Bin Lin Zhou, Eileen Xu, Gary Jiang
Period 2

CLAM presents:
Tank Terror

Description

Our game is based off of the Flash game AZ Tanks. In this game, the objective is to control a tank that can shoot bullets in order to eliminate all opposing tanks. The map that you and your enemies are spawned in is randomly generated. It is a maze with walls that you cannot pass through. The bullets are able to bounce off walls. Thus, you want to dodge existing bullets while also approaching and shooting bullets at your enemies. You cannot be damaged by your own attacks, nor can NPCs be damaged by their own attacks. It will be a 1-2 player game in which you can play against someone else and even add up to 3 NPC (non-playable characters) enemy tanks. In addition, there will be power-ups that spawn on the map in regular intervals. These power-ups can modify the base attack of your tank, giving you an advantage. The goal of this game depends on whether you are playing on the single-player or two-player modes. For the single-player mode, the goal is to survive as many rounds as possible, with each round defined by a new maze and respawned enemies. You lose when your tank is eliminated and win each round when all enemies are eliminated. For the two-player mode, the goal is to beat your opponent. Whoever reaches 5 wins first is the victor. If a round is completed with no players alive (only NPCs remain), there is a tie.

Functionalities:

- Score tracker
    - We will utilize a global instance variable that will increase by 1 for every round won by the corresponding player. There will be two of these variables, one for each player in the two-player mode, in which only one will be used in the single-player mode.
    - The score(s) will be displayed on the screen after each round.
- Win Screen
    - For the single-player mode, the score will be displayed on an end screen after the player's tank is destroyed and the game ends.
    - For the two-player mode, the victor will be displayed depending on who wins 5 rounds first.
    - Both of these will be displayed in processing using the instance variables from the score tracker.
- Maze generation
    - Using a set maze size, we can use recursive backtracking and code similar to that of the carveMaze lab to randomly generate mazes.

- Tank spawning
    - Player and NPC tanks will spawn in random valid positions.
    - Tanks will be spawned a set minimum distance away from other tanks to avoid rounds that end too quickly.
- Player movement
    - We will have instance variables to track the location of player tanks at all times, updating them every time the tanks move. For multiplayer, there will be two sets of these variables, one for each player.
    - By changing these variables, we can easily create the forward and backward movements. If a tank is rotated, we can utilize trigonometry to change these variables such that they work.
    - Using the rotate() method in processing, we will rotate the tanks through the A/← and D/→ inputs on a set angle interval.
    - Tanks cannot occupy the position of a maze wall or another tank.
- Player shooting
    - By pressing the corresponding shoot button, a bullet will be created directly in front of the tank and travel forward continuously in correspondence with the initial direction of the shooting tank until a boundary or enemy tank is reached.
    - If a boundary is hit, the bullet will bounce off at an angle equal to its incidence angle relative to the normal of the boundary (see: law of reflection) and continue moving until the timed duration of the bullet is reached.
        - The time duration is stored in an instance variable with a method that updates its value.
    - If an enemy tank is hit, both the tank and bullet are destroyed.
- NPC movement
    - Using the position of the player tanks, each NPC tank will move toward the nearest player tank. Similar instance variables compared to player tanks will be used to store the locations of the NPCs.
    - NPC movement will be similar to player movement.
    - Tanks cannot occupy the position of a maze wall or another tank.
- NPC shooting
    - NPCs will have a set range in which they can detect a player tank.
    - If a player tank is within that range, an attack will be fired in the direction of the player.
    - Bullet will work the same as in player shooting.
- Power-ups
    - In an instance variable that stores the timer for each power-up's spawn time, a random powerup will be spawned every 7 seconds in a random valid position, 10 seconds after the start of each round.

- If a tank occupies the same space as a powerup, that tank will gain that power-up. The player's base attack is then modified and the power-up disappears from the map.
    - 3 power-ups we can implement are:
        - Laser (one-time use): the attack travels the path that it would take within the set lifespan of a bullet instantly
        - Phasing bullet (one-time use): the attack can travel through walls but not through outer boundaries of the maze.
        - Machine gun (one-time use): reduces the attack downtime to 0.5 seconds instead of 5 for a set amount of time
    - Power-ups, unless specified, last indefinitely until used.
- Power-ups will be stored in instance variables for each player tank.
- NPC tanks CANNOT pick up power-ups.

