# CMPT 276 Project – Group 17
# Phase 1: Requirement Engineering and Design
# February 16, 2024

Created By:
Jonas Lam
Yanjun Qian
Brendan Shen
Sheila Nicholson

# Game Description:

You are a psychology student attending SFU, slowly progressing towards obtaining your degree. It is the midst of finals season and you've unintentionally slipped into a deep slumber. Utterly exhausted from the constant studying, assignment deadlines and cramming of missed lectures, you slowly blink your eyes open and are shocked to find yourself lost in the deep, dark maze known as Robert C. Brown Hall. It haunts you, even in your dreams… Now comes the challenge. You must evade rabid raccoons, zombie professors and looming failed exams to make your way out. With the help of copious amounts of caffeine, A+ papers and a cozy bed or two along the way you just might be able to pass your exams, earn your degree and be released from the labyrinth that is Robert C. Brown Hall. Good luck!

## Overview:

Main Character: A psychology student having a nightmarish dream in which they are lost in Robert C. Brown hall. The objective of this character is to get out of the building alive.

Enemies:
Note: Interaction between the main character and enemy ends the game.
**Level 1- Racoon:** A rabid racoon chasing after you. The creature's only ambition is to bite and infect the student.
**Level 2 - Zombie professor:** They want to talk to you about your latest assignment. Did you use ChatGPT?
**Level 3 - Failed exam:** Your worst nightmare has come true. You failed that exam you spent endless hours studying for. How will your GPA ever recover from this?

Rewards:
**Coffee:** Player movement speed is increased for 5 seconds.
**Bed:** You need those extra zzz's with all the studying you've been doing. Player gains 5 points.
**A+ Paper (\*bonus reward\*):** You were told that course was a GPA booster, looks like it's paying off. Player gains 10 points. Reward spawns randomly and disappears after a set amount of time.

Punishments:
**Vortex:** A whirling black object. Composition unknown. When the player comes into contact with the tile they are moved to another area of the map.
**Pile of books:** You trip over them. Whoops. Player loses 5 points.
**Mysterious smoke:** The tile containing smoke is hidden until the player moves to it. Slows player movement speed for 5 seconds.

Map Appearance:
- Faded orange floors
- Concrete walls
- Complicated corridors
- Flickering lights
- Dungeon ambience

# Use Cases:

**Use Case: Initialize the game**
Primary actor: Player
Goal in context: allow player to access and customize game options from the main interface
Preconditions:
    -The game file is executed successfully.
    -The user presented with the the main interface
Trigger:
    -Player executes the game.
Scenario:
1. Player executes the game.
2. Game opens in a separate window
3. Main menu interface is displayed, featuring buttons for Play, Options, and Quit. After clicking Play, there are the three difficulties Easy, Medium, and Hard, which are all unlocked.
4. Player chooses the desired difficulty level.
    -"Easy" level - Racoon:
    1. The racoon has the slowest speed of all the enemies since this is the easy level.
    2. Vortex teleportation minimum spawn distance away from other punishments as outlined above = 3 tiles (subject to change)
    3. Less punishments spawn (less books, less tiles, and less vortexes)
    4. Less normal, more bonus and additional rewards spawn (less beds mean that the user has to collect less of the necessary rewards to escape the level, and more A+ papers and coffee provides more opportunities to gain more points and to get speed boosts to help reposition should the player need it)
    -"Medium" level -Zombie Professor:
    1. There are more books spawn around to trip the student over
    2. The amount of vortexes and smoke tiles should be increased compared to the        previous easy level though
    3. Fewer A+ papers and coffees could spawn, along with more punishments and beds
    4. Vortex teleportation minimum spawn distance away from other punishments as outlined above should be reduced to 2 tiles
    5. Zombie Professor move faster than the Raconn
    -"Hard" level - Failed Exam:
    1. The Failed Exam should move the fastest
    2. More punishments (vortexes, smoke tiles and pile of books) spawn as a result of the harder level
    3. Even fewer or no A+ papers spawn. If the failed exam moves faster than the player, then we can't shrink the amount of coffees, otherwise you'd just lose by default. (could be balance)
    4. Vortex teleportation minimum spawn distance away from other punishments as outlined above should be reduced to 1 tile (subject to change)
5. Player may elect to navigate into the Options menu to configure additional settings of the game.

