Escape Room

CS352: Computer Graphics & Visualization Lab

Project Report

Course Instructor:
Dr. Somnath Dey

Submitted By:

Subha V Gopal – 200001074

Govind Kizhakke Mepad – 200001025

Aryan Yadav - 200001010

Introduction

Our project is to create an Escape Room. Escape rooms generally involve finding clues and solving puzzles to help you unlock the door leading to the exit. So we designed a room with 5 clues and provided you with a hint by which you can find the password to unlock the door. You can move around and explore the room, interact with objects and enter the password to the locked door to exit the room. You can also toggle the background music in the room.

Specifications

The libraries required are:

- iostream
- stdio.h
- string
- vector
- cstring
- unistd.h
- glew
- glut
- freeglut
- glm
- SFML-Audio

To run the project, first install all the required libraries mentioned above. Download the object and bmp files and maintain their proper directory structures as mentioned in the code. Then build the code by linking the libraries for glut, glew and sfml-audio. The command for this is:

```
g++ code.cpp -lglut -lGL -lGLU -lGLEW -lsfml-audio
```

The key controls in the project are:

- Keyboard functions - You can move around the escape room using the WASD keys on the keyboard. Other keys on the keyboard also have functions which are mentioned in the game(Eg. pressing Q to return an object to its original position)
- Mouse functions - You can move the camera by moving the mouse in the direction in which you want the camera to turn to. You can left click objects with the mouse to interact with them and change their positions in some way to help you find clues.
- Password input - When the exit door is clicked, a keypad appears on the screen. The correct password needs to be entered through the keyboard, upon which the door will open. The door remains locked unless the password is entered correctly.
- Background music - You can use the keyboard key mentioned in the game(M) to toggle the background music on/off.

Functionalities Implemented

- **Camera and Movement**:

We implemented a FPS Camera in this project. Camera turns to the direction you move your mouse, enabling looking around using your mouse . It enables you to move using ‘W’, ‘A’, ‘S’, ‘D’ keys.

Looking around:

Whenever the mouse is moved, PassiveMouseFunc is called. It measures the amount the cursor has moved by using mouse positions. We need to store the angles at which the coordinate system is to be rotated, which will be controlled by the mouse input. The effect of looking right or left will be controlled by rotating the coordinate system around the Y axis (the angle associated with this is called Yaw), and that of looking up or down will be produced by rotating around X axis (the angle is called called Pitch) as the center of rotation.To apply this rotation to the scene, we will use a separate function. This new function will be called before drawing, and will take care of positioning the camera correctly before rendering the scene.

Moving Around:

glutKeyboardFunc is called whenever a key is pressed and glutKeyboardUpFunc whenever a key is released.Using the keyboard we determine the motion direction and calculate the new position of the camera and call the above mentioned function for positioning the camera correctly before rendering the scene. We also check if movement is possible before changing the camera position.

Reference: https://thepentamollisproject.blogspot.com/2018/02/setting-up-first-person-camera-in.html

- **Ray Tracing for Object Selection via Mouse Click:**

The main idea is to save the matrix of picking area to select buffer and draw selected object with each object name. Then calculate the minimum z-depth out of all of the objects being hit, as that will be the object that has been selected by the user.

If mouse click occurs,then we draw a z -axis ray to that point until the ray collides with the last 3D object model(in our case always house). Then we select a pick area around the click point, OpenGL will get matrix information of the pick area by changing the render mode from GL_RENDER to GL_SELECT. Finally, we will draw all objects with GL_SELECT mode. By this step, the vertices with the object name are stored to select buffer. To calculate z-depth we use the GL-RENDER mode to calculate the number of hits by the ray and store its information in a stack and iterate through it for minimum z-depth.

Reference : https://www.glprogramming.com/red/chapter13.html

- **Object Arrangement**

The objects in the room were handled and arranged through Blender, which is an easy way to scale or position object models as required. The objects were exported as .obj files and were loaded in the program. Some objects were created in Blender and some were open source online models.

Reference : https://free3d.com/

- **Object Interaction**

When the left mouse button is clicked, we use ray tracing to find the object that has been clicked on. The selected object is then transformed according to the needs(Eg. Translated, Rotated or some other action is performed) so that interaction with objects is possible.

