Overview

> This is an *Alternative Homework:* **Focus on completing the main homework assignment first.** Consider attempting this one only if you have time. See final section for more information on alternative homework assignments.

Homework 3 involves writing a text-only game. This doesn't directly relate to web development, but is a great exercise to improve your general programming chops. The in-class activities already contains something like this. See `3.2-functions/solutions/2_functions_game.py` and `3.2-functions/solutions/5_methods_bonus.py`.

1 Requirements

- **Must have a Python code file named "start.py" that starts the game**
- Like other text adventure games, the game must consist of moving from location to location, each one described with text.
- **Game requirements**
  - **Locations:** There must be at least *10 locations* or rooms you can move to using "north", "south", "east", and "west"[1]
  - **Items:** There must be *at least* two key items that can be found, picked up, and then used to unlock sequential doors in order to win.[2]
  - **Text:** Each room must be briefly described in text.[3]
  - **Winning condition:** There must be a way for the user to "win" at the game, such as by getting to a final room or location.
  - **User input:** The user must be be prompted to move about in the game, or perform actions such as picking up the key. If they user types something incorrect, it should ask the user again instead of crashing or doing something unexpected.
  - **Changing element:** To receive full credit, you must implement one "changing" element to make the world seem more "alive":
    - Another character that moves from location to location[4]
    - An event that happens after a certain number of steps no matter where the player is.[5]
    - A door or room that changes based on some sort of in-game weather or time[6]
- **Code requirements**
  - **Data-types:** To receive full credit, you must store inventory and rooms in dictionaries. Adding a new room has to be as easy as adding to a list or a dictionary.[7]
  - **Functions:** You must create and use at least 3 functions which between them, use at least one return statement and at least one argument. All Python code must be in functions – no "loose" code[8]
  - **Statements:** Your game must use at least one of each of the different Python branching and looping statements: `while`, `for`, `if`, `else`, `elif`. Additionally, you must use at least one `and` and at least one `or` to combine your conditions.

[1] Not all locations need to have all directions. Also, special instructions like "climb ladder" are fine too.

[2] Any sort of equivalent puzzle is acceptable, also, as long as it is some sort of "inventory item" that can be picked up, and only with this inventory item can the player proceed.

[3] You should not copy any text from in-class activities.

[4] This could either be a villanous monster, or a person that can be talked to. Their path could be random, or fixed.

[5] This could be a phone call like in the activity, or a timer count-down.

[6] Every time you move could advance the in-game "time", so there could be a door that only opens at midnight, for example.

[7] That is, you cannot only rely on functions & print statements for the different rooms or locations like in the solution to the `3.2-functions` activity. You must store the locations using Python data-structures which in turn get `print`ed out.

[8] With the exception of the initial invocation of the "main()" function, see below. Also, using classes is okay.

2 Submission

- Must use a brand-new repo from all previous homework[9]
- Repo must contain a `README.md` document, and a `SOLUTION.txt` document. The former should describe the game and how to play, the latter should contain the sequence of commands necessary to win at the game.

3 Research

Text adventure games, also known as "interactive fiction", is a genre of text-only video games that reached its peak popularity in the 70s and 80s. If you aren't familiar with them, a good idea is to spend a few minutes[10] trying existing IF games to get an idea of an "example solution" of what you should try building, except that yours should be much shorter in length.

- **Online:** You can play several 80s classic text adventure games in the browser: zorkonline.net
- **macOS:** A text adventure called Dunnet is available by running: `emacs -batch -l dunnet`
- **Linux:** Running `sudo apt install bsd-games` will install one of the oldest video games ever written, called ADVENTURE and written in 1975. Once installed, you can run `adventure` to try it out. (also `battlestar`)

4 Tips

4.1 General tips

- Create an interactive fiction game with at least several rooms you can navigate between by typing in "north", "south", "east", or "west" at a prompt.
- To avoid taking too much time thinking of a theme or or game-text, consider just mapping a generic hotel.[11]
- Be careful about getting side-tracked! You'll be graded on meeting the code requirements, *not* on an entertaining game or well-written story.[12]
- Create and use "helper" functions. Helper functions are ones that perform useful tasks to assist you in writing the game.[13]

4.2 Draw it first

- Put into use all the tools you learned for coding: Pseudocode, state diagrams, class diagrams, and more.
- Diagram a map on paper of the world you want to make.

[9] Every alternative homework should have a fresh repo each time. This is to further flesh out your GitHub profile.
[10] Only a few minutes... remember, you need to do the homework, so be careful not to get sucked in!
[11] There can be a long hallway with many rooms to search, one of which will cause the player to win, although the player must first find a necessary key.
[12] While you won't be docked for writing a lot or investing time in an interesting game, it definitely will not make grading any easier, and may make code review harder. Focus on good code, not good prose!
[13] For example, in the activities, we wrote functions that kept on asking for an answer until the user supplies a valid answer.

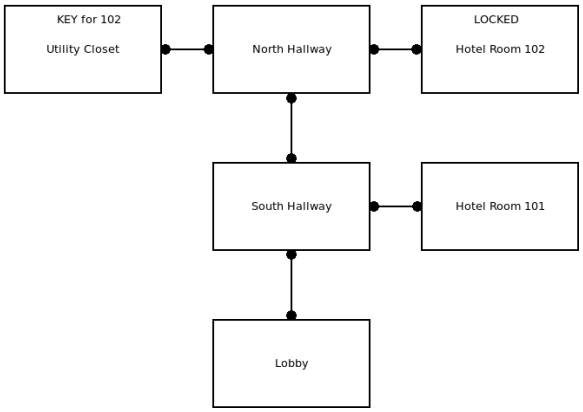

Figure 1: Map for a 6 room "Hotel" themed adventure game, with one key and one locked door

4.3 Using a "main" function

Your scripts initial code must be in a "main" function. The main function must be written and invoked as follows.

```
def main():
    bedroom() # as an example, the first room is the "bedroom"

# And at the bottom of the same file:
if __name__ == "__main__":
    main()
```

This is "best practices" for writing Python. This will cause `main()` to be invoked when your file is running as a script.[14]

[14]The reason this is best practice is that it also allows the code to be "imported" without main being run. Next week we'll go over in detail what "import" means.

Appendix: Alternative Homework

Unlike regular homework assignments, "Alternative Homework" assignments do not contribute to a cumulative solo project, but instead are one-off assignments.

They are typically as hard as the solo project homework.

Why do alternative homework?

- Be better prepared for the interview process. Often you will have "take-home" interviews that involve making similar one-off projects from scratch.
- Extra reinforcement of the concepts of each week.
- Flesh out your GitHub profile with more repos. You want at least 10+ repos, so doing these extra homework will help with this sort of volume.
- **For students seeking career change, it is recommended you attempt to fully complete both homework assignments.**

Grading

Alternate homework are optional, but you may still turn them in to receive feedback.

NOTE: You may request to have the alternative homework graded and apply to your grade instead of the solo project homework, but it is recommended you attempt to fully complete the solo project homework first.