Name: Ethan Parks

Title: Tower Defense

Project Summary: Aliens enter the screen on one side and follow a set path through the map and off the screen. The player must use currency generated from defeating aliens to buy and place towers on the map which work together to defeat the aliens before they exit the map. The player may choose from a range of upgrades to the turrets to make them stronger or change their effects. The player receives points towards a high score upon defeating an enemy.

Project Requirements:

ID	Requirement
UC-01	As a user, I must be able to select a map to play.
UC-02	As a user, I must be able to place towers on the map.
UC-03	As a user, I must be able to upgrade my towers.
UC-04	As a user, I must be able to pause the game.
UC-05	As a user, I must be able to undo my tower placement.
UC-06	As a user, I must be able to undo tower upgrades.
UC-07	As a user, I must be able to save my game.
UC-08	As a user, I must be able to load my game.
UC-09	As a user, I must be able to see my score.
UC-10	As a user, I must be able to see my local high scores.
UC-11	As a user, I must be able to see my current amount of currency.
UC-12	As an admin, I must be able to add currency to the player.
UC-13	As a user, I must be able to start a new game.
UC-14	As an admin, I must be able to remove currency from the player.
UC-15	As an admin, I must be able to spawn new enemies.

UI Mockups:

ID	Requirement	How the user will interact with UI to meet requirement
UC-01	As a user, I must be able to select a map to play.	The game will take a command line argument to select the map to play. Once the program is started, that map is automatically loaded.
UC-02	As a user, I must be able to place towers on the map.	The user will be able to place towers on the map by clicking the buy tower button (as seen in the lower right corner of figure 1) and then clicking on the map to place the tower.
UC-03	As a user, I must be able to upgrade my towers.	The user will be able to click on the tower she wants to upgrade and a window will appear with a drop down box for her to select an upgrade for the tower, as seen in figure 2.
UC-04	As a user, I must be able to pause the game.	The user will be able to click the play/pause button as seen in the middle of the lower panel of figure 1.
UC-05	As a user, I must be able to undo my tower placement.	The user will be able to type undo into the console to undo tower placement.
UC-06	As a user, I must be able to undo tower upgrades.	The user will be able to type undo into the console to undo tower upgrades.
UC-07	As a user, I must be able to save my game.	The user will be able to type save into the console to save the game.
UC-08	As a user, I must be able to load my game.	The user will be able to load a previous game by passing the save file as a command line argument to the program. The game will immediately start from the load file.
UC-09	As a user, I must be able to see my score.	The user score is displayed in the upper right hand corner as seen in figure 1.
UC-10	As a user, I must be able to see my local high scores.	The user will be able to type highscore into the console to view their high score for the current map.

