

Project Proposal Components [15 pts]

- Project Description [2.5 pts]: The name of the term project and a short description of what it will be.

Name: Rogue 112 (a Galaxy Shooter Game)

Description: My term project will be a 2D arcade-style space shooter game with a galaxy (similar to Star Wars) theme. It will feature multiple different modes, and the player will pilot a ship to shoot down multiple enemy ships while dodging enemy bullets.

- Similar projects [2.5 pts]: A 1-2 paragraph analysis of similar projects you've seen online, and how your project will be similar or different to those.

[Jason's Term Project] <https://www.youtube.com/watch?v=koTBrCNEhH8>

[Galaxy Attack: Alien Shooter, Galago, Phoenix 2] app store

[Sarathi's Term Project] <https://www.youtube.com/watch?v=AH9JweZajJs>

My project will be similar to games like Galago, Galaxy Attack: Alien Shooter, and Phoenix 2 on the app store. These games range from simple to more complex bullet sequences, and players can interact with easy to hard skill levels. I also analyzed a few term projects on the term project gallery that also feature 2D shooter scenarios. My project will be similar because it also features the player ship having the objective of passing and killing all enemy ships and bullets without touching any of them. I will also have a function similar to a "bomb" that is a limited-use skill to clear the screen.

However, my project will also involve different modes with different bullet patterns rather than increased difficulty modes. My project will also only have one ship that the player uses (the player cannot select through different ships). My enemy ships will likely have less complicated movement than online apps.

- Structural Plan [2.5 pts]: A structural plan for how the finalized project will be organized in different functions, files and/or classes.
 - Initialize game:
 - Set up game window, how large the screen will be, what buttons the player can press.
 - Player controls:
 - Mouse Press: for the player to select different modes and start the game.
 - Mouse Move: once the game level is started, the player ship location is assigned to follow the player's mouse.
 - Key Press: for the player to start/exit the game and to use the special skill.
 - Collisions:
 - Determine if the player's ship collides with an enemy bullet, or an enemy ship.

- Determine if the player's bullets collide with enemy ship.
 - Special Skill:
 - If special skill is used, determine what happens and how we will clear the screen but continue with the game.
 - Scoring:
 - Keep track of player's score
 - Assign scores for doing different things such as killing an enemy ship, killing different level enemy ships.
 - Game Over:
 - Determine when will the game be over
 - Make a game over screen
 - Option for the player to restart
 - Graphics:
 - Find graphics for the ships and bullets and different elements
 - Find backgrounds to enhance the game.
 - TP1 Update:
 - Make an image file for graphics
 - Make the enemy/player ships able to rotate
- Algorithmic Plan [2.5 pts]: A plan for how you will approach the trickiest part of the project. Be sure to clearly highlight which part(s) of your project are algorithmically most difficult, and include some details of how you expect to implement these features.

I think the most tricky part of this project will be classifying the different elements and figuring out how to differentiate how the player ship or enemy ships will react when it touches different classed bullets. I also think it will be difficult to coordinate how "bullets" will be fired. I plan to use OOP to give different attributes to different objects.

TP1 Update: I think the most tricky part of this project will be to code a simple AI for the enemy ships to determine when to fire most accurately at the player ship. I also think getting familiar with PIL and importing graphics into the game will also be a challenge.

- Timeline Plan [2.5 pts]: A timeline for when you intend to complete the major features of the project.
 - Week 1 (11/24): Basic movements of objects (player ship, enemy ships, bullet patterns), Collision reactions
 - Week 2 (12/1): skill usage, scoring, graphics, background, try to reach MVP
 - Week 3 (12/6): Figure out difficult parts that haven't been completed yet (simple AI, obstacles) and see if additional features beyond MVP can be added.
- Version Control Plan [1.5 pts]: A short description and image demonstrating how you are using version control to back up your code.





I am going to use Google Drive to back up my code. At each week, I will upload my code file into its respective Drive folder.

My Drive > 15-112 TP ▾

Type ▾ People ▾ Modified ▾

Name ▾	Owner	Last modified ▾	File size	
Week 3 (Grahpics, Backgrounds)	me	4:29 PM me	—	👤 ⬇️ ✎️ ☆ ⋮
Week 2 (Collisions, Skill Use)	me	4:29 PM me	—	⋮
Week 1 (Basic Movements)	me	4:29 PM me	—	⋮

TP1 Update: I figured out how to use GitHub to backup my code, and I will routinely backup my code to github with a comment on what I changed.

 ellylai started making 3 modes	402cdea 1 hour ago	🕒 3 commits
 TP	started writing classes	last week
 TP file 3 (collisions)	started making 3 modes	1 hour ago
 main.py	Create main.py	last week

- Module List [1 pts]: A list of all external modules/hardware/technologies you are planning to use in your project. Note that any such modules must be approved by a tech demo. If you are not planning to use any additional modules, that's okay, just say so!

I am not planning to use any additional modules.

(MVP Requirements):

- ☐ To reach MVP:
 - ☐ Code Organization
 - ☐ Code is organized at 112 standards
 - ☒ ~~Meaningful OOP: Player, Enemy Classes~~
 - ☐ User Interface (UI):
 - ☐ Decent game interface
 - ☐ User Experience (UX):
 - ☐ Player and enemy movement
 - ☐ Player and enemy can die
 - ☐ Player and enemy ships can rotate
 - ☐ Win/lose condition
 - ☐ Algorithmic Complexity:
 - ☐ Mathematically modeled various bullet types with different patterns that always target/home on the player location (minimum of 3 needed):
 - ☐ Sinusoidal

- ☒ Straight line
- ☒ Zig-zag
- ☒ Homing (required)
- ☐ Acceleration
- ☒ Rectangular collisions
- ☐ Obstacles (rectangular shapes randomly generated on the screen that can obstruct sightlines/movement)
- ☐ Simple Enemy AI to best attack the player

Instructions for the game:

To move your ship: the ship follows your mouse movement, so use your mouse to control the ship and dodge enemy bullets.

Objective: in each level, the objective is to kill all enemies in the level without dying yourself.

Your ship can take 10 hits. In each level, the enemies will have different health.

To use special skill: every five kills, you will reload your skill use quota. Press 'space' to use the skill and clear the screen of enemies and bullets.