- **Password checking**

We maintain some functions and variables to check the password input. The entered input letter is stored and is printed on the screen so as to give the illusion of the person actually typing in the keypad. When 5 letters are entered, the input is matched against the actual password. If correct, the door swings open, and if incorrect, the password variables are reset to the initial state. This is repeated until the door is open.

- **Displaying text on screen**

We have two different ways in which we display text on the screen. To display help text on the upper left corner of the screen, we use glutBitmapString() function which displays fixed size fonts. However it requires less memory and are easier to render, hence are a better alternative as this text needs to always be display on the upper left corner.
To display the entered letters on the keypad, we use the glutStrokeString() function. This displays scalable fonts and are more flexible, so we can give the illusion that the player is typing letters in the keypad by correctly positioning the printed text.

Reference : https://www.opengl.org/resources/libraries/glut/spec3/node1.html

- **Loading objects**

Objects were loaded through their .obj files. A .obj file contains details about vertices, vertex normals, vertex texture and face indices. Each of these were loaded into structures through the loadOBJ and constructOBJ functions in the code. Vertex Array Objects(VAO)s were created to hold object data, which was binded to it. The objects were then displayed through the drawOBJ function. The .obj files are named using a convention so that the function can just append the name at the end of the path to the object folder.

Reference : http://www.opengl-tutorial.org/beginners-tutorials/tutorial-7-model-loading/

- **Background music**

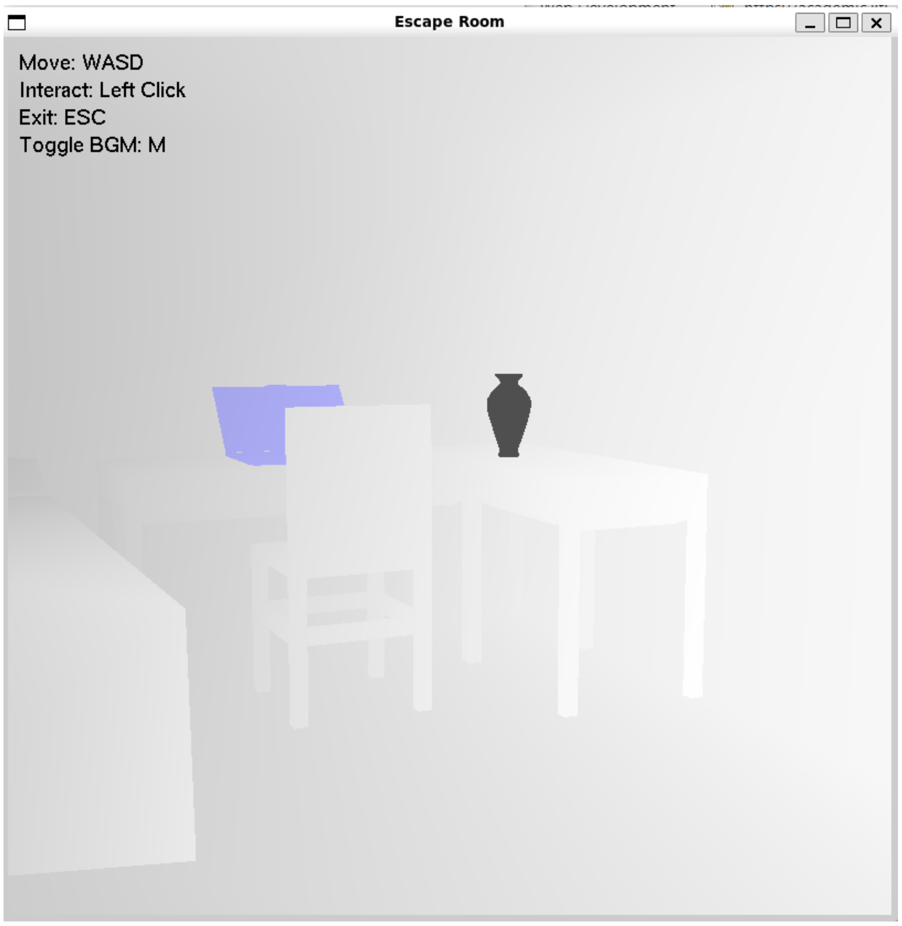
Background music was implemented using the audio part of SFML(Simple and Fast Multimedia Library) library, which provides an interface to include windows,graphics, audio, etc. in C++. We create a sound buffer and load the necessary sound in it. Then we create a sound object and attach the loaded sound buffer to it. We use the play() and stop() functions in SFML-audio to play and stop the music as required.