6.      Player enters the game with a specific difficulty level or returns to the main menu.
Exceptions:
     -Transition from the main page to the menu page is unsuccessful due to technical issues or errors.
Priority: Medium
When available: Begin of the game execution
Channel to actor: GUI
Secondary actors: N/A
Channels to secondary actors: N/A
Open issues: N/A
Notes/Comments:  N/A

**Use Case: Punishments**
Primary actor: Player
Goal in context: allow users to experience penalties within the game for various events, providing an avenue for the user to lose the game.
Preconditions:
        -the game is actively being played by the player
        - the player is on a tile where punishments can occur
        -the game timer is running normally
Trigger:
        -player is on the way to get specific reward or escape from enemy, or is caught by an enemy

Scenario:
   1.  Encounter Vortex:
        • Player moves onto a tile containing a vortex
        • Upon contact with the vortex, the player is abruptly moved to another area of the map , altering their current position and potentially disrupting their planned path or strategy.
        • The vortex cannot teleport the player within a given minimum distance of a punishment (vortex, books, or smoke tiles) or the enemy specific to the level chosen. This restriction may be tightened, loosened, or eliminated based on the difficulty chosen.
        • The vortex disappears after it is encountered.
        • A visual/sound effect is played to indicate the teleportation.


   2.     Tripping over Pile of Books:
        • Player moves onto a tile where a pile of books is present.
        • The player accidentally trips over the books, leading to a loss of 5 points (can be balanced)
        • Points deduction is displayed to the player by a visual cue and a sound effect.
        • The pile of books disappears after it is tripped over.

   3.     Encountering Mysterious Smoke:

- Player moves onto a tile containing mysterious smoke.
- The tile with the smoke remains hidden until the player moves to it, adding an element of surprise and uncertainty.
- Upon encountering the smoke, the player's movement speed is significantly reduced for 5 seconds (can be balanced), making navigation more challenging until the effect wears off.
- A visual or audio effect will indicate to the player that they have stepped on a smoke tile and that their speed has been slowed.
- The smoke tile remains hidden but does not disappear throughout the level being played. That is to say, if the player forgets where it is, they could step on it again and be inflicted with the debuff multiple times in a level.

4.      Get caught by the enemy:
- Player's character is caught by an enemy specific to the difficulty chosen while attempting to navigate the game's environment.
- Player dies and the game gets terminated.

Exceptions:
     -The punishment may not work as expected if the player moves onto a tile occupied by an   enemy immediately after being moved by the vortex.
Priority: High
When available:Throughout gameplay when the player encounters specific triggering events.
Channel to actor: Visual and auditory cues within the game interface, including animations, sound effects, and score notifications.
Secondary actors: Enemies
Channels to secondary actors: N/A
Open issues:
     Need to have enough testing to refine the punishment mechanics expected work.
     Need to playtest enough to get balancing issues under control.

**Use Case: Rewards**
Primary actor: Player
Goal in context: allow user to receive rewards within the game for various events
Preconditions:
    -the game is actively being played by the player
    - the player is on a tile where rewards can occur
    -the game timer is running normally
Trigger:
    - players want a higher score or need to have enough points to survive in the game.
Scenario:
   1. Encounter a cup of coffee:
- Player move onto a tile that contain a cup of coffee
- Upon interaction, player's movement speed is increased for 5 sec (can be balanced)
- Visual of the speed boost effect present (can add sound)

- The coffee disappears after the player drinks it.
2. Hopping on a bed (regular reward):
    - Player move onto a tile that contain a bed
    - Interacting with the bed add 5 points to the player (can be balanced)
    - The bed disappears after the player sleeps in it.
    - All beds must be slept in (collected/stepped over) in order to advance through to the next level.
3. Get a A+ paper (bonus reward):
    - Player move onto a tile that contain a A+ paper
    - Interacting with the bed add 10 points to the player (can be balanced)
    - A+ paper periodically disappears and respawns in a random location.

Exceptions: N/A
Priority: High
When available: Throughout gameplay when the player encounters specific triggering events that concern with reward items
Channel to actor: Visual and auditory cues within the game interface, including animations, sound effects, and score notifications.
Secondary actors: N/A
Channels to secondary actors: N/A
Open issues: N/A


**Use Case: Movement on the map**
Primary actor: Player
Goal in context: Allow the player to navigate and move the character within the game's map environment.
Preconditions: Game clock is running normally.
    The map/game has loaded successfully without any problems.
    A valid input key was pressed.
Trigger: Player presses a valid input key (I would assume we're using WASD)
Scenario: The following assumes the duration of the event is one (1) game tick.


