Game Proposal: Escape Phase
CPSC 427 - Video Game Programming

Team Members:

James Zhu	26196287
Kenji Hiebert	85676914
Leo Foord-Kelcey	83974592
Oliver Cowley	91454381
Ricky Lau	86339603
Xinru Jiang	94227386

Story and Gameplay:
Our game is a 2D technical platformer built around the theme of “layers” and switching between them to complete levels. The background story starts with the main character catastrophically messing up their lab experiment, which splits the world into different layers and teleports them far away from the lab. Before their supervisor finds out, they’ll need to navigate through the interleaved world layers and get back to the lab to set things right.

Game levels are split into four regions: the lab, a cave, a forest and a city. Each region will consist of several predetermined levels and introduce new hazards/enemies. Anytime during a level, players will have the ability to switch between different “layers”, which will provide a different view of the same level. For instance, some platforms only appear on Layer 1, while others only appear on Layer 2. Players can only interact with the environment in the same Layer they’re in, but enemies/hazards can affect the player irrespective of their Layer.

There will also be power-ups scattered semi-randomly throughout a level. When encountering a powerup, the player will gain a persisting powerup. On collision with an enemy, projectile or hazard, the player is sent back to the most recent checkpoint and will also lose their powerup, if they had one.

Technical Elements:
- **Rendering**: Each level (with its constituent platforms, hazards and layers) will be rendered, along with the player character. We will also render a start and level selection screens.
- **Assets**: Textures for platforms, enemies/hazards and the player sprite will be loaded using sprites. Animations for player and enemy movement will be included using sprite sheets. We will also have audio assets for sound effects and background music. Most will be publicly available or AI generated; some may be created by the team or commissioned from artists.
- **2D geometry manipulation**: The player character and enemies will need to be translated within the game world to simulate movement. We will also need to detect collisions with walls, platforms, enemies/hazards and collectibles.

- **Gameplay logic/AI**: Enemies will travel along paths and may have different effects on the player. We will also need to keep track of projectiles and player interactions with objects between/within layers.
- **Physics**: Gravity based physics, which will affect player/enemy movement and falling objects.

Advanced Technical Elements:

Unique gameplay mechanics on each layer

- Layers may have distinguishing characteristics (e.g. different gravity strengths, limited visibility, lower friction floors, etc.)
- Impact if not implemented: Lower depth of gameplay experience
- Alternatives: do not implement

Scorekeeping

- The game will keep track of the number of deaths, time taken to complete a level and other relevant metrics to calculate a score for the player's performance in a level
- Impact if not implemented: Players may not be motivated to find optimal solutions to levels
- Alternatives: use the number of player deaths as the only metric

Enemy tracking and pathfinding

- Enemes will detect player position and path towards them automatically
- Impact if not implemented: Enemies are more predictable; easier gameplay
- Alternatives: Enemies move along pre-determined paths

Devices:

Keyboard:

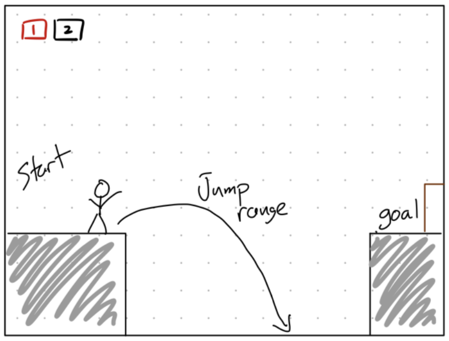
Layer shifting: 1, 2, 3, (4)
Directional movement: Arrow keys
Sprint: Shift+directional or double tap directionals
(UI navigation: arrow keys)

Mouse:

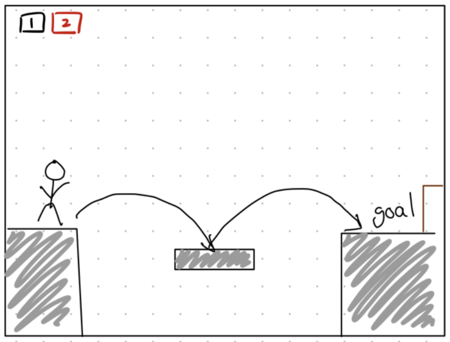
UI navigation: Main menu and level selection

