# Tank Apocalypse

https://github.com/MarzP/GameTrio.git

# Part 1 - Find a Pair and put names below:

Name 1 - Mariano Pimentel
Name 2 - Maria Zuk
Name 3 - Jonathan Toennesen

# Part 2 - Idea

Using your knowledge of Unity so far, come up with a game idea that can be completed by you and your partners before next class.
Describe the idea: 2D game: Shoot Asteroids. Don’t Die. If you die, you suck.
What is the main objective of the game? - Shoot obstacles (asteroids/meteors) and don't get killed by them.
Does your game have enemies? - asteroids/meteors
Who is the main character? - Tank
What will the user interactions be like? - Clicking to shoot

# Part 3 - Implementation

Implement your game idea!
How are you organizing the development? (How are you organizing your Unity code, is any research or team management going into this?)

- Importing, reorganizing and recycling the SpaceShooter and Tanks projects into one project.

# Part 4 - Reflection

Answer the following questions
Was it possible to complete your project idea within the 2 days? How long did it take?

- Yes, because we kept the goal very short and aimed to make the minimum viable game. It took the time during class to get github working, and several hours outside of class to finish.

Which part took the most amount of time?

- Debugging the asteroid respawn and destruction.
- We also had issues with merging between cross-platform (mac/windows) even though it didn't matter in some past projects github just refused to merge anything.

How did you break down the tasks? Explain what each person did on this project. (If it was primarily pair-programming that’s ok!)

- Jonathan: Exported/Imported the prefabs and scripts we wanted to use.
- Maria: Got Github working for all of us
- Marz: Changed the scripts and prefabs to work in a different setting.

How would you expand on this project if you had a week to work on it?

- I would import different models for the tank and asteroids just to make it visually different from the tutorials we’ve worked on before.

What part was the most difficult to implement and why?

- Github, because there were a lot of unexpected errors, that we managed to work around, rather than solve.