How does it work?

Objective: The goal of this game depends on whether you are playing on the single-player or two-player modes. In both cases, you want to shoot bullets towards your enemies while avoiding your enemies' bullets. For the single-player mode, the goal is to survive as many rounds as possible, with each round defined by a new maze and respawned enemies. You lose when your tank is eliminated and win each round when all enemies are eliminated. For the two-player mode, the goal is to beat your opponent. Whoever reaches 5 wins first is the victor. If a round is completed with no players alive (only NPCS remain), there is a tie and no player is awarded points.

Running the game:

1) The game will prompt the player(s) to select either the single-player or two-player mode.
2) Next, the game will prompt the player(s) to select the number of desired NPCs on the map.
3) A randomly generated map of standard size will be created.
4) The players (and NPCs if selected) will be spawned in random valid positions within the maze.
5) The game will start after a 3 second countdown.
6) Powerups will start spawning 10 seconds after the start of the round and continue spawning on 7 second intervals
7) The round ends when only 1 player is alive for two-player mode or when 0 players are alive for both modes and the score is updated accordingly
8) After a round is over, a new random map is generated and the game proceeds to a new round
9) In single-player, the game ends when the player tank dies
10) In two-player, the game ends when a player reaches 5 points (5 won rounds)

11) Victor messages are displayed

Controls:
Mouse to select prompts from the game

1) Single-player mode
    - W Moves the tank forward in the direction it is facing
    - A Rotates the tank counterclockwise
    - S Moves the tank backward from the direction it is facing
    - D Rotates the tank clockwise
    - Q Fires attack (typically a 5 second cooldown for base attack)
2) Two-player mode
    - Player 1: WASDQ keys
    - Player 2:
        - ↑ Moves the tank forward in the direction it is facing
        - ← Rotates the tank counterclockwise
        - ↓ Moves the tank backward from the direction it is facing
        - → Rotates the tank clockwise
        - / Fires attack (typically a 5 second cooldown for base attack)

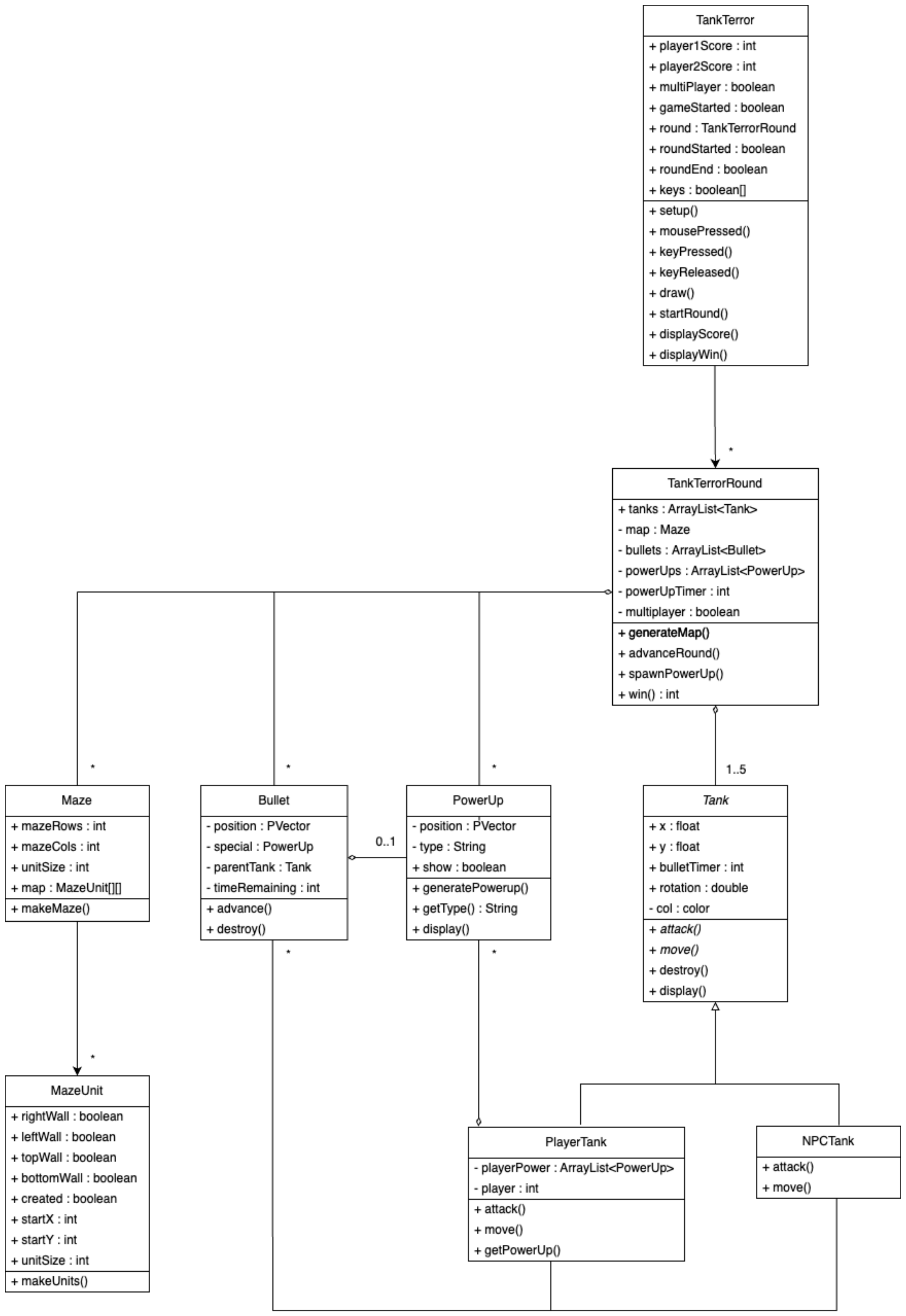
Functionalities/Issues
Current Functionalities
- Start screen (player mode selection)
- Tank spawning
- Maze generation
    - Using only random wall generation instead of backtracking
- Basic player movement (forward, backward, rotation)

