420-J11-AS Game Engine III 2nd December 2019
Felix Soumpholphakdy

Sample Final Exam

In this final exam, there are 10 C++ problems. These problems will test your understanding of the Unreal Engine 4 actor and your ability to implement algorithms in C++.

Start with a base First Person C++ project. All implementations are done through programming and nothing through Blueprint. Properties that should be editable from the editor are underlined.

Problem 1. Player HP

[10 points] Add HP (health points) to the player class. This should be a float to represent the ratio (%) of HP. Create a Widget Blueprint to visually show this HP as a progress bar on the player's screen. Bind the value of the progress bar to the HP property on the player.

Problem 2. Caltrop

[10 points] Implement the class *Caltrop*, which are spike traps that harm the player when touched. They are visually represented by blue small cubes. They must have physics enabled on their mesh component through code. Declare a damage property which is the amount of damage caused to player on hit. When hit, they damage the player and are destroyed.

Problem 3. Heal Kit

[10 points] [10 points] Implement the class *HealKit*, which is a box that can be picked up by the player on collision and heals them over time. This box is green and must visually rotate on the Z-axis. When activated, the player will regain gradually heal to full over a certain duration defined in seconds. Play a sound when this heal kit is activated.

Problem 4. Launch Pad

[10 points] Implement the class *LaunchPad*, which is a floor tile that launches the player forward and up when stepped on. This actor is visually represented by a yellow flat box. When the player physically hits the box, a forward force, relative to the player's orientation, is applied on them to push them forward. Additionally, a small up force is also applied to lift the player slightly. The value of both the forward and up force should be editable from the editor.

Problem 5. Proximity Mine

[10 points] Implement the class *ProximityMine*, which is a harmful trap that activates by proximity to the player. Visually, it is a black box. When the player is within a certain distance of this mine, it activates and turns red. It then waits a certain delay before exploding. When exploding, it spawns a particle system and damages the player if they stayed within the defined distance. Finally, projectiles are able to destroy this mine.

Problem 6. Lootbox

[10 points] Implement a system that simulates opening a lootbox. When the player presses the 'F' key, then this system will randomly pick an item based on defined weights. There will be a high chance (60%) to find a common item, a low chance (30%) to find a rare item, and a very low chance (10%) to find a legendary item. These chance properties should be constants that are easily editable in a header file. To simulate receiving an item drop, simply display a debug message to the screen (i.e. "You receive a common/rare/legendary item!").

Problem 7. Cubemon Enemy

[10 points] Create a Cubemon enemy with a HP property that is displayed as a progress bar above its head. Through code, this in-world UI must always face the player camera. Visually, the Cubemon is a cube with a second smaller cube on top of it. When hit by the player's projectile, the Cubemon loses 10% HP.

Problem 8. Furthest Cubemon

[10 points] Implement a skill that damages the furthest Cubemon within a certain defined radius (initially 500). Through code, find all Cubemons within the radius, then find the furthest one and deal 10% (0.1) damage to it. This skill is activated by pressing the ‘Q’ key.

Problem 9: Sort Cubemon HP

[10 points] Implement an enemy HP sorting utility that is activated by the ‘H’ keyboard key. This utility will find all Cubemons in the level, then sort them in order of health points in descending order. Then, print the name of each Cubemon and their name in the sorted order. Example:

```
Cubemon_1 has 1.0 HP
Cubemon_3 has 0.5 HP
Cubemon_2 has 0.25 HP
```

Problem 10. Alternate Laser Fire

[10 points] Implement an alternate fire mode for the player’s gun. The right mouse button will fire a laser following the camera’s forward vector. This laser will deal 80% (0.8) damage to the first enemy it hits. For every subsequent enemy hit, the damage will be halved. For example, the third enemy will only receive 20% (0.2) damage.