Module 5 - Items, Caves, Non-Sticky Walls, Shooting
Due. Friday, April 2, 2021 (note there can be no extension beyond April 4th for this assignment)

1. Items
Add items that the player can pickup in the game. When the player enters a square containing an item they interact with that item. Mobs do not activate items when they enter the square and they can pass through a square containing an item.

After the player interacts with an object the object is no longer drawn (it disappears).

New mesh objects have been added to the game. These are:

Mesh #	Item
4	sword
5	open chest
6	armour
7	potion bottle
8	key
9	ring
10	chicken leg
11	coin
12	scroll
13	fish bones
14	bow
15	closed chest
16	crystal
17	skull
18	arrow

The items can be created using their item number (4 to 18) in the same manner that the mesh objects numbers 0 to 3 were created.

To install these object in your copy of the game engine:

-download the file containing the new objects and unpack it
-copy the .obj and .ppm files into the models directory

Copy only the .obj and .ppm files. Do not copy the unpack directory into the models directory.

Randomly place an object in each room. The items to create are:

1. open chest - when the player interacts with this item, print a message to the terminal saying that treasure was found

2. armour - when the player interacts with this item draw a 2D armour icon in the lower left of the screen

3. sword - when the player interacts with this item draw a 2D sword icon on the lower left of the screen (above the armour icon)

4. key - when the player interacts with this item draw a 2D key on the lower right of the screen, a key is needed to activate the stairs down from the level on which the key was found. If a player does not have a key for the current level then the stairs down will not work for them. Once the player goes down the stairs with the key then those stairs are permanently unlocked and the user wont need a key in order to use those stairs again. The key is removed from the players inventory when they go down the stairs. The player will need to find a new key for each level before they can go down the stairs. There must be at least one key per level. Mark the key on the map 1 (the non-fog-of-war map).

5. coin - the coin produces the same response as the open chest, the coin should rotate around the y-axis while it exists.

2. Cave Level
Use Perlin noise to create an underground cave level. The floor to the level is flat. The ceiling of the level is created using the Perlin noise on top of a parabola. To create the ceiling, first use the parabola y = 1 - (x^2 + z^2)/2 (where x^2 and z^2 are powers of two). Use the range of +1 to -1 for x and z. You will need to scale the x and z coordinates so they are within the range of -1.0 to +1.0 for the equation. The result (y) will be in the range 0 to 1 so you will need to scale that up to a reasonable height for the cave. This will create a dome shaped cave. Once this is done, add Perlin noise to create a rough surface for the roof of the cave. Use the noise to offset the points on the parabola in the y axis. This will likely create some holes in the parabola and you should fill them in a manner similar to the outdoor terrain.

Apply textures to the cave walls so it is easy to see the elevation.

Place several of the responsive AI's from assignment 4 in the cave. They can activate when the player moves within 10 spaces of the mob.

Create five levels for the assignment. The levels are:
outdoor
maze 1
cave 1
maze 2
cave 2

The stairs up and down should appear on the map for the cave levels.

3. Non-Sticky Walls
When the player collides with a wall they should continue moving. They currently get stuck when they collide with the wall. They should slide along the edge of the wall. They should still collide with the wall and not be able to enter the wall space.

There are several ways to approach this problem. One is to notice when the player enters a new cube location in the x and z coordinates. If the player is trying to enter an occupied space in the world array they should not be allowed to move in the direction which crosses into the occupied space. At the same time they should be allowed to move in the direction that does not move into an occupied space.

For example, if the current (old) position is (25.5, 25.0, 10.3) and the new position is (26.2, 25.0, 10.5) then the player is moving one unit in the x axis in the world array (the integer value of the x coordinate). The player is also moving 0.3 in the z axis but is still in the y world coordinate of 10. The movement is to the world location (26, 25, 10). If that space is occupied in the the world array then the player cannot move into it. Their x coordinate cannot change to 26.2 and it must keep a 25 x coordinate. The player can still move in the z axis and can move from z=10.3 to z=10.1. This will allow the player to move along the wall but not enter the occupied space.

Described in another way, the player cannot move in the direction towards the wall but they can move in the direction that is parallel to the wall. So if the x coordinate is changing to a new square then you can move in the z direction. If the z coordinate is changing to a new square then you can move in the x direction. In both cases this is true as long as the new movement does not enter an occupied space.

There are other ways to approach this problem. You can use them if you wish.

4. Shooting
Use the mesh model for the arrow (number 18) to allow the user to fire arrows at the mobs. If the player has the bow then they can press the space bar to fire an arrow. The arrow should move in a straight line in the direction the player is facing. If it crosses a square containing a mesh then it hits the mesh. When this happens the mob is removed from the game.

Only one mesh is removed per arrow. The arrow has a range of ten squares. Only one arrow can be in flight at a time. If the player tries to fire an arrow while one is in flight then nothing happens. The player has an unlimited number of arrows.

Use line of site to determine the path of the arrow.

Place a bow on the first maze level. Show the bow on map 1.

Starting Location
Start the viewpoint on the outdoor level, near the stairs down.

Choosing Parameters
It is important to pick values for parameters such as colours, speed of objects, the effect of gravity so they are easy for the marker to see. If the effect of a parameter it isn't obvious or is difficult to see then it will be marked as missing or incomplete.

Make sure colours are distinct. Choose velocities for moving objects that are fast enough to be seen.

Coding Practices
Write the code using standard stylistic practices. Use functions, reasonable variable names, and consistent indentation. If the code is difficult for the TA to understand then you will lose marks.

As usual, keep backups of your work using source control software.