Future Functionalities (by next meeting)
- Player shooting
- Player movement
    - Make it so that the tank cannot move through the walls of the maze
- Start screen (selection of NPCs)
- Powerups
- Maze generation
    - Using recursion to ensure that every unit in the maze is accessible

Issues
- Tank movement

- Problem: More than one key press would not function. A player could not simultaneously rotate and move forward. This also meant that only one player tank could rotate/move at a time.
    - Solution: We stored the keys into an array of booleans and then accessed elements in the array using key codes in the PlayerTank class. We also added keyPressed() and keyReleased() methods to the TankTerror class to change the values in the array to either true or false.
- Problem: Rotating the first player also rotates the second player.
    - Solution: We used pushMatrix and popMatrix to isolate transformations to a single tank

UML Diagram


TankTerror
+ player1Score : int
+ player2Score : int
+ multiPlayer : boolean
+ gameStarted : boolean
+ round : TankTerrorRound
+ roundStarted : boolean
+ roundEnd : boolean
+ keys : boolean[]
+ setup()
+ mousePressed()
+ keyPressed()
+ keyReleased()
+ draw()
+ startRound()
+ displayScore()
+ displayWin()
*
TankTerrorRound
+ tanks : ArrayList<Tank>
- map : Maze
- bullets : ArrayList<Bullet>
- powerUps : ArrayList<PowerUp>
- powerUpTimer : int
- multiplayer : boolean
+ generateMap()
+ advanceRound()
+ spawnPowerUp()
+ win() : int
1..5
*
*
*
Maze
+ mazeRows : int
+ mazeCols : int
+ unitSize : int
+ map : MazeUnit[][]
+ makeMaze()
Bullet
- position : PVector
- special : PowerUp
- parentTank : Tank
- timeRemaining : int
+ advance()
+ destroy()
0..1
PowerUp
- position : PVector
- type : String
+ show : boolean
+ generatePowerup()
+ getType() : String
+ display()
Tank
+ x : float
+ y : float
+ bulletTimer : int
+ rotation : double
- col : color
+ attack()
+ move()
+ destroy()
+ display()
*
*
*
MazeUnit
+ rightWall : boolean
+ leftWall : boolean
+ topWall : boolean
+ bottomWall : boolean
+ created : boolean
+ startX : int
+ startY : int
+ unitSize : int
+ makeUnits()
PlayerTank
- playerPower : ArrayList<PowerUp>
- player : int
+ attack()
+ move()
+ getPowerUp()
NPCTank
+ attack()
+ move()