Object Descriptions

Object purpose summaries:

- **GameLocations object:** Keeps track of the locations of the player, the Wumpus, and the hazards. Also handles Wumpus movements unless there's a separate Wumpus object.
- **Player object:** Keeps track of the player's inventory and score. (This could be done by the same person doing the GameLocations object).
- **CaveConnections object:** Keeps track of which rooms in the cave are connected to which other rooms via tunnels/doorways.
- **CaveBuilder object:** Builds the cave complex by generating the blocks for the rooms, and creating or removing blocks to open or hide tunnels between rooms.
- **Game Control object:** Implements gameplay rules, coordinates all the other parts of the game.
- **Scoreboard / High Score object:** Calculates the player's current score, and optionally saves a list of high scores (including displaying a high score scoreboard).

Additional objects (these are only added when there are more than five students)

- **Trivia object:** Manages the trivia questions for the game (including asking questions and loading questions from a file).
- **Sound object:** Plays sounds for the game, from a selection of different themes.

The GameLocations Object

The GameLocations Object tracks the locations of all of the objects in the current game. The tasks it performs are as follows:

- Keep track of where the hazards are
- Keep track of where the Wumpus is. This includes controlling Wumpus behavior (that is, asleep, awake, moving).
- Keep track of where the player is
- Indicate which warnings to give in each room.
- Obtain hints to help the player.

The Player Object

The Player Object keeps track of the player and all information associated with the player. The tasks it performs are as follows:

- Keep track of player inventory
    - Arrows
    - Gold coins
- Keep track of how many turns the player has taken
- Compute ending score of player

The CaveConnections Object

The CaveConnections Object keeps track of the data that describes the "connectivity information" for adjacent rooms. Adjacent rooms may either be connected by way of a tunnel, or not connected at all (separated by a wall). The tasks it performs are as follows:

- Read and parse cave data from a file – implementing 5 different files
- Track how many rooms are in the cave
- Keeps an internal data representation of the cave sufficient for representing connections for each room in the cave
- Exposes appropriate methods and/or attributes for other objects to query these connections for purpose of building the cave, tunnels, and determining which rooms are adjacent.

The CaveBuilder Object

The CaveBuilder Object "draws" the cave complex by creating the blocks and mobs of the room, tunnels, and hazards. The tasks the CaveBuilder Object performs are as follows:

- Create the rooms, including the hexagonal form of the room, with each side containing either a tunnel to an adjacent room or a wall blocking access to an adjacent room.
- Create the pits (they may be hidden and then created the moment the player walks into them), the bats, and the Wumpus as needed.
- Build "warnings" as visible cues in the rooms, such as skulk on the walls, different blocks on the floor, etc. These indicate the presence of a hazard or

Wumpus in an adjacent room. (Note that the GameLocations object is responsible for deciding which warnings belong in the room; CaveBuilder only displays what it’s told to show.)

The Game Control Object

The Game Control Object coordinates all of the other pieces of the game:

- The Game Control Object starts / restarts the game, including when the player dies/respawns.
- The Game Control Object implements game logic: communicating with other objects to handle what happens when the player moves to a new room, shoots an arrow, runs out of arrows, etc.
- The Game Control Object interacts with every other object.

The Trivia Management Object

This component of Hunt the Wumpus is used to resolve conflicts and purchases during gameplay and provide secrets or hints that help the user to progress through the game. The user will interact with it, when it is called by other game objects for the following:

- Purchasing additional arrows: 2 out of 3 trivia questions must be answered correctly.
- Purchasing a secret: 2 of 3 correct answers.
- Saving from a bottomless pit: 2 of 3 correct answers.
- Escaping the Wumpus: 3 out of 5 correct answers.

The Trivia Management System will handle asking questions and getting the user response itself.
Judges at Finals will accept projects substituting a different mini-game instead of trivia. However, substitution is allowed ONLY after a full trivia implementation is in place. **You must build trivia first!** If a different mini-game of similar scope is developed instead of trivia, the mini-game object must still be responsible for displaying the mini-game state and handling user input. Also, as defined in the spec there are cases where the user will receive trivia answers. If a different mini-game is developed, the mini-game object is still responsible for providing hints or other advantages in playing the mini-game to replace trivia answers.

The Scoreboard / High Score Object

This component of Hunt the Wumpus is constantly re-calculating the player's current score as they play the game, and shows the player's current score, and optionally manages scores achieved by players that have won the game. The user will interact with this component in two ways:

- Storing a new high score
- Viewing existing high scores

The default high score data for the game should be the names of all of the participants in this project, with zero scores and arbitrary cave names.
w
The high score data will be exposed in a couple of ways: the user will be able to bring up the high score table through the game's menu system, and the high score table will automatically be displayed after a player finishes a game.

At no time will there be more than 10 scores tracked by the *High Score System*. If a new score is submitted, that is good enough to be included in an already-full high score table, the new score will cause the lowest other score to be removed automatically.

The Sound Object

The Sound object is responsible for playing sounds upon requests from the Game Control object. The tasks it performs are as follows:

- Reads list of sound files from its configuration file.
- Sounds files are grouped into sound themes (alternative sets of sounds).
- The Game Control object gets a list of available sound themes from the Sound object and selects (randomly or based on user’s input) an active scheme.
- The Game Control object calls the Sound object to play a particular game sound (player moves, player shoots an arrow, Wumpus moves, trivia pops up, win, lose, etc.) from the selected theme.
- The Sound object opens and plays the sound file that contains the requested sound.
- There should be at least two sound themes.