This week's activity, which is also the final one for this class, will be on the final data structure we have come across: the Graph. We have recently used a Graph to represent relationship dynamics, but for this activity we are going to use one to represent a map. Which we are going to use to make an adventure game.

The Premise

The basic idea of old-style adventure games (notable examples including The Hitchhiker's Guide to the Galaxy, Zork, Avalon, and more) is to provide the player an initial narrative prompt and then tell them what actions they may perform at their current location. Navigation to other areas is key to these kinds of games, as well as (often obtuse) puzzles that require collecting various items and using them in novel ways to advance the story.

Note: Do not feel trapped by the traditionally RPG/fantasy theming of many classic adventure games. You could make a game that emulates an episode of Friends if you really wanted to! Or a Sherlock/Poirot/Mulberry style mystery, maybe a quest to find the coat you forgot in a classroom yesterday that takes you to the seedy underbelly of the university to find the lost-and-found box… whatever you want, really.

Try to think of this less as a video game and more as an interactive story, and stories can be as varied as the human experience!

The Program Details – How we're using Graphs

Each Vertice of our Graph is going to be an area of your adventure game. They can represent a room in a building, a town the player can travel to, a particular piece of furniture in a room; you are free to use whatever context/environment you want for your adventure game.

Each Edge of our Graph is going to have a cost associated to it, but this cost is not going to just be a 'distance' or 'payment'; we are going to use the idea of a weighted graph to denote the item requirements that the player needs to have to be allowed to traverse to certain areas.

The simplest example would be to imagine we have two rooms A and B. The edge between A and B will narratively represent a locked door, so the player will have to perform some action in room A to acquire a key. When trying to move from room A to room B, your program would check if they have the key item, and only let them through if they do.

Now, we can abstract this concept from just being an item to being, say, an action they have performed: check if the player has 'Has talked to Phoebe' in their event log before they can leave the apartment, for example. We could even re-integrate the usual idea of a weighted graph and have a toll that does deduct from the player's wallet on moving between certain vertices.

The feedback a player receives when they cannot go to a location can be a good opportunity to provide direction to the user on how to proceed in your game. For example, telling them ‘you do not have the key’ or ‘Phoebe asked you to speak with them, remember?’

This is all to say that you should not feel constrained by the basic ideas of what a Graph is when planning how to make your program; you understand what a Graph needs to be, now expand on that premise to make it customized to the adventure game you want to make. Add layers to the information each Vertice and Edge holds and tie them into the mechanics of what the player will have to do to succeed at your game.

We spend time in class talking about specifically formatted weighted graphs because those work well with certain algorithms or traversal strategies, but we are using graphs very differently than those purposes for this assignment so feel free of the bonds of standard use cases. Get abstract and be creative!

The Program Details - Okay but what do I actually do for a good grade?

You are expected to create a short narrative experience that uses at least 6 distinct areas to allow the player to navigate your story and do things in each area. At least 3 of the edges between areas should have some cost associated to them that requires the player to acquire an item or do something which will allow them to pass through.

Behaviorally, we would expect that the player would be prompted to use commands like ‘take item_name’ or ‘talk person_name’ to perform actions. The player should be told upon entering an area what things or people there are to interact with. (Again, this could be developed into having a ‘search’ command to reveal things if that is relevant to your design ideas). Basically, what verbs can the player perform at the locations of your game? What do you need to tell them so that they can use those verbs?

We expect to see a novel implementation of a Graph for this purpose, especially as the areas each Vertice represents will likely have some amount of complexity to represent the things the player can do or obtain at that location, and the Edges will potentially have a dynamic system of deciding if the player can travel along them or not.

You do not need to adhere to the GraphADT you were given in class, but you should still be considerate of good programming standards in your implementation.

You should provide documentation that outlines the design ideas you have for this program; this will be a good opportunity to decide on what kind of story you want to tell, and how you will represent those ideas in your code. The documentation should be focused first on the general design ideas of your story (map sketches, what each location has, etc.) and then how you will translate those ideas to your program (a UML diagram is expected).

What to submit:

- Documentation outlining your ideas for the narrative experience.
    - Should include some planning of the program architecture based on what you want to do with the story.
- The code that you wrote to implement the narrative experience you had designed.
    - Should be commented well.
    - Should use a Graph data structure.
    - Usual expectations on code quality
        - This includes architectural considerations.

You will be assessed based on the quality of your code, the quality of your documentation, and the user experience when engaging with the program that you wrote. A considerable portion of the code quality will be based on your usage and implementation of Graphs.

Due: December 6'th, 2023 @ 9 PM

Late submissions are not accepted without prior discussion to extend the deadline.

I will not answer questions about the assignment asked after December 5'th @ 9 PM.

This activity is expected to be completed solo. You may always discuss course concepts with your fellow students and how to approach a problem, but your work should be your own and only reflect your own ideas. Include in your comments any websites (such as Stack Overflow) that you consulted during your work and what contributions they made.

Information gathering is a valuable skill and knowing how to find information online is important to be able to do; letting me know what information you needed to seek out allows me to know what to focus on teaching in the future.