Reference : https://www.sfml-dev.org/tutorials/2.5/audio-sounds.php

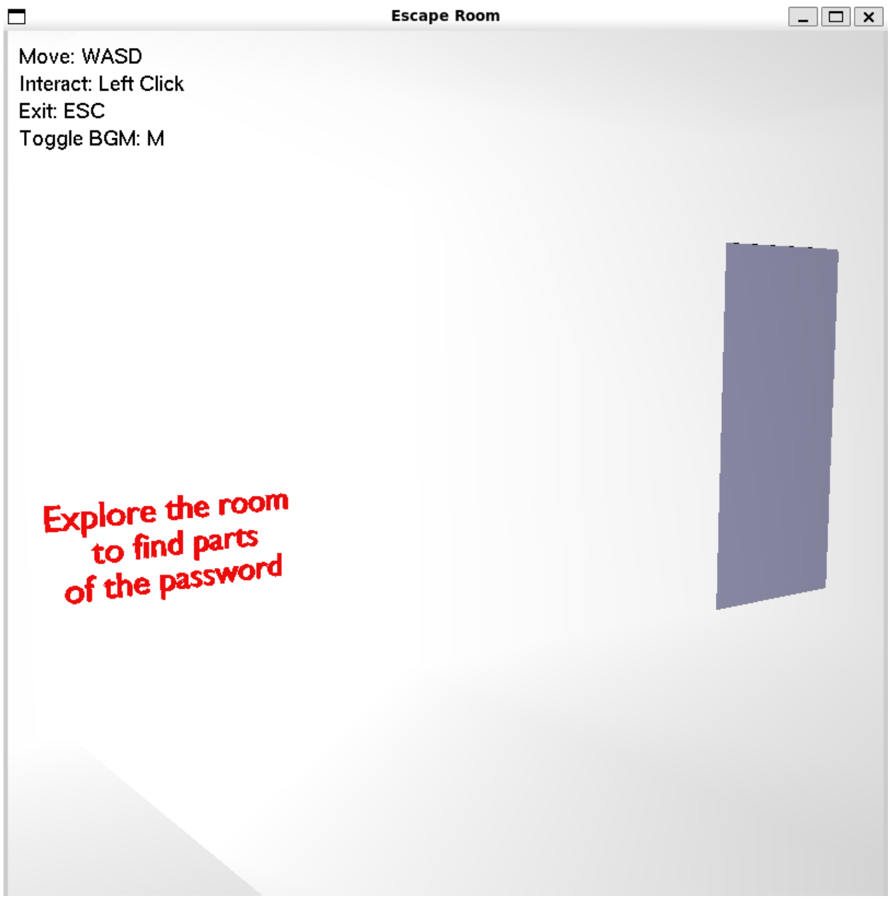
- **Moving Area**

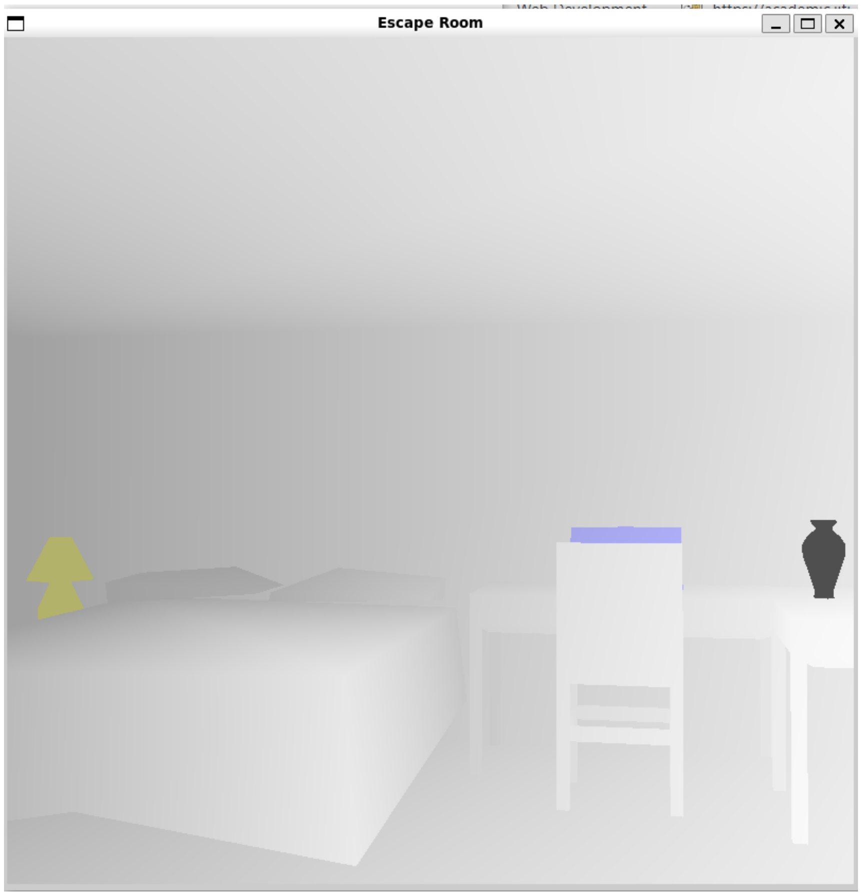
The moving area is hard coded and is checked in the checkMovement() function, which checks if the character can move from their current position as per the room arrangements.

