Stage 4: Player

Now that we have a world with items, it makes sense that a player should be able to pick up items and carry them around. Our next goal includes:

- The player can pick up items from the current room.
- The player can carry any number of items, but only up to a maximum weight.
- Some items cannot be picked up. An item that is too heavy (more than the player can carry) cannot be picked up.
- The player can drop items in the current room.

As we think about how to add a player to our project, we quickly notice that the Game class is the only place where it can fit in. However it is not a solution that is well-designed. After carefully thinking about this, we realize that we should add a Player class to our project that will manage all the data about a player and then we can just create a Player object at the start of the game. Here are the steps you need to take:

1. Refactor your project to introduce a separate Player class. Look at the field that you have in the Game class. Which are specific to a player? At the very least the player's current room (and therefore the previous room) is player specific. It is also likely that the score is specific to the player. Carefully move these fields into the Player class, provide appropriate methods for accessing and manipulating these fields. Test to be sure that your project is working as it was before you move on. Once it is working, check your changes into Github.
2. Allow a player to pick up a single item. This requires two commands: TAKE and DROP.
   (a) Extend your implementation to allow the player to carry any number of items. You will need to use a collection for storing items that the player is carrying.
   (b) Add a restriction that limits the amount of items that a player can carry. Do this by introducing a variable that stores the maximum weight that can be carried by a player. Your game should distinguish between items that are too heavy to take at all and items that you can't take because you are carrying too much already.
   (c) Implement a command that prints out items currently carried and their total weight.
   (d) Add an Easter egg to your game that can increase the weight the player can carry. For example, add a special cookie that if the player finds and eats it, it increased the weight the player can carry.
   (e) Be sure to thoroughly test your project and submit your changes to Github.
3. In addition to being limited on how much weight your Player carry, your player should also be limited in some other way (e.g., health, energy, etc)
   (a) Add appropriate field(s) to the Player class to account for his current level (of health, of energy, etc.)

   (b) Update the attributes for each turn. Each turn is represented by an iteration of the loop in the processCommand method in Game.

   (c) Be sure to thoroughly test your project and submit your changes to Github.

4. Update your UNLOCK command that it checks the players inventory for the appropriate key for unlocking the door. Recall that the door stores a reference to the key that is needed to unlock it.

5. Update your design document appropriately for this stage.

As usual, continue to improve your game by working on the game play. Each room's description(s) need to give the player enough information to be able to make progress through the game. Make it easy for the player to read by using escape characters (`'\t'`, `'\n'`, etc.) in your descriptions. This is critical for making your game enjoyable to play.

In order to continue you must:

- Refactor your project to have a separate Player class
- Implement TAKE, DROP and INVENTORY (or equivalent)
- Update your design document to include these commands and any additional items added to your world.

Penalties (will not prevent you from moving on, but will lower your overall score):

- Not limiting your player's ability to carry objects by using a weight limitation.

When you are ready upload your project archive (from Github) to Moodle for review. Your submission will be reviewed within 48 hours (often sooner). The review process will ensure that you have completed this stage before allowing you to progress to the next stage.