Final Project

Goals-
Design a program to satisfy provided requirements
Use good OOP style including inheritance, polymorphism, and pointers

You will design and implement a text-based game or puzzle where the player moves through a series of rooms or compartments. They will need to gather items to achieve some purpose. The details are left to you!

Specific requirements. You will create a series of rooms or compartments or spaces for the player to move through. For simplicity I will call them spaces, as they do not need to be a physical compartment. Each space will be a class with (at least) four pointer variables that link to other spaces. Even if your structure is linear, such as a train, you will still have four pointer variables in the class. Be creative and try to use them all! You might consider secret passages or "stepping" outside the train? The space will also have appropriate data members. You must have at least 5 spaces of at least 3 different types.

NOTE: The requirement for four pointers in each room/compartment is to simplify coding. If you need more pointers for your theme, feel free to add more.

You will have a space abstract class that will have a special pure virtual function. Each type of space will have a special action. It could be the controls in the engine, or to control access to one of the doors (pointers), or it could simply turn on the lights.

You will have at least 3 derived classes for different types of spaces. To continue with the train theme you could have a passenger car, or baggage car. Or one space could have controls such as the engine.

You must have a goal for the player. Maybe it is to solve a crime (like Clue?). Maybe the aliens have invaded your spaceship and normal weapons do not work but broccoli causes them to leave! ☺ Based on your theme the player must discover the solution. You may have a random goal. One time broccoli works, next time it is a pillow.

You must have some way to keep track of which space the player is in. The player will have a container (backpack, knitting bag, or notebook) to carry "items". The container must have some limit. One or more of these items will be required as part of the solution, such as a "key" to open the locked door.

You will change the structure of your spaces. Specifically you will add at least one space and you will remove at least one space. They do not need to be the same space. To continue with the train theme say you initially have:

baggage, passenger, passenger, club, Pullman (old fashioned sleeper)

Later on, when the player goes through the train it is:

baggage, passenger, passenger, club, observation

or:

baggage, passenger, passenger, observation, Pullman

The goal may be a puzzle with clues in each space. It does not need to be a physical possession. It should not require free-form input. It is frustrating if you did not spell

something correctly. You should have a time limit to urge the player on. This does not mean a literal clock, just some way to prevent the 'game' from going on indefinitely. The player must interact with parts of the structure, and not just simply collect things. This can be throwing something at the monster, operating a light switch (or other control), opening doors, or singing to get the baby back to sleep.

Up to you. You must develop a theme for your game. Make it interesting, if not fun. BTW, notice the graders get a "best in show" award of extra credit so make it interesting and/or challenging!

To encourage you to make it interesting you can email your theme to the instructor by midnight Sunday, 31 July. I will post them, stripped of any names, and the class can vote on which one they find the most creative, best or whatever criteria you want to use. The TAs will get to vote too. The winner will get some extra credit! ☺

What to submit in your zipfile.

- Your makefile
- Source and header file for each of your classes
- Source and header file for your program code
- Your reflections document

To make it easy on your grader please provide an additional menu option. It should reveal the goal to the 'player'. If the grader does not know the goal they cannot tell if the program is working correctly. This information does not need to be elaborate. Something like, "the broccoli causes the aliens to leave", or "the murder was done by Mrs. Duck, club car, fishing pole." HINT: Always be kind to your grader! ☺

SUGGESTION- As always, use incremental development. The exact sequence may depend upon your theme. At a minimum you will need to move in the structure. Get that working and then flesh out the details. The player must be able to pick up a carry items, to the limit you impose. Do the design first! If you try to plan out your theme as you write code you will make it much more difficult.

Grading:

- programming style and documentation (10%)
- create the basic structure of 5 spaces of at least 3 different types- this includes how creative you were with the linked structure (15%) **(5%)**
- all links work as intended (10%)
- correctly add a node/space during runtime (5%)
- correctly delete a node/space during runtime (5%)
- prompt the user to start the game and execute all commands as intended (5%)
- properly implement some sequence of actions required for the player to exit (other than just moving through the spaces) (10%)
- correctly implement a time limit (5%)
- correctly implement and use the special function (10%)
- reflections document to include the design description, test plan, test results, and comments about how you resolved problems during the assignment (20%)
- "best in show" (+5%)
  the graders will have the option of giving 5 points extra credit to one student

they find had the most creative theme, interesting situation, or whatever criteria they want to use. If they feel no one they graded is worthy then they are not required to award it.

If you do not submit all files in a zip file your project will NOT be graded.
If you do not have each class in a separate source and header file your project will NOT be graded.
If your program has a segmentation fault you will lose 20 points.