SEATING CHART QUIZ
GAME DEVELOPMENT LEADS TO GAMES...
BUT WHAT IS REQUIRED TO DEVELOP A GAME?
WHAT IS REQUIRED TO DEVELOP A GAME?

- Open world games?
WHAT IS REQUIRED TO DEVELOP A GAME?

- Fighting games?
WHAT IS REQUIRED TO DEVELOP A GAME?

- Cellphone games?
**ENTER THE GAME ENGINE...**

- Unified system for creating games
- Integrates workflow of multi-disciplinary team members
- Provides libraries/APIs for useful systems
- Built to support particular features a game requires (e.g. a visual novel engine will have entirely different considerations than an engine for arena-based shooters)
GAME ENGINES AND WORKFLOW

- Integrates workflow of multi-disciplinary team members
  - Art and animation pipeline
  - Level, systems and narrative design
  - Underlying code and technology to tie everything together
GAME ENGINES AND LIBRARIES

- Provides libraries/APIs for useful systems
  - Level and asset editors
  - Graphics (3D and 2D)
  - Physics
  - Animations
  - AI
  - Networking
  - Sound
  - GUls
  - etc...
Every engine is different

- Learn to work within the constraints of the system rather than fighting against them

Performance matters

- Consider how your code effects frames-per-second, battery life, and download size

Work within the project’s scope

- Weigh long-term technical debt against meeting approaching deadlines

Build for the team, not yourself

- Understand and communicate with other disciplines and programmers
WHAT THIS COURSE IS NOT

- Not a game design class!
- Not an engine-building class (that’s CS354r)
- Not a UE4 tutorial class (although we will use Unreal Engine 4)
WHAT THIS COURSE IS

- A way to see and interact with a large-scale software system (specifically a game engine)
- An introduction to the basics of C++, event-driven programming paradigms, and game engine features
- A chance to hone your ability to critique and design engines and engine features
- An environment to master team-based development and clear communication
COMMUNICATIONS

- We’ll be using Piazza and Discord for questions and answers to specific problems, and Discord for class communication/in-class discussion
  
- Please join both so you are able to receive announcements, keep up on issues, and ask questions
  
- Students should work together before asking for teacher or TA involvement
  
- Grades and assignments will be done via Canvas
BOOKS AND RESOURCES

- Recommended “textbook”: “Game Engine Architecture” Jason Gregory
  - Good exposition of many engine technology and design
  - Not required but useful

- Other useful books:
  - “Game Programming Gems 1-8”
  - “3D Game Engine Design” David Eberly (lots of equations, less exposition, good math background and computer graphics)

- Websites:
  - www.gamasutra.com
  - www.unrealengine.com

- Game developer technical and trade news GDC Vault and Siggraph archives
GRADING

- Projects, reports, and labs (no tests)
  - 5 major projects
  - Groups of 3 assigned by the TA for the first 4
  - Self-forming teams allowed for the final project
- 8 labs with in-class work days
- Potential quizzes to check comprehension and attendance
GRADING

- Groups will be graded as one, but adjustments will be made based on individual performance
  - Each group will be evaluated both on the project submissions and in-between milestones submitted via git
  - We will use pull requests/pushes to assess how much each group member contributed to the project
- For the final project, you will set your own milestones and goals
  - You will be graded based on how well you achieve your goals factoring in degree of difficulty
WORKING IN GROUPS

- Working in groups is an acquired skill and the most important thing you’ll learn in here!
  - For some information on group functioning, read [http://www-honors.ucdavis.edu/vohs/index.html](http://www-honors.ucdavis.edu/vohs/index.html)
- We assign teams – like in industry
- Group evaluation exercises throughout the semester will ensure an even distribution of work (and grades)
  - You must evaluate teammates (even if only to say nice things about them!)
  - Low performance and poor team evaluations can result in failing the class
I am in the process of getting UE4 on the lab machines but there’s no official timeline if/when that will happen.

You MUST include screen capture of your program in action, a short report documenting key features and where they are implemented in the code base, and screenshots to simplify checking over the project.

UE4 projects are extremely heavyweight, so you MUST use version control for submitting your projects.

We’ll use GitLab, so make your repos private.

You’ll branch a “code-freeze” version for each project/milestone and submit repo information via Canvas. **Any modifications to the code-freeze branch after the project deadline will deduct from your late slips.**
We will be exploring game development and common programming paradigms via Unreal Engine 4

- [www.unrealengine.com](http://www.unrealengine.com)

- A commercial game engine with fully readable source code

- Can build from source or use the binary installation

- Note: we are not modifying the engine directly

- Please be sure to use the requested version of UE4 to avoid potential build issues
PROJECT TOOLS

- Since we are understanding the game development pipeline, in addition to UE4, we will work with:
  - Source control (Gitlab)
  - Agile Development (Trello)
  - and touch on Continuous Integration (Gitlab)

- We will cover how to set these up in class, but you will be responsible for going beyond the material when your project requires it.
TOOLS FOR CONTENT CREATION

- Models and art are the biggest expense in real games
- This course doesn’t require outside art assets, but:
  - You can use Blender in the lab or other programs on your own machines
  - Acknowledge any assets you download/purchase
  - Assets should be usable in the Linux environment even if you develop in non-Linux environments
- You may need to write format converters if you have a good tool that produces output that you can’t input. This is a big deal in the real world as well as in class!
SOME UNREAL DEMOS...

- UE4 demo from 2012:
  - https://www.youtube.com/watch?v=wYa8tHPhbDo

- UE5 demo from 2020:
  - https://www.youtube.com/watch?v=Oa2drgVThbs