**Game Specification Form Student ID: trtm63 Level 3/4**

|  |  |
| --- | --- |
| **Marking Criteria** | **Describe how your game matches the criteria** |
| **Game design (10%)** | |
| Game Goals: | Primary Goal: Complete all 5 levels by collecting both treasures in each level (so 10 treasures in total). Need to avoid enemies in order to not lose lives and need to avoid falling off the platforms to reach both treasures in each level.  Secondary Goal: A level goes on forever once its treasure is collected, so you can try to beat your high score (each level has its own high score) |
| Game Type: | Platformer |
| **Core development (30%)** | |
| Game scene (visual representation [2D, 2.5D or 3D], internal data structure): | 2D  The game scene consists of platforms and the enemies/items/treasure that spawn on top. These platforms are randomly generated as you progress upwards. Only the first five platforms are hard-coded. Each level also contains a different platform type, music and background. Enemy/Item spawn rate is also different per level.  Game state is updated once the player reaches a certain height on the screen, after which the camera shifts upwards and new platforms/enemies/items/treasure spawn.  Individual level high scores are persistently stored. Whether a level was completed (collect both treasures) or not is also persistently stored. Both use text files to do so.  The main game is coded in the main.py file. I also have a settings.py file containing solely variable names/values and a sprites.py file containing classes for all objects (like player, platforms, enemies, etc.) that appear in the game. |
| Game flow / game progression (e.g., navigation, screen scrolling, levels): | Infinite Screen Scrolling Upwards -> You have to jump from one platform to the next, avoiding enemies or falling off the platforms and collecting the treasure when it spawns.  5 Levels, each introducing new enemies and increasing in difficulty.  Player Navigation is done using left/right arrow keys and the keyboard  Menu Navigation is done using specific keys on keyboard as instructed on the screen |
| Game interaction (e.g., action detection and response generation): | Walking off the screen from the left side makes you appear on the right side (and vice versa)  Player firmly stands on platforms (cannot fall through platforms)  Using collision masks to have more accurate collisions against enemies  The following items spawn randomly on platforms and give you a special effect when colliding into them:  Heart -> Gives you an extra life  Bomb -> Kills all enemies  Star -> Gives you immunity for a short while against enemies  Collision against enemies makes you lose a heart  Collision against enemies gives you a 2.5 second immunity from being damaged by an enemy (basically a 2.5 seconds recovery period)  Collision with items/treasure/enemies makes sounds  Collision with the gold treasure will complete/end  Falling off the edge of platforms makes you lose the level  Once a platform is not on the screen anymore due to you jumping upwards, your score increases by 1 |
| Game object (e.g., use of sprite, 3D objects, animation, multimedia): | Sprites are used on platforms, enemies, items, treasure and the main player  Sprites are used to animate the player and enemies  Each level and menu screen has music playing  Sounds are played when certain events occur, like falling of the screen, colliding with the enemy or picking up an item/treasure  Images from noteworthy locations in Durham are used as backgrounds images for each level |
| **Game mechanics (30%)** | |
| Game rules / logics: | Falling off the lowest/last platform makes you lose the level  Can pick up item/treasure by colliding with it  Picking up an item will result in some effect  Picking up the gold treasure will complete/end the level  Your score increases by 1 each time a platform disappears from the screen due to you jumping upwards  You start each level with 3 hearts  Hitting an enemy makes you lose a heart  Once all lives are lost, the game is over |
| Game challenges: | Still-standing and moving enemies can decrease your hearts  The higher the level, the more enemies will appear.  The higher the level, the more the difficult types of enemies will spawn.  Different enemies have varying movement speeds.  Only have a limited amount of platforms on the screen, so falling off all platforms can easily happen. |
| **Good use of game engine (15%)** | |
| Choice (pyGame, Unity): | pyGame |
| User input (keyboard, mouse, joystick): | keyboard |
| Game object interaction (e.g., event triggering, collision detection): | Collision detection happens between player and platforms/enemies/treasure/items  A collision mask is used for collisions between player and enemies to make the collisions more accurate/reliable  Event triggering is used to navigate main menu / level selection and to control player during levels |
| Incorporate multimedia content: | Sprites are used for main player/enemies/items/treasure/platform  Sounds are used when certain events occur  Each level/menu has its own music  Each level has its own background picture (location within Durham) |
| Other features used (e.g., asset, incorporation of external libraries): | Only the pygame library was used |
| **Demonstrate creativity (15%)** | |
| Game economy (e.g., support to game type, game feedback, game difficulty): | Game difficulty is increased as you do higher levels in three ways:  New Enemies -> Later levels have enemies that are stronger/faster  Spawn Rate -> New and existing enemies spawn at higher rates in later levels  Level-specific movement: Slippery ice platforms and strong sideways wind can make player movement harder on level 2 and 3 respectively  To help with increasing difficulty, types of items and item spawn rate also increase in higher levels |
| Advanced Interaction (e.g., game physics, object tracking, steering behaviour): | Player steering incorporates velocity, acceleration and friction -> these values are adjusted on certain levels (like snow level where it is more slippery (less friction) or bridge level where there exists sideways wind  Player steering wraps around each level (going left off the screen makes you end up on the right side of the screen and vice versa)  Random platform generation occur in a manner that it should almost always be possible to reach the next platform as you are infinitely going upwards  The player is tracked by constantly calculating its height and moving all objects down when the player reaches a certain height to make room for new platform/enemy/item/treasure spawns. |
| **Game optimisation and configurability (50%) [For Level 4 Students Only]** | |
| Include optimisation to enhance game performance (e.g., game related functions, game scene and objects, interaction, rendering, media content): |  |
| Make the game flexible to support making changes (e.g., game scene and objects, game flow / progression): |  |