Design Document – HW5

Deliverable 1

Ethan Meli & John Kim

Gameplay Requirements

This game is a rough combination of The Game of Life, Monopoly, and Shoots and Ladders. It is played by rolling a dice to move spots on the board. This will be a **2 player board game.**

Each space has a different event, such as:
Move Forward or Backward
Earn or Lose Money
Gain Power-ups or Upgrades

Players can use power-ups that they collect as they traverse the board. These may help them sabotage the other player or provide strategic advantages.

Playing Field:

Board in the center: The main area with traversable tiles.
Player Sections (Left and Right): Two distinct sections, one for each player. These sections will be greyed out depending on whose turn it is.
Dice Display in the middle: A central UI element showing the dice roll outcome.
Power-up Abilities: Each player will have access to their available power-ups, which will be displayed in their section of the UI.

The Common Theme:

The struggle of a Computer Science student (minimal success possible). The theme will include references to common student life challenges, such as deadlines, coding struggles, and small successes.

Game Pieces:

- Dice: Roll to dictate amount of spaces to move
- Power up “cards”
- Player pieces
- Dollars

Players will collect items by landing on specific tiles on the board.

Power-ups/Upgrades:

- Sabotage other player upgrade (purchased with dollars)
- Shop items able to be purchased with money with several different upgrades
- Power up cards, unique and granting the player 1-time abilities

---

Begin State:

All players start at the same beginning tile, with the same amount of starting dollars (and other stats).

End State:

All players reach the end tile.

Design Patterns:

Singleton Pattern for the game loop manager (e.g., LifeManager.cpp). So that there is only one instance managing the game state.

Factory Pattern for creating board tiles (e.g., TileFactory.cpp). Different tiles (e.g., forward, backward, earning money, powerups, etc) will be created using this pattern.

Tile Class:

Base class representing a board tile.
Derived: Specific tile types, PowerupTile, MoneyTile, JoblossTile, etc

PowerUp Class:

Manages power ups that are available to players.
Subclasses for each type of power up: MovementPower, StealMoneyPower, etc

Player Class:

Represents player status and current position on board. Also handles stats like amount of dollars, power ups, etc.

Board Class:

Manages the state of the game board. Initialization and movement logic will happen here.

Rough Method Layout:

LifeManager

- static LifeManager* instance() – single instance of the game manager
- void startGame
- void endGame
- void nextTurn
- bool checkEndCondition
- void updateUI

Board

- void initializeBoard
- void movePlayer(Player* player, int x) – Moves player x spaces
- void tileAction(Player* player) – For tile effects
- Tile* getTileAt(int index) – returns tile at given index

TileFactory

- Tile* createTile(TileType type) – Creates a tile of a given type

Tile

- int getType

PowerUp

- QString getDesc
- Subclasses – MovementPower, StealMoneyPower

Player

- void move(int spaces)
- void collectPower(PowerUp* power)
- void usePower(PowerUp* power)
- void updateStats
- int getPos
- void setPos

Game Loop Flow:

startGame starts the game and initializes board, player, and other startup related data. Set up the UI.
Player Turn is initiated by **nextTurn**, and the player will be able to roll a dice, move, check the tile (tileAction), and receive certain things depending on the tile type.
Game will end when the **checkEndCondition** says the game has ended. **endGame** will be called to determine the winner.

Deliverable 2 goals:

Board Initialization:

The game board should be fully initialized with all tiles (e.g., forward, backward, money tiles)
Tiles should be randomly placed or based on a predefined pattern.

UI Implementation:

The player sections should be visually distinguishable and greyed out during the opponent’s turn.
The dice roll should be displayed in the center of the screen.
Powerups should be visible and interactable.

Basic UI Flow:

- Display the dice roll (might just be a text field) in the center.
- Highlight the current player's section (left or right).
- Display player stats (position, money, power-ups).
- Show the board and update the player’s movement after each dice roll.
- Display power-ups as they are collected and used.