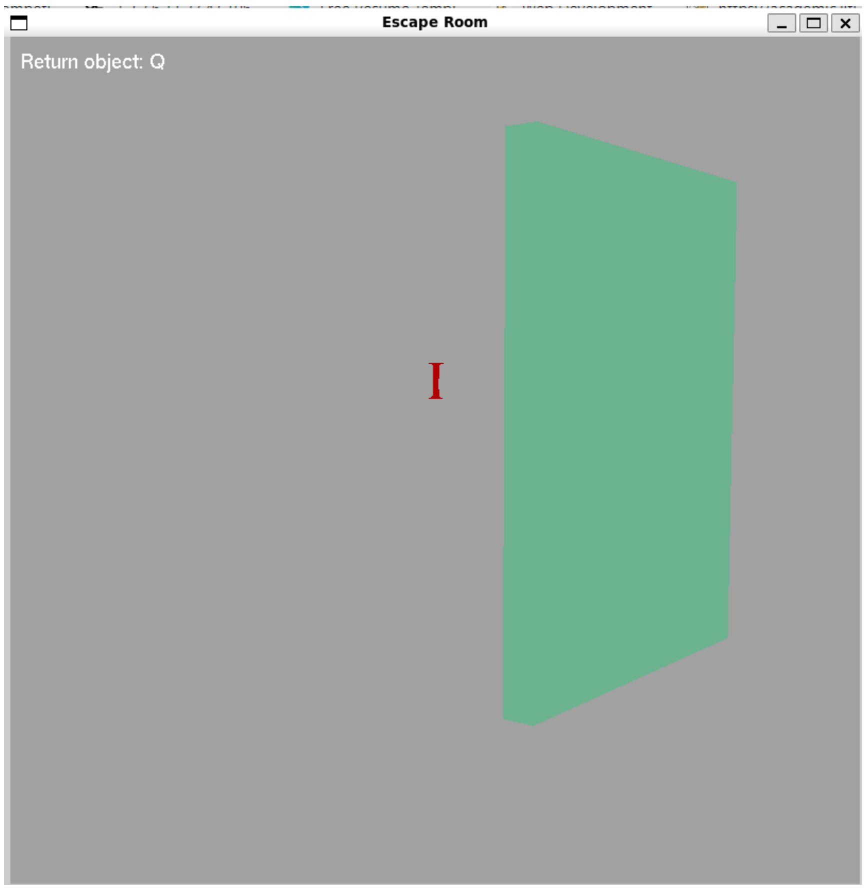
Output

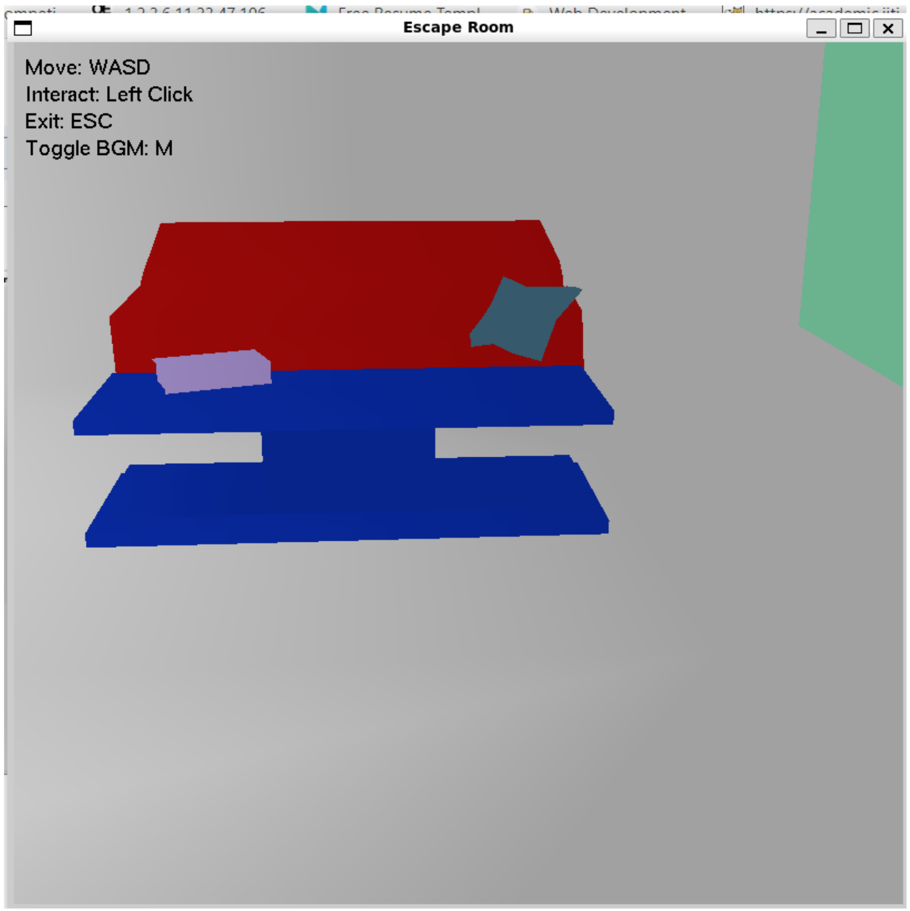