Concepts:

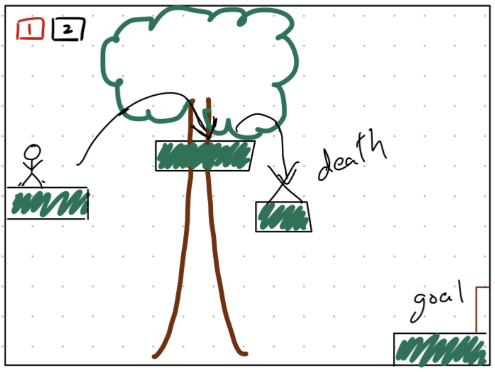
Phase 1



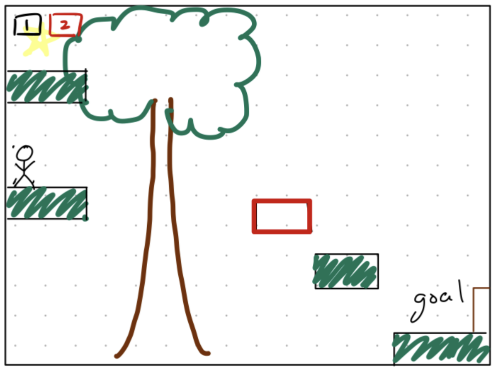
Phase 2



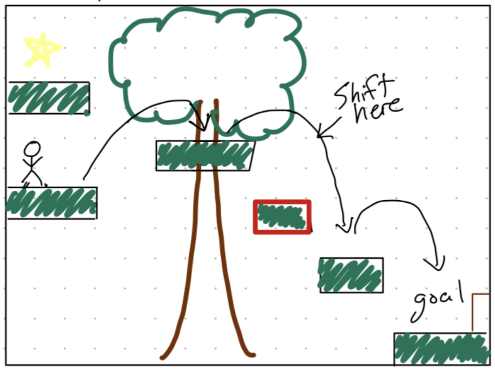
Phase 1



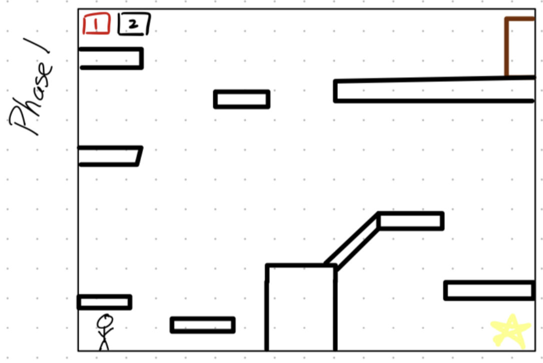
Phase 2

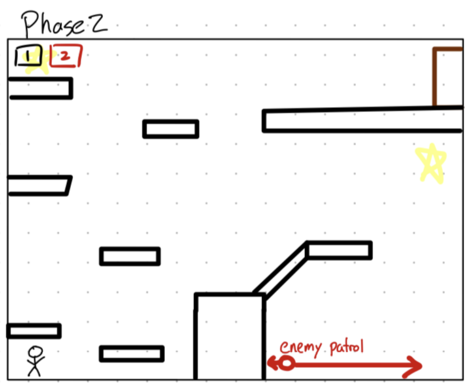


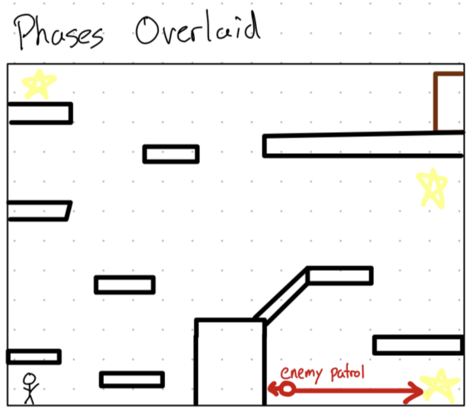
Both phases overlaid

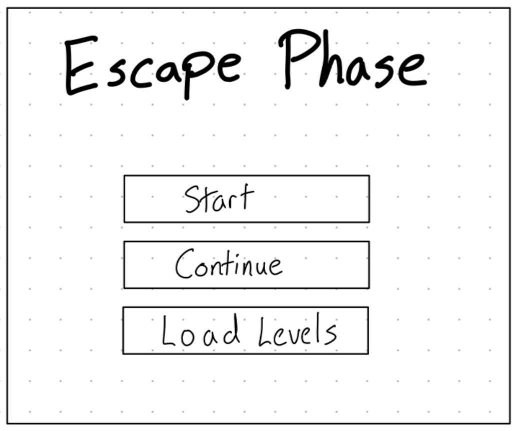


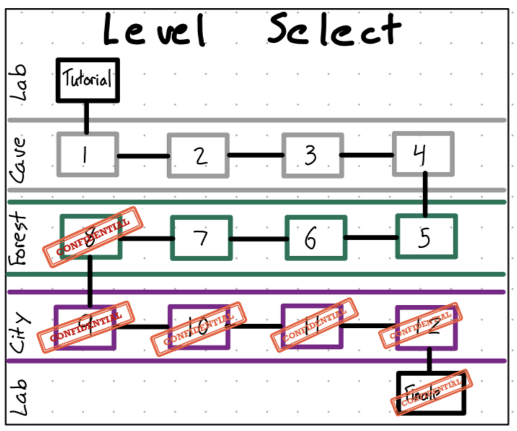
(not shown to players)











Tools:

Specify and motivate the libraries and tools that you plan on using except for C/C++ and OpenGL.

- gl3w: OpenGL function pointer loading (header-only)
- GLFW: Cross-platform window and input
- SDL/SDL mixer: Playing music and sounds
- stb.image: Image loading (header-only)
- glm: The GLM library provides vector and matrix operations as in GLSL

- Json-loader: Store and load game data

Team management:

We will conduct weekly planning and design sessions on Friday. Additional open working sessions will be conducted on the weekends.

Tasks will be tracked using GitHub Projects and GitHub Issues. A local feature branch will be created corresponding to a GitHub issue, and PRs will require at least one approval before merging to main. Merge conflicts will be resolved by rebasing to the latest main, and commits will be squashed when merging.

Development Plan:

Skeletal Game (M1 due Oct 12)

Week 1: Sep 25~

- Initialize repo (fork A1)
- Layout the hierarchy/structure for levels
- Rendering of first level and layer (greybox)
- Rendering of main character (greybox)
- Basic player control (move left/right, sprint)