1. W: Player's model in the game moves one (1) unit upwards/north.
2. A: Player's model in the game moves one (1) unit left/west.
3. S: Player's model in the game moves one (1) unit downwards/south.
4. D: Player's model in the game moves one (1) unit right/east.

Exceptions: If the player presses something invalid (other than WASD) nothing will happen. No movement, no warning screen, no nothing.
Priority: Medium
When available: After finished the location setting of reward and punishment
Channel to actor: Visual and auditory cues within the game environment.
Secondary actors: N/A
Channels to secondary actors: N/A
Open issues: N/A

**Use case: Player wins the game**
Primary actor: Player
Goal in context: To successfully reach the end of the particular level the player is completing.
Preconditions: The player currently has a positive or zero score. Their character has evaded the enemy and has maneuvered to the tile next to the door tile on the current level.
Trigger: The main character enters the tile containing the door of the current level of the map.
Scenario:
1. Player enters tile containing the door
2. The game ends
3. A screen appears informing the player that they have won the game. The player's current score and the time values are displayed.

Exceptions: N/A
Priority: Essential, must be implemented
When available: All levels of the game
Channel to actor: GUI. Visual cues within the game interface, including animations, sound effects, and score display.
Secondary actors: N/A
Channels to secondary actors: N/A
Open issues: N/A

**Use case: Player loses the game**
Primary actor: Player
Goal in context: Player is unsuccessful in finishing the particular level and loses the game
Preconditions: Two scenarios can cause this use case to occur, they will be outlined separately:
>    Precondition A: Player has a positive or zero score and their character is on a tile adjacent to a punishment item
>    Precondition B: The player's character is on a tile directly adjacent to a tile occupied by an enemy

Trigger:
>    Trigger A: The player's score becomes negative
>    Trigger B: An enemy moves to the tile that the player's character currently occupies

Scenario:
>    Scenario A:
1. Player moves their character to a tile
2. Tile contains a punishment item
3. Score is calculated based on the penalty of the item, the resulting score is negative
4. The game ends
5. A screen appears informing the player that they have lost the game. The player's current score and the time values are displayed.
>    Scenario B:
1. An enemy moves to the tile that is currently occupied by the main character
2. The game ends
3. A screen appears informing the player that they have lost the game. The player's current score and the time values are displayed.

Exceptions: N/A
Priority: Essential, must be implemented.

When available: All levels of the game
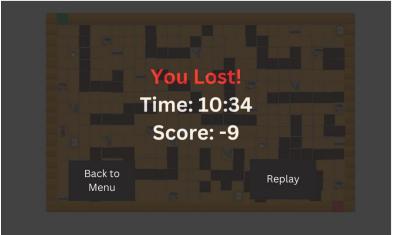Channel to actor: GUI. Visual cues within the game interface, including animations, sound effects, and score display.
Secondary actors: Enemy
Channels to secondary actors: N/A
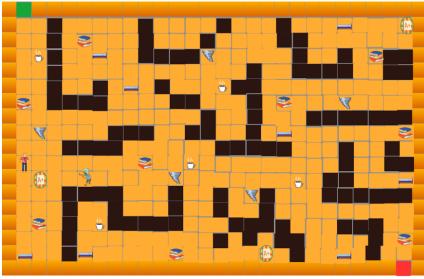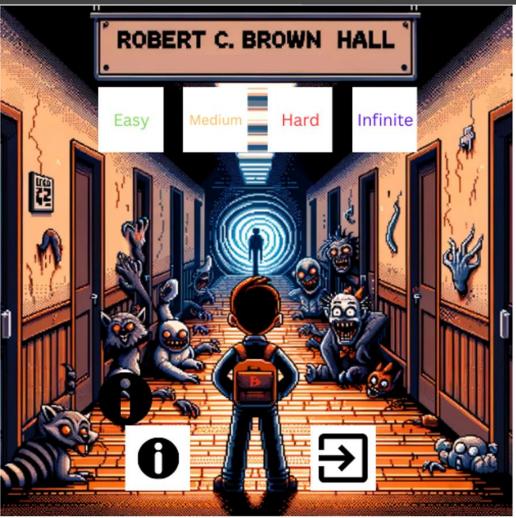Open issues: N/A