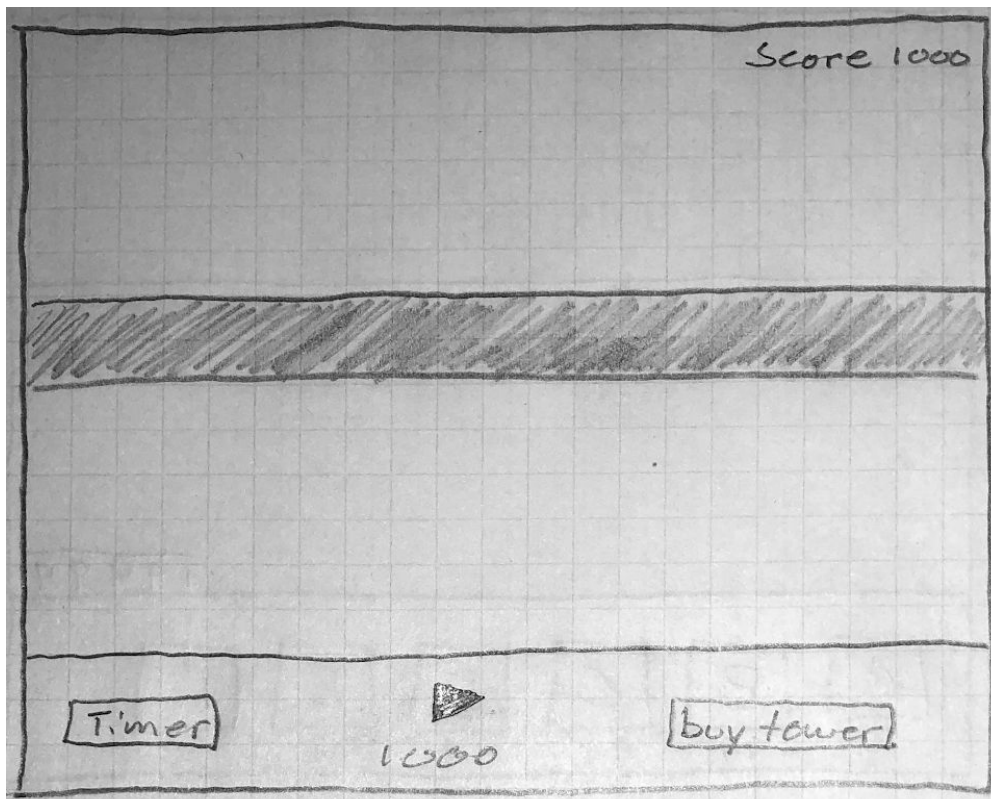| UC-11	As a user, I must be able to see my current amount of currency.	The amount of currency the user has is visible below the pause/play button.
UC-12	As a user, I must be able to start a new game.	The user will be able to type new into the console to start a new game.
UC-13	As an admin, I must be able to add currency to the player.	The admin will be able to type add x to add x amount of currency to the player.
UC-14	As an admin, I must be able to remove currency from the player.	The admin will be able to type remove x to remove x amount of currency from the player.
UC-15	As an admin, I must be able to spawn new enemies.	The admin will be able to type spawn in the console to force new enemies to spawn.

Figure 1:

Figure 2:

Enemy

- -health:float
- -isFlying:boolean
- -location:Point
- -speed:float

- +getHealth():float
- +setHealth(_health:float):void
- +getIsFlying():boolean
- +setIsFlying(_isFlying:boolean):void
- +getLocation():Point
- +setLocation(_location:Point):void
- +getSpeed():float
- +setSpeed(_speed:float):void

Model

- -currency:float
- -enemies:ArrayList<Enemy>
- -path:ArrayList<Point>
- -score:float
- -towers:ArrayList<Tower>

- +getCurrency():float
- +setCurrency(_currency:float):void
- +getEnemies():ArrayList<Enemy>
- +setEnemies(_enemies:ArrayList<Enemy>):void
- +getPath():ArrayList<Point>
- +setPath(_path:ArrayList<Point>):void
- +getScore():float
- +setScore(_score:float):void
- +getTowers():ArrayList<Tower>
- +setTowers(_towers:ArrayList<Tower>):void
- +update(deltaT:float):void

BaseProjectile

- -direction:Vector3d
- -location:Point
- -target:Enemy
- -damage:float

- +explode():void
- +clone():BaseProjectile
- +getDamage():float
- +setDamage(_damage:float):void
- +getDirection():Vector3d
- +setDirection(_direction:Vector3d):void
- +getLocation():Point
- +setLocation(_location:Point):void
- +getTarget():Enemy
- +setTarget(_target:Enemy):void

Tower

- -location:Point
- -projectile:BaseProjectile
- -projectiles:ArrayList<BaseProjectile>

- +fire():BaseProjectile
- +getLocation():Point
- +setLocation(_location:Point):void
- +getProjectile():BaseProjectile
- +setProjectile():void
- +getProjectiles():ArrayList<BaseProjectile>

Controller

- -model:Model
- -view:View
- -isPlaying:boolean

- +actionPerfromed(event:ActionEvent):void
- +update(deltaT:float):void

Projectile

- +explode():void

Decorator

- -base:BaseProjectile

- +explode()

SlowProjectile

- +explode()

AreaOfEffectProjectile

- +explode()

View

- -addTowerBtn:JButton
- -canvas:JCanvas
- -currencyLabel:JLabel
- -frame:JFrame
- -pauseBtn:JButton
- -scoreLabel:JLabel
- -timeLabel:JLabel

- +getAddTowerBtn():JButton
- +setCurrencyText(currency:string):void
- +draw(towers:ArrayList<Tower>, enemies:ArrayList<Enemy>):void
- +getPauseBtn():JButton
- +setScoreText(score:string):void
- +setTimeText(time:string):void