**KF6018 Computer Graphics and Animation Application Report**

The application is a prototype of a small 2.5D (2D gameplay with 3D graphics) video game. The game is a local multiplayer platform shooter for two to four players (played on a single keyboard). The players play against each other. They can move around the map and shoot projectiles, which cause knockback to others. If a player is knocked down from the environment platforms, they lose a heart. If a player loses three hearts, they are eliminated. Last player standing wins the game.

Although there should not be any performance issues, it is recommended to use the best device available. What FPS (frames per second) the application runs at can be checked in the web browser’s console (*Ctrl*+*Shift*+*J* in Google Chrome). All physics and animations are delta-timed, meaning the game’s logic is executed at the same pace regardless of framerate.

The application requires a web browser and Python to be installed (Google Chrome and the latest version of Python are recommended). The user must navigate to the folder the application was downloaded to and run the command: python -m http.server --directory . , then open the web browser and go to the web address: 127.0.0.1:8000/index.html . The webpage should display a PLAY button; pressing the button will start the game.

**PLAYER**: Each player is represented by a small colour-coded figure holding a large grey canon. They can move left and right, jump, fall through a platform, and shoot. Every action has some animation associated with it, for example walking on the ground will emit dust particles, shooting will spawn a bullet and cause knockback to the figure and recoil to the canon and emit fire and smoke particles. Player movement works as expected: gravity affects the player if they are not grounded, repeated jumping is prevented, coyote time is implemented for better experience. If two players collide, they repel each other. If a player is knocked down from the platforms, they fall to their death, losing a heart, upon which they are respawned again and fall in from the top of the screen. For a short period of time, they are invincible to other players’ bullets, indicated by a flashing animation.

**BULLET**: Bullets are spawned by players when they shoot. They fly, slightly bobbing, across the screen in the direction they were fired at, emitting fiery particles, until they collide with an object (a platform or a player) or get out of the bounds (in which case they are destroyed to lower the object count). Upon collision, they explode into fiery explosion particles that turn to smoke. If a bullet collides with a player, it knocks the player back in the direction the bullet was travelling.

**ENVIRONMENT**: The environment consists of several flying platforms and a dark “low-poly”-styled background. The platforms are propelled by thrusters, which emit fire particles that turn to smoke. The platform hitboxes are one-way, meaning the player can jump through them from below. The background is randomly generated at the beginning of each game.

**CAMERA**: There are two cameras: perspective camera shows the game scene and orthographic camera displays the HUD (head-up-display). The perspective camera moves around the game depending on the current position of the players and zooms in and out depending on their proximity to better display the action. Camera shake animation of different strength and length is played when a player fires their canon, a bullet collides with an object or when a player falls to their death.

**HUD**: Simple HUD with fade-in/fade-out animations shows colour-coded hearts of each player and displays a winner screen if a player wins the game. Arrows are spawned if a player leaves the bounds of the screen to show their position (however these are rendered with the perspective camera to accurately reflect the position).

Many of the game’s values can be edited in the file called “global.js” in the “game” folder, such as the number of players, maximum health, or the players’ controls; note that the webpage needs to be hard-reloaded after any change (*Ctrl*+*Shift*+*R* in Google Chrome). Some of the values from this file would be turned into a settings menu if the game was to be developed further.

Default Control Schemes (left, right, jump, fall through platform, shoot): [P1: A, D, W, S, R]; [P2: J, L, I, K, P]; [P3: F, H, T, G, U]; [P4: Left, Right, Up, Down, N]