CS 120: Object-Oriented Software Development(Spring 2022)

Project Feature: Non-Player Characters

Due Date: **Wednesday, May 4, 2022**

Non-Player Characters (NPCs) are found in most games and include all the people that your player will interact with in the game (but not monsters with whom the only interaction is combat). NPCs include people and elements of your game (e.g. computers, cell phones) with which your player will interact by having a conversation.

Our conversations take the form of *dialogue trees* that will be familiar to anyone who has played an adventure or role-playing game. The NPC says something and the PC is given a list of possible responses to choose from. Based on what they say, the NPC says something else. The conversation continues like this until the NPC ends the conversation.

We build our dialogue trees by using a string to track the PC's responses so far, called the *key*. The initial key is always "hi", and later keys come from concatenating the previous key to the text typed by the player. See the example conversation below, with the key shown after each response:

key: hi
Dr. Sliva: Welcome to the Conversation Class. What can I help you with?
    a) I want to have a conversation.
    b) I don't understand which class should store my conversation.
    c) I have no idea what is going on.
Enter the letter of your response: a
key: hia
Dr. Sliva: So you want to have a conversation, huh? Do you want to talk about why the sky is blue?
    a) Not really.
    b) If you really want to.
Enter the letter of your response: b
key: hiab
Dr. Sliva: Did you know that the sky is blue because molecules in the air scatter blue light from the sun more than they scatter red light?
    a) No I didn't.
    b) Yes I did.
Enter the letter of your response: b
key: hiabb
Dr. Sliva: Wow, you are smart, aren't you. **Goodbye**.
The conversation ended with the key: hiabb
You can use the final key to cause something to happen in your game if the right thread was followed.

Every conversation has a special word or phrase called the *signoff* that will cause the conversation to end if the NPC says it. In the example above, that word was "Goodbye".

1 Design

For each NPC in your game, you will need to add the following into your design document:

- The name of the NPC.
- Where in the game the NPC is found.
- The responses the NPC can give:
  - The key that causes the response.
  - The text of the response.
  - The replies that the PC can give afterward.
- Any actions that can occur as a result of the conversation, such as the NPC giving you an item or attacking you. If these actions only occur for some paths through the tree, specify which keys lead to them.

The final part of our design is to add GWTs for a new command: TALK. This command expects the name of an NPC that is in the current room, and initiates a conversation with that NPC.

2 Implementation

1. Read through the `Conversation` class that we added to your project. You should not need to modify this class, but do need to figure out how to use it.
2. Create a `Character` class with the following fields:
   (a) A name
   (b) A `Conversation` object which will contain a map of the responses.
   (c) An inventory (if appropriate)
3. Add your NPCs to your `World` class. This should be a separate method that:
   (a) Sets up a conversation object for a character
   (b) Instantiates `Character` objects
   (c) Adds any items a character may hold (if appropriate)
4. Add characters to the `Room` where they can be found. Be sure that the player is told that there is a character in a room when appropriate. You can do this either by modifying the description of the room to include the fact that a character is always there, or you can implement a character string (like your exit string or item string) which prints whenever a character is in the room the player enters.
5. Implement and test the TALK command.