go from concept to final product. My expectation is that this should be enough time to perform a **solid** implementation of a game akin to Gauntlet, Super Mario Brothers or Legend of Zelda (obviously, there's no need to have 40 hours of game play here). If you want to do a game that is relatively light on graphics, (say poker, or backgammon) that's probably ok too, but I will want to know in your design document where your effort is being placed (perhaps networking and AI).

Each game should also have one significant technical showpiece. In general, a technical showpiece is an aspect of your game that has two features: (1) it makes the game more complex to implement (and more technically interesting) than say Pac-Man, Defender or DigDug; and (2) it should make your project interesting to someone who hates computer games. High quality AI, interesting datastructures/algorithms for collision detection and point location, or a multiplayer game that uses networking all count as technical showpieces. Multiplayer, real-time, games that implement networking will receive a 10% bonus on the technical showpiece score for the final project. Projects whose technical showpiece revolves around implementing a data structure or algorithm should plan to perform an experimental evaluation of their work. Typically, this will be a short write up with one or more graphs to illustrate the properties of the implementation.

Deliverables:

- Project Pitch (brief in-class presentation)
- Design Document (written document)
- Individual Status Reports (Due each Sunday 7 days after the design document is due)
- Group Status Report (brief in-class presentation)
- Working Game (on display at showcase)
- Git Repository & Documentation (turned in at showcase)

Each deliverable is described in detail in the following sections:

Part 1: Project Pitch *(15 pts, Due November 10)*

As with Project 1, here the pitch is a short, in-class presentation for you to share a game concept with the class and get some immediate feedback before you finalize your proposal. The pitch should be a short (7min) presentation that contains the essence of your proposal:

- A description of the game idea and overall mechanics (what will the player do?)
- A description of the technical showpiece and an argument as to why someone who hates games would be interested in it
- Thoughts on how you will task out the development between group members

You will be graded on how well you communicate the ideas behind your game, however, I will also give you feedback immediately as to whether I think the game is reasonable in terms of complexity – hopefully, this will help you refine your ideas before the design document is due.

Part 2: Design Document *(80 pts, Due November 10)*

[80 points] Design Document:

As with the previous project, the first formal step will begin with a high level design document. In this first week, you should map out the core ideas behind your project. Again, this document should serve to help me understand your project and what to expect when it is complete. I don't want details like a class hierarchy or UML.

Your document should come in close to four or so pages and should contain three main sections:

Game Overview: Here, you should describe at a high level what the game is. Plan to spend around a page/page and a half on this part. You should be sure to cover at least the following points:

1. How will the game be played?
2. What types of interactions are possible?
3. What are the visual entities in the game?
4. What will the player do?
5. What makes this idea interesting, or why do you think this will be fun?

This section should let anyone know what to expect in your final project. To help clarify your intention, it makes sense to draw parallels to existing games (e.g., we're going to build a multi-player/multi-level defender with bosses at each level's end). **Note that the description here *should all be with respect to your low-bar* expectations**. Don't set your sights too high here, there will be time for that later. This section should describe a feature set that you're ready to commit to and that meets the expectations outlined in the first section of this document.

Development Strategy: In this section, you should describe what you see to be potential sticking points in your low-bar goals. *Begin by describing your starting point: do you have existing code you'll be reusing or are you starting from scratch?* Then examine what you see to be the hardest parts of the project or that parts with the most unknowns. Describe how you will approach these aspects of development. Finally, break the development process into subgoals/milestones and describe the role(s) each team player will play to achieve each of these milestones. I don't want to see a day by day description of your plan, rather, think of 2-3 main milestones and what you intend to have complete for each of these. *Be sure that one of your milestones corresponds to the alpha due date (Dec 2) in the course schedule and is highlighted as the "alpha release". I will want to see this milestone functioning and you should describe the feature set that you expect to be complete for this deliverable.* As before, the goal here is not to write the timeline in stone, but to help you map the project out, and begin working in a well paced manner. This part will probably be 1 page or less, but this limit should be considered flexible, the goal here should be completeness, not page count.

Technical Showpiece(s): Each game should also have one significant technical showpiece. In general, a technical showpiece is an aspect of your game that has two features: (1) it makes the game more complex to implement (and more technically interesting) than say Pac-Man, Defender or DigDug; and (2) it should make your project interesting to someone who hates computer games. High quality AI, interesting datastructures/algorithms for collision detection and point location, or a multiplayer game that uses networking all count as technical showpieces. Multiplayer, real-time, games that implement networking will receive a 10% bonus on the technical showpiece score for the final project. Projects whose technical showpiece revolves around implementing a data structure or algorithm should plan to perform an experimental evaluation of their work. You must describe this experiment along with the type of results/graph you plan to capture and report upon. You should spend about a page on describing the technical showpiece you are planning for your low-bar implementation.basically the goal here is to convince me that they are non-trivial.