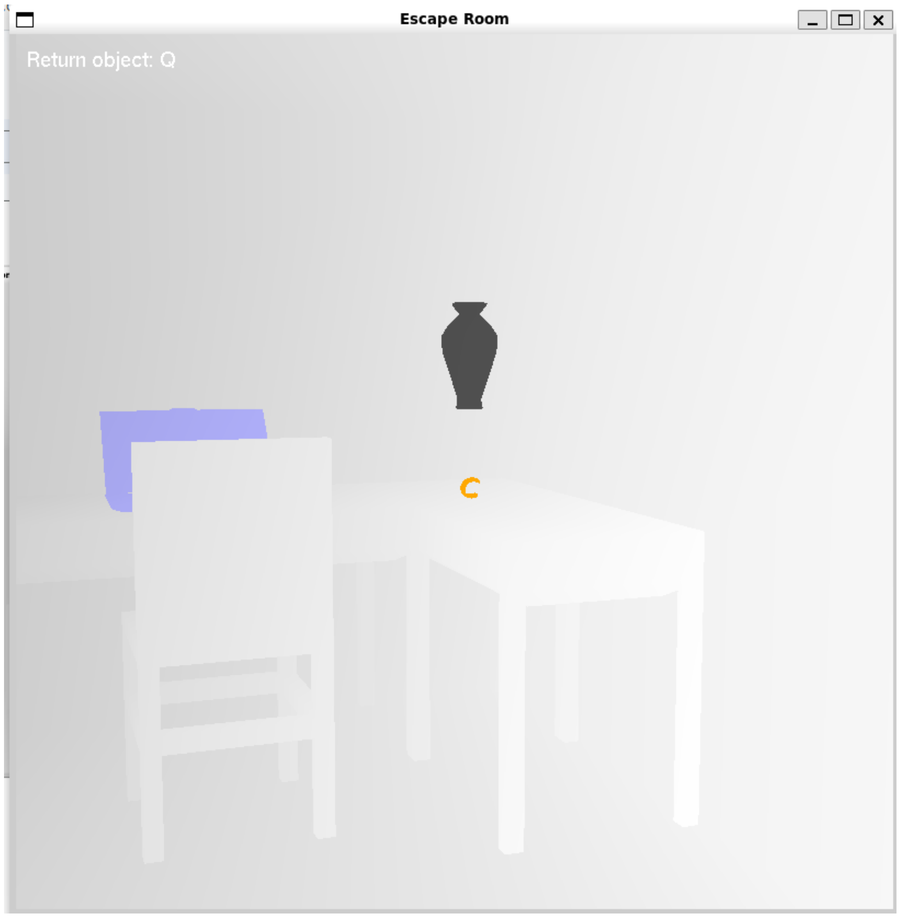
Move: WASD
Interact: Left Click
Exit: ESC
Toggle BGM: M