**UI Mock-up:**

**You Won!**
Time: 00:43
Score: 120

Back to Menu    Next Level    Replay

**You Died :(**
Time: 04:21
Score: 2

Back to Menu    Replay

# ROBERT C. BROWN HALL

Easy    Medium    Hard    Infinite

# UML Diagram:

**Coffee**
- rewardType : string
- scoreBonus : int
- modifierSeconds: int

+Coffee()
+ scoreModifier(): int
+ modifySpeed: void

**Bed**
- rewardType : string
- scoreBonus : int

+ Bed()
+ scoreModifier() : int

**aPlusPaper**
- rewardType : string
- scoreBonus : int

+aPlusPaper()
+ scoreModifier() : int

1..3

****
**rewardItem**
- rewardType : string
- scoreBonus : int

+ scoreModifier() : int

**Position**
- X: int
- Y: int

+ Position(int, int)
+ setPosition(int, int): void
+ getXPosition(): int
+ getYPosition(): int

**<interface>**
**Item**
+ scoreModifier() : int

**<interface>**
**Character**
+ moveCharacter(): Void
+ displayCharacter(): Void

**Raccoon**
- characterSprite: Image
- damagePoints: int
- movementSpeed: int

+ Raccoon()
+ scoreModifier() : int

1..2

****
**punishmentItem**
- damagePoints: int
- itemSprite: Image

+ scoreModifier() : int

**Hero**
- playerInventory: Item[]
- playerScore: Score
- characterSprite: Image
- movementSpeed: int

+ Hero()
+ displayCharacter(): Void
+ moveCharacter(): Void
- deathAnimation(): Void

****
**Enemy**
- characterSprite: Image
- damagePoints: int
- movementSpeed: int

+ moveCharacter(): Void
+ damagePlayer(): Void
+ displayCharacter(): Void
+ scoreModifier() : int

**zombieProfessor**
- characterSprite: Image
- damagePoints: int
- movementSpeed: int

+ zombieProfessor()
+ scoreModifier() : int

**failedExam**
- characterSprite: Image
- damagePoints: int
- movementSpeed: int

+ failedExam()
+ scoreModifier() : int

**Vortex**
- damagePoints: int
- itemSprite: Image

+ Vortex()
+ scoreModifier() : int
+ moveCharacter(): Void

**pileOfBooks**
- damagePoints: int
- itemSprite: Image

+ pileOfBooks()
+ scoreModifier() : int

**Wall**
- wallSprite: Image
- wallLength: int
- isMapBorder: bool

+ Wall()

**goodTile**
- tileSprite: Image
- tileItem: rewardItem

-tileSteppable: Bool
+ goodTile()
+ getTileSprite(): Image
+ getTileItem(): rewardItem

**neutralTile**
- tileSprite: Image

+ neutralTile()
+ getTileSprite(): Image

**Map**
- tileGrid: Tile[][]
- Walls: Wall[]
- Player: Hero
- enemyList: Enemy[]

+ Map(tile[])
+ loadMap(): Map

1..4

1...256

1..40

****
**Tile**
- tileSprite: Image

+ getTileSprite(): Image

**badTile**
- tileSprite: Image
- tileItem: punishmentItem

+ badTile()
+ getTileSprite(): Image
+ getTileItem(): Item

**timeElapsed**
- secondsElapsed: int

+ timeElapsed()
+ timeElapsed()
+ setTime(): void
+ getTime(): int

**Score**
- currentScore: int

+ Score()
+ setScore(int): void
+ getScore(): int

**HUD**
- currentMap: Map
- currentHUD: HUD
- currentScore: int
- time: timeElapased

+ HUD()
+ displayHUD(): void

**Game**
- currentMenu: mainMenu
- currentMap: Map
- currentHUD: HUD
- currentPlayer: Hero
- gameMusic: Sound

+ main()
+ startGame(): void
+ loadHUD(): void
+ endGame():void
+ pauseGame(): void
+ makeSound(): void
+ modifyHeroSpeed(): void

0...3

**mainMenu**
- backgroundSprite: Image
- difficultyLevels: String[]
- menuOptions: String[]
- menuButtons: Image[]

+ mainMenu()
+ startGame(): void
+ displayMenu(): void

**mysteriousSmokeTile**
- slowSpeed: int
- handicapTime: int
- tileSprite: Image
- smokeSprite: Image
- isVisible: bool

+ mysteriousSmokeTile()
+ speedModifier(): int
+ engageSmoke: Void