High Bar: The fourth section of your document should show how your low-bar version could be transformed into your *high-bar* version. Keep this section realistic, and use the knowledge you gained from Project 1 to scope your ideas.

Low Bar Checklist: Finally, *on a separate page*, create a checklist of low-bar items. In essence, you want to highlight the features you've discussed above and boil it down into ~6-12 features that I can look for in your final product. Each item should have a short name followed by a brief description of the item (1-2 sentences typically). *Items that correspond to deliverables for your technical showpiece should be called out in some fashion (e.g., with a footnote, or label).* As with Project 1, I will use this list to evaluate your project and gauge the overall complexity to ensure it will earn an A/A- if well polished.

Grading: The paper will be graded as follows:

65% of the points will be based on how well you address the questions above;
35% of the points will be based on the overall quality of your delivery;
upto 10% extra credit will be based on how creative or interesting the project is

A note on creativity: designing a good game is a creative process, and one that has some inherent risk. That is, not all creative and interesting ideas will actually yield fun and interesting games. If you want to clone an existing game, that's fine, and it's a good way to help ensure that your game will be reasonably fun. However, in doing so, you've essentially outsourced the creative task to someone else.

Turning it in: The paper is nominally due at midnight. **Instead of emailing me the document, you must check it into a git repository and give me read access to that repository** (my public ssh key can be found on Angel under the course documents, or you can add me to your project on the encs gitlab server). When I clone your repository, I want to see the design document and that there is at least one checkin for each group member – it can be trivial (creating the issues.mkd file, for example), but I should be able to verify that everyone in the group has commit access.

Part 3: Individual Status Reports *(7 pts per report)*

One week after the design document it due you must begin to send me individual status reports each Sunday. Thus, the status reports are due:

November 23
November 30
December 7
December 14

The status reports should be a brief synopsis of the week's activity to keep me informed about how your group is working. In particular I want you to address the items (often one sentence per item will be sufficient)

1. What did *you* do since the last status report?
2. How is your team working together? (if you're having team dynamics issues, let me know!)
3. How confident are you about meeting your low bar objectives?

The status reports are critical because they can help me intervene if a group is not functioning well. It is in your best interest to be honest and forthcoming in these reports as we may be able to avoid costly missteps if issues can be identified early.

Part 4: Group Status Report *(15 pts, Due December 3)*

The status report is a short, in-class presentation to help share your progress with the rest of the class. As with the project pitch, the status report should be short (~7min) but it should also convey enough detail that it's clear you're making progress on your game. In particular you should:

- Prepare a screen shot or demo of your game in it's current state
- Describe what remains to be done to meet your low bar goal
- Describe one technical challenge you faced and how you solved it

You will be graded on how well you deliver your report and communicate the issues surrounding your technical challenge.

Part 5: Game Showcase *(310+ pts, Due December 17)*

The final game showcase will be held during our final exam slot. As with Project 1, *attendance at the showcase is mandatory*. Guests are typically invited to the showcase, so it's critical that your game is finished and in good form!

Deliverables for the showcase:

- a git repository demonstrating the CS447 workflow with your final version as the working copy
- a readme indicating: (1) controls for the game; (2) cheat codes; (3) a list of the originally proposed low-bar goals and their current status (complete/partially complete/incomplete); (4) a list of other goals that were completed but not proposed in the original low-bar.
- a statement indicating the licensing terms for your game (e.g., a creative commons license)

Roughly speaking the grading will follow these guidelines:

- 60 points will be for basic functionality. If you've created some sort of functioning game, you will get these points.
- 100 points will be based on playability and the overall level of polish. If the game is fun to play, balanced and has appropriate controls, it will score well in playability. If the game is asthetically pleasing (art and sound), has appropriate transitions and splash screens, and meaningful starting and end states, it is probably polished. A game that is both playable and polished will likely impress a non-computer science student.
- 60 points will be given for a completed and fully functioning technical showpiece.
- 60 points will be based on technical complexity. Roughly 45-50 points will be given for a game mentioned in the overview section of this document
- 30 points will be based on how well your git repository follows the CS447 workflow and whether you've included the readme and license terms as described above
- I'll give upto 10% extra credit for games that go beyond expectations in terms of quality or complexity.