ICS 31, Summer Session 10-wk, 2017

Lab 4

You will write a very simple text adventure game. When the game is started, a "$" prompt is printed on the screen to let the player know that he/she can type in a new command. To start playing, the player will need to load a dungeon file which describes the rooms in your dungeon and how they are connected. The player begins in an initial room which is specified in the dungeon file. When the dungeon file is loaded, the description of the initial room is printed on the screen, followed by a prompt on the next line. Then the player can issue commands to move in a direction, either north, south, east, or west. When a player moves, a description of the new room that he moves into is printed on the screen and the player is prompted to enter a new move. The adventure ends when the player enters the "quit" command.

The following is a running example showing how your program should respond to commands. The text typed by the user is in bold.

```
$ loaddungeon dfile.txt
This is an open field west of a white house, with a boarded front
door.
$ north
North of House: You are facing the north side of a white house.
$ east
Behind House: You are behind the white house.
$ south
South of House: You are facing the south side of a white house.
$ quit
```

The following is a running example showing how your program should respond to commands. The text typed by the user is in bold.

Dungeon Files

The first command issued by the user when running the program must be the **`loaddungeon`** command which defines the structure of the dungeon by loading a dungeon file. The program must be able to load any properly formatted dungeon file.

Dungeon File Format

The dungeon file will be a text file containing information about each room in a specific format. Each line of the file will contain information about a single room in the dungeon. Each room is associated with the following 6 fields of information.

1. ***Room number***: This is a positive integer which uniquely identifies the room.
2. ***Description***: This is a string which is printed when the player enters the room.
3. ***North room***: This is the room number of the room immediately to the north of this room. If there is no room to the north of this room then this value is -1.
4. ***South room***: This is the room number of the room immediately to the south of this room. If there is no room to the south of this room then this value is -1.

5. ***East room***: This is the room number of the room immediately to the east of this room. If there is no room to the east of this room then this value is -1.
6. ***West room***: This is the room number of the room immediately to the west of this room. If there is no room to the west of this room then this value is -1.

All 6 fields of information about each room are contained in order on a single line in the dungeon file. Each field of information is separated by one or more spaces. The room described on the first line of the file is the initial room where the player will start after the dungeon file is loaded. Here is an example dungeon file which describes the rooms used above in the running example.

```
1 "This is an open field west of a white house, with a boarded front door." 2 -1 -1 -1
2 "North of House: You are facing the north side of a white house." -1 1 3 -1
3 "Behind House: You are behind the white house." -1 4 -1 2
4 "South of House: You are facing the south side of a white house." 3 -1 -1 -1
```

Commands

Your program should accept the following commands.

1. **loaddungeon**: This command loads the information contained in a dungeon file. The command takes one argument, the name of the dungeon file. This command should be the first command issued by the player after starting the game.

2. **north**: This command moves the player into the room to the north of the current room. If there is a room to the north then the description of the new room is printed. If there is no room to the north then the message, "You can't go there." is printed to the screen.

3. **south**: This command moves the player into the room to the south of the current room. If there is a room to the south then the description of the new room is printed. If there is no room to the south then the message, "You can't go there." is printed to the screen.

4. **east**: This command moves the player into the room to the east of the current room. If there is a room to the east then the description of the new room is printed. If there is no room to the east then the message, "You can't go there." is printed to the screen.

5. **west**: This command moves the player into the room to the west of the current room. If there is a room to the west then the description of the new room is printed. If there is no room to the west then the message, "You can't go there." is printed to the screen.

6. **quit**: This command should end the program

Additional Requirements

- Be sure to define a function called `StartDungeon()` which starts the execution of your code. Your program should call `StartDungeon()` to begin running your code.
- Be sure to include comments describing every function that you write. For each function you should have a one-line docstring describing the function, and a comment before the function definition describing the inputs and outputs of the function, and their types.

- Create a namedtuple to describe each room.
- Store all of the room namedtuples in a dictionary and use the room numbers as the keys.