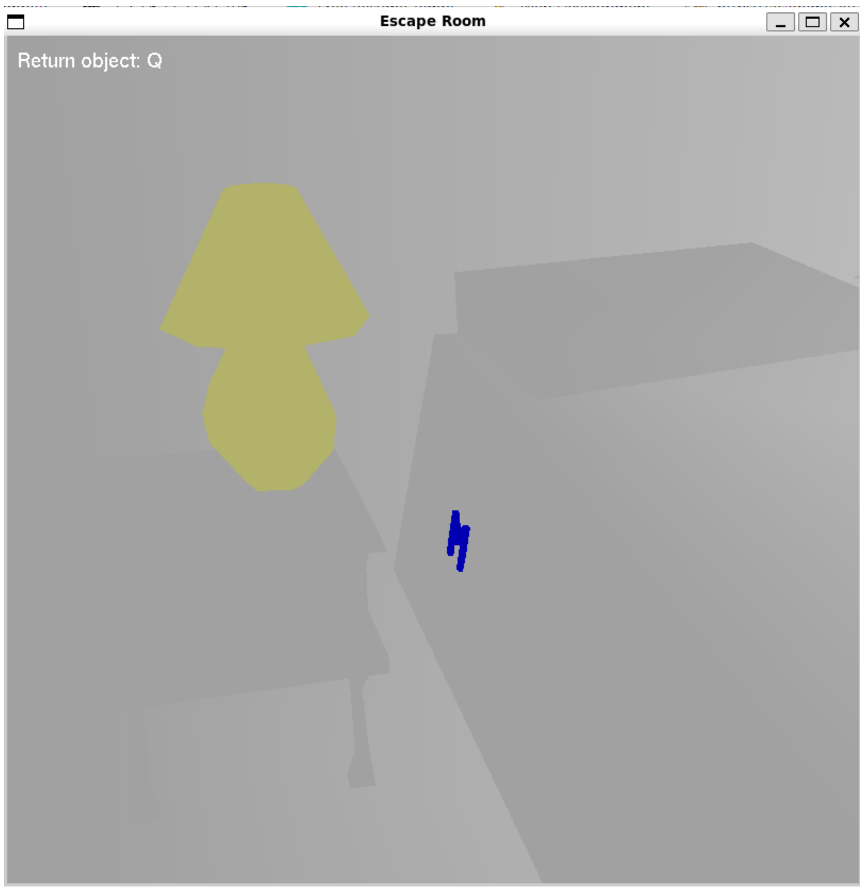

Escape Room
Move: WASD
Interact: Left Click
Exit: ESC
Toggle BGM: M
Explore the room
to find parts
of the password

