Project 2: Ogopogo - Text Monster Game

Using Python, students will use casting, Boolean expressions, lists and while loops to create a text-based adventure game!

Overview

Monster stories are shared in many cultures around campfires and in the wilderness. Ogopogo is Canada's legendary Lochness monster [ogopogo] residing in the waters of scenic Okanagan region in British Columbia. This game takes place in three connected bodies of water (Okanagan Lake, Skaha Lake, and Vaseaux Lake). The user has to traverse the lakes in search of the prize. Along the way they collect items and fight monsters. On each move the user has seven possible commands: `left`, `right`, `up`, `down`, `grab`, `fight`, `help`. If the input is invalid (not one of these commands,) the game will let the user know. Otherwise, the game will execute the user's command. The goal of the game is to collect the prize guarded by the boss monster.

Details

Behavior

- The game has three lakes. Each lake is made up of four shore areas, arranged along the north, south, east and west. A shore can contain: a bouncy castle, a kayak, a paddle board, a beach chair, or nothing.
- At the start of the game, the user is placed on one of the shores.
- The user can try to move to an adjacent shore. If there is no shore in that direction, the game should report this. The user can also move from one lake, to another, if the lake contains a canal.
- The game prints out the contents of the current lake after every command.
- The user can grab swords or magic stones if they land on a shore that has them. The sword or stones are no longer on that shore once grabbed.
- Monsters guard some shores. The user can use a sword to defeat a monster using the `fight` command. The sword and monster disappear after fighting. If they have no sword, the user can exit in the direction from which they came. If the user fights without a sword, they will be defeated and the game will end. If they try to walk past a monster, they will be killed and the game will end.
- A sword and magic stones are required to defeat the boss monster.

Implementation Details

- The game should be implemented using lists.

- Use a list to keep track of the user's items. At the beginning of the game it should be empty. A maximum of three items can be held at once.
- There should be three regular monsters placed throughout the game. These require a sword to defeat.
- There should be a boss monster in the room just before the room that contains the prize. This monster requires magic stones and a sword to defeat.
- The user can only go up/down if there is a canal.
- The program should not allow the user to run past a monster, past bounds of the game.
- The game is won when the player grabs the prize.
- The game is lost if the user fights a monster without a sword, fights the boss monster without a sword and stones, or tries to move past a monster.

Design Considerations

Game Board

The game board is the basis of the game. The following is a way to think of a smaller game board as a set of three lists: one for each lake (north, south, east, west)

```
lake_1 = ['nothing', 'canal', 'nothing', 'nothing']
lake_2 = ['canal', 'canal, 'nothing', 'nothing']
lake_3 = ['canal', 'nothing', 'nothing', 'prize']
```

The above code has each lake being its own lists. Feel free to use a different implementation, but this should work for our purposes.

User Position

It will be useful to keep track of the user's position through a variable.

```
user_shore = 0
user_lake = lake_1
```

This would put the user at the position of the first shore (north) of the first lake.

Validating User Input

You will need to check the input of the user to make sure it is valid for the current game state:

```
if user_input == "down":
    current_shore = user_lake[user_shore]
    if current_shore != "canal":
        print("Can't go there; there are no canal.")
```

Grading

Scheme/Rubric

Functional Correctness (Behavior)	
Game has three lakes	5
User can move `left` or `right`, but not beyond theshores	10
User can only move `up` or `down` at an appropriate canals	5
`Grab` adds an item to the users collected items	5
User can only collect 3 items	2
`Help` lists all possible commands	2
Monsters either disappear if user has a sword or defeat the user	5
A sword can only be used once and then it disappears	6
Boss monster needs sword and magic stones to be defeated	5
Prize is blocked by boss monster	5
Sub total	50
Technical Correctness	
Correctly use of lists	15
Correctly appends items to list of users collected items	15
Correctly uses if statements to check items in user's possession	15
Correctly using `or` statements and `and` statements	15
Sub total	60
Total	110

Extra Credit

- Add the command `run`, which allows a player to run past a monster instead of fighting. This should work 40% of the time. (Hint: Research the random library.)
- Implement the board using nested lists (each item of the list is a list.)

[ogopogo]https://en.wikipedia.org/wiki/Ogopogo#References © 2019 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About