Modern Computer Games  
COMP 521, Fall 2019  
Assignment 1

Due date: Wednesday, Oct 2, 2019, by 6:00pm

Note: Late assignments will only be accepted with prior written permission of the instructor. You must explain all answers and show all work to get full marks! Please make sure your code is in a professional style: well-commented, properly structured, and appropriate symbol names. Marks will be very generously deducted if not!

Description

This assignment requires you implement a relatively simple “mini-game” in Unity3D. You will thus need to install and become familiar with the Unity game development environment from “http://unity3d.com/”. This is a commercial product, but for everything you do in this course the free (personal) version is more than sufficient. Please use the latest downloadable 2019 version (currently 2019.2.4 or 2019.2.5), and restrict your usage of Unity assets to free versions. Note that Unity is mainly supported on Windows and Mac, although Linux is possible as well.¹

The tasks below focus on building game mechanics and structures. Aesthetics are not a factor in grading—solid objects should be opaque rather than wire-frame, but you do not need to find or use external models or textures; creative use of basic assets and asset-combinations can be used to accomplish all objectives. Assume a first-person perspective is required unless otherwise specified.

Also note that the Unity site has links to helpful forums and short tutorials on using Unity for different purposes, and those will be your main resource for dealing with the software.

Overall design

The main, static play area consists of a large rectangular area, styled as an outdoor terrain (add some vegetation and ground texture; small plants are ok, but no trees). The area is fully surrounded by impassible terrain (mountains or walls), to ensure the player does not leave the area. At one end of the rectangular area is a dense, 8x8 “forest” area, structured as a maze (see below). This area is surrounded by a wall of trees (around the 8x8 shape), with no obvious entrance. Floating above the (non-forest) area at random locations (and visible but just out of reach) are 3 objects.

The player begins in the rectangular area, and is able to move (keyboard/mouse), jump (space), and fire projectiles (left mouse button). Jumping should raise the player up so the player is about level with the floating objects, but not allow them to escape the scene in general or go over the forested area.

Player projectiles should be visible objects moving rapidly (but noticeably) in a straight line out from the player in the direction of the camera. They eventually encounter either a floating object, the forested area, or the boundary (i.e., try to leave the scene). In the latter two cases the projectile should just disappear on contact. Only one projectile may exist at a time, although there is no limit on the number that may be fired.

When a projectile encounters one of the floating objects, both it and the projectile disappear. Whatever order the player destroys the floating objects, upon disappearing the 3rd one drops a small object (key), which should visibly fall to the ground. Once the key lands, the player moving over the key should result in the key being “picked up” and disappearing from the scene.

With the key, a single entrance (i.e., 1 cell in the boundary of trees) to the forest maze should appear and allow entry into the maze. The maze itself is grid-based, using trees as obstacles, and should have only one exit (which can be anywhere). Movement through the maze should also be discrete (logically at least—the player is always on exactly one tile of the 8x8 grid). Within the maze the player should be able to jump to see much more of the maze, but cannot move while jumping (so it’s useful for observation only). Firing projectiles should have no effect or be disabled while within the maze.

The maze is constructed in 3D, even if it is presented and navigated in 2D. The 3rd dimension is time—whenever the

¹Versions lag behind some, currently version 2019.1.0f2. See thread https://forum.unity.com/threads/unity-on-linux-release-notes-and-known-issues.350256
player moves to a new grid cell in the maze they also advance in the time dimension, and trees should appear and disappear according to the current state of the maze at that time. Should this result in the player being blocked and unable to move, or if they voluntarily press the `esc` key at any time, reset the maze and teleport the player back to the start. Ensure the maze is solvable in at most 16 steps, and if the player exceeds that maximum time then reset the maze and teleport the player back to the start.

The maze should be different every time the game is started. You may use any algorithm you wish to create the maze. It does not have to be a perfect maze nor does it need to fill the entire grid, but it should include multiple dead-ends. Note that not every non-tree cell at every time needs to be reachable; each time-state of the maze should depict a significant part of a maze, but due to future changes and inability to backtrack in time those parts may not be accessible.

To help with grading, the next step toward the maze solution should always be clearly, visibly indicated, such as by giving a different colour to the tile the player should step on next to make progress (at least whenever a solution is still possible). Once the player gets through the maze the game terminates (win). A visual indication should be given, and no further action should be possible by the player.

**Specific requirements**

1. You must provide a non-trivial initial, static terrain as described above. It should be bounded and styled as described above. Use a standard WASD/mouse controller for motion and looking around, with a a first-person camera view. Ensure sufficient ambient or other light to easily see within the terrains and maze.  

2. The player can fire one projectile at a time, which moves visibly and directly away from the camera in a horizontal plane. Projectiles terminate at scene and forest boundaries.  

3. The three floating objects should exist, be destroyed by projectiles, with the 3rd one dropping a key for the player to pick up, as described.  

4. Players can jump, reaching a height sufficient to fire projectiles that hit the floating objects, and also to observe the maze structure from above.  

5. The maze structure is uniquely generated on each playthrough, with structure and behaviour as described.  

   a. Sized $8 \times 8 \times 16$.  
   b. Appropriate wall and floor representations.  
   c. Dynamically and randomly generated to be (highly likely) different in each play.  
   d. Solvable, maze-like, with dead-ends.  
   e. Time (3rd dimension) moves forward as a player steps through the maze.  
   f. The maze can be reset.  

6. The next step in the maze solution should be indicated, and winning state detected.  

**What to hand in**

Assignments must be submitted on the due date **before 6pm**. Submit your assignment to *MyCourses*. Note that clock accuracy varies, and late assignments will not be accepted without a medical note: **do not wait until the last minute**.

For the Unity questions, hand in an exported project containing all files needed in order to reconstruct and run your simulations. Note that Unity exports can be extremely large, and take non-trivial time to upload (another reason last-minute submission may not work out well).

This assignment is worth 15% of your final grade.