Week 2: Oct 2~

- Advanced player control (jumping)
    - Stretch goal: layer shifting
- Introduce basic enemies and collectibles
- Basic collisions with enemies and collectibles
- Define boundaries of level

Week 3: Oct 9~

- Further stability enhancements (continuous execution and graceful termination)
- Basic physics (gravity influenced jumping)
- Basic level playthrough from start to end
- Additional testing

Minimal Playability (M2 due Oct 30)

Week 1: Oct 16~

- Introduce randomized powerups
- Introduce randomized powerup locations (choose among hard-coded spots)
- Implement sprite sheet animation for player movement
- Introduce basic layer shift (if not completed in week 2 of M1)
- Add basic tutorial to game

Week 2: Oct 23~

- Further stability work (lack of lagging, consistent resolution and aspect ratio)
- Introduce randomized movement of enemies

Playability (M3 due Nov 20)

Week 1: Nov 6~

- Implement additional regions
    - New enemy and reward sprites
- AI-based movement of enemies (Minimax with Alpha-beta pruning)

Week 2 Nov 13~

- Shadowing
- BGM soundtrack
- Handling unexpected user input
- Physics based animation of (enemy) attacks

Final Game (M4 due Dec 4)

Week 1: Nov 20~

- Complete implementation of all regions and all levels
- Perfect user experience - Tutorial should be self-explanatory and easy to understand
- Optimize ease of navigation between different layers

Week 2: Nov 27~

- Fix any of the previously discovered bugs
- Robust end-to-end user testing
- Additional third party physics engine integration