Data Structures & Arduino Assignments
By: Jeldrik Gaal, Loai Qubti

Description:
You are a firefighter who is also an arsonist. In this game, feel free to explore and burn some trees in the forest, go to the firestation, or go home to your partner.
In this project, for the Arduino project: You could use the different attached sensors to interact with the game. As for the Data Structures project: the game uses three different data structures mentioned below.

Data Structures:
All data related scripts can be found in Project Folder > Assets > Scripts > Data Structures
Data is managed by GameManager.cs script (In Scripts folder)

The following are the data structures used in our project:

1) List: for Inventory (InventoryList.cs)
2) Tree: for Conversations (ConversationManager.cs, Tree.cs, Node.cs)
3) Graph: for Game Map (Graph.cs, GraphEdge.cs, GraphNode.cs)

List
We used the list for a simple changeable inventory. It adds, removes and finds items within the inventory. We did not utilize the inventory beyond picking up items; and for some cases requiring a condition (to have something in the inventory) in order to allow a certain action. Such as: having a card in the inventory in order to move through a door.

Tree
The tree was used for creating the very detailed and story rich dialogue options. Each Scene has its own dialogues and all of them are represented as a tree. I created a Tree class and a Node class that could hold all the data we needed for our conversations. For the 2 classes I defined all the basic functions we needed and some more specific ones fit to our gameplay (for example see ‘getNextDisplayOptions’ that returns all options the player can currently choose from.
In the ‘ConversationManager.cs’ the game logic of the conversation gets handled and the trees themselves are being defined. Some small changes to the trees are also being made dynamically during the game in reaction to how the user plays. The conversation changes depending on what Items the player has in its inventory. Because we made the classes it was very to add this new functionality and have the different data structures work interconnected inside the gameplay.

Graph
We used the Graph for the sole purpose of the Game Map. Where-in we created a “transitioning scene”, that you automatically go to every time you enter a new door. This transitioning scene, based on the “current location” (Current Node), checks for the edges it has connected and

displays the next locations accordingly. It can be considered that the transitioning scene is a visualization of the edges of the ***current location/node.***

Collaboration Notes

Jeldrik worked on the Tree data structure: developing the conversation manager, and all of the conversation nodes and trees.
Loai worked on the Inventory list as well as the game map graph.
Both of us collaborated on the creation of the box controller we made with Arduino; more specifically: Jeldrik worked on the rotary knob and the card insert reader; while Loai worked on the joystick movement (with support from Jeldrik) and the fire sensor module.

README for Unity Setup

- Game must always start in the Forest scene in order to work.
- COM might need changing to the respective USB port your laptop has.
- All Arduino related scripts can be found in Project Folder > Assets > Scripts > Arduino