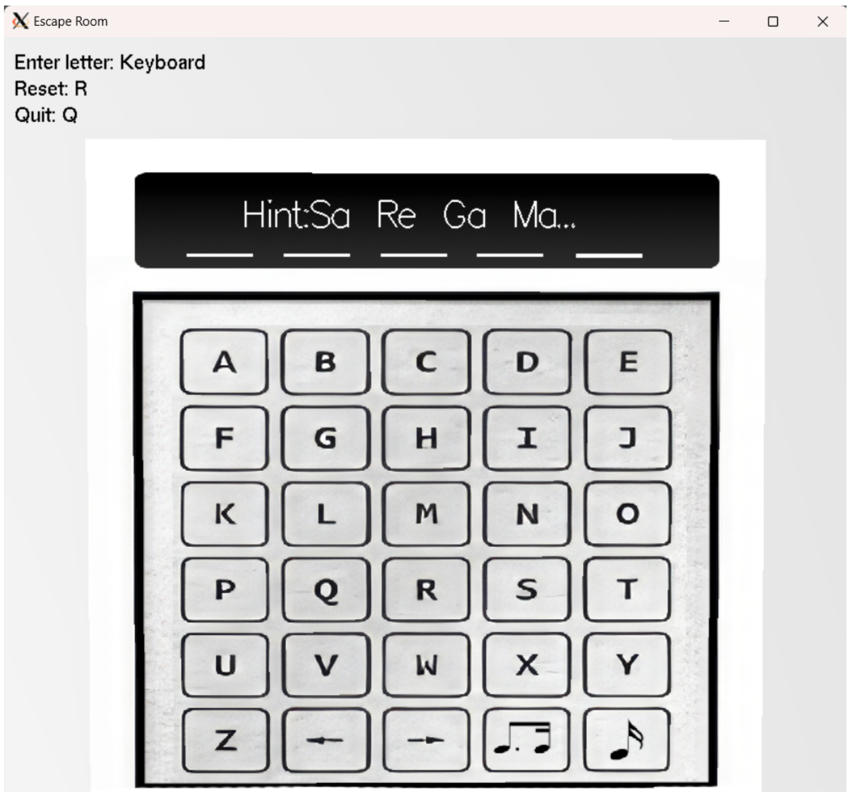

Escape Room

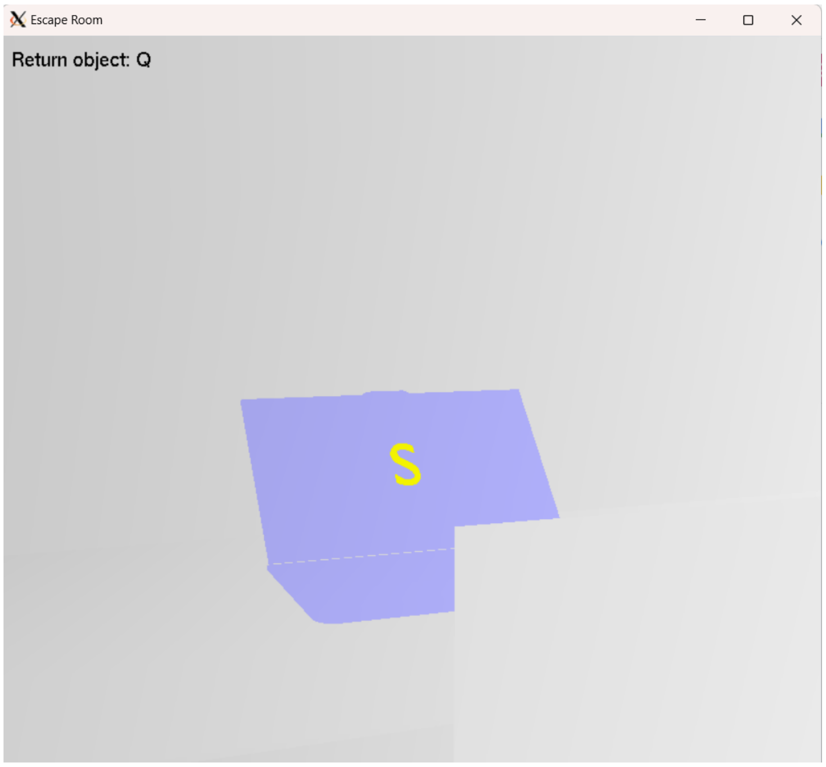

Escape Room
Return object: Q


Escape Room
Move: WASD
Interact: Left Click
Exit: ESC
Toggle BGM: M


Escape Room
Return object: Q


Escape Room
Return object: Q


Escape Room
Enter letter: Keyboard
Reset: R
Quit: Q
Hint:Sa Re Ga Ma...
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z



Return object: Q