**DESIGN PLAN**

The first step in my design plan was to actually take a step back and brainstorm and really cool game concept. I’m a big fan of sci-fi space movies, so I came up with an idea of a biologist on a secret mission being the sole survivor of an explosion in a spaceship and he has 90 minutes to get his gear and escape on the ships escape pod.

Once I had the story line outlined, I began the process of class hierarchy brainstorming. I knew I needed an Inventory and Space class. One to hold the container of all the objects of items in the game the user would need to interact with, and the Space class for all the rooms that need to be navigated through. I would need a main control class that could interact with both of those objects, as well as the child classes. I then realized there should be a menu class so that each room could have its own unique menu object for exploring and interacting with the room (space).

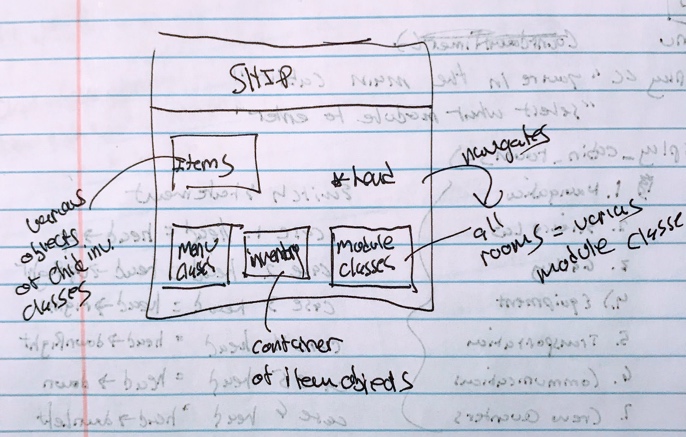
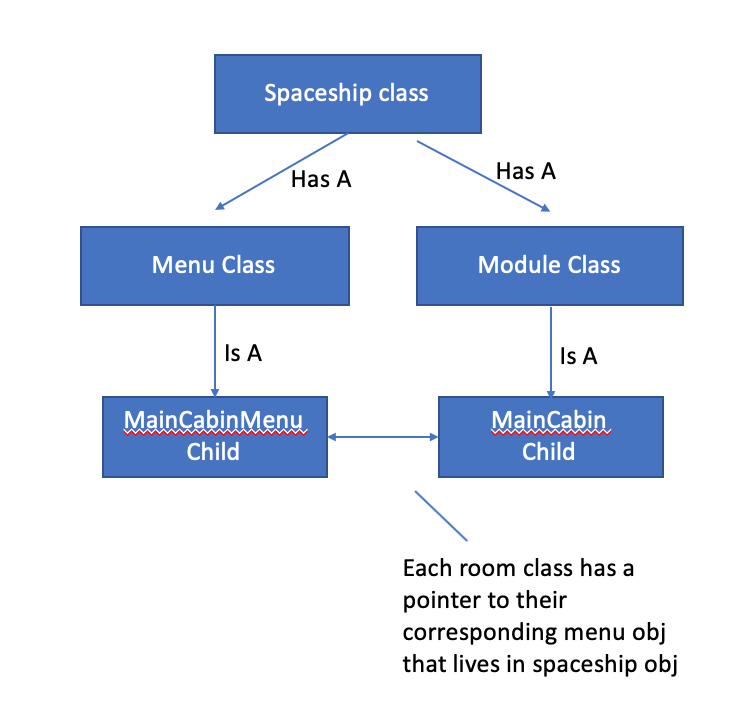
 

Figure Where objects would be instantiate

Now that I had my hierarchy - I began sketching out how the objects would work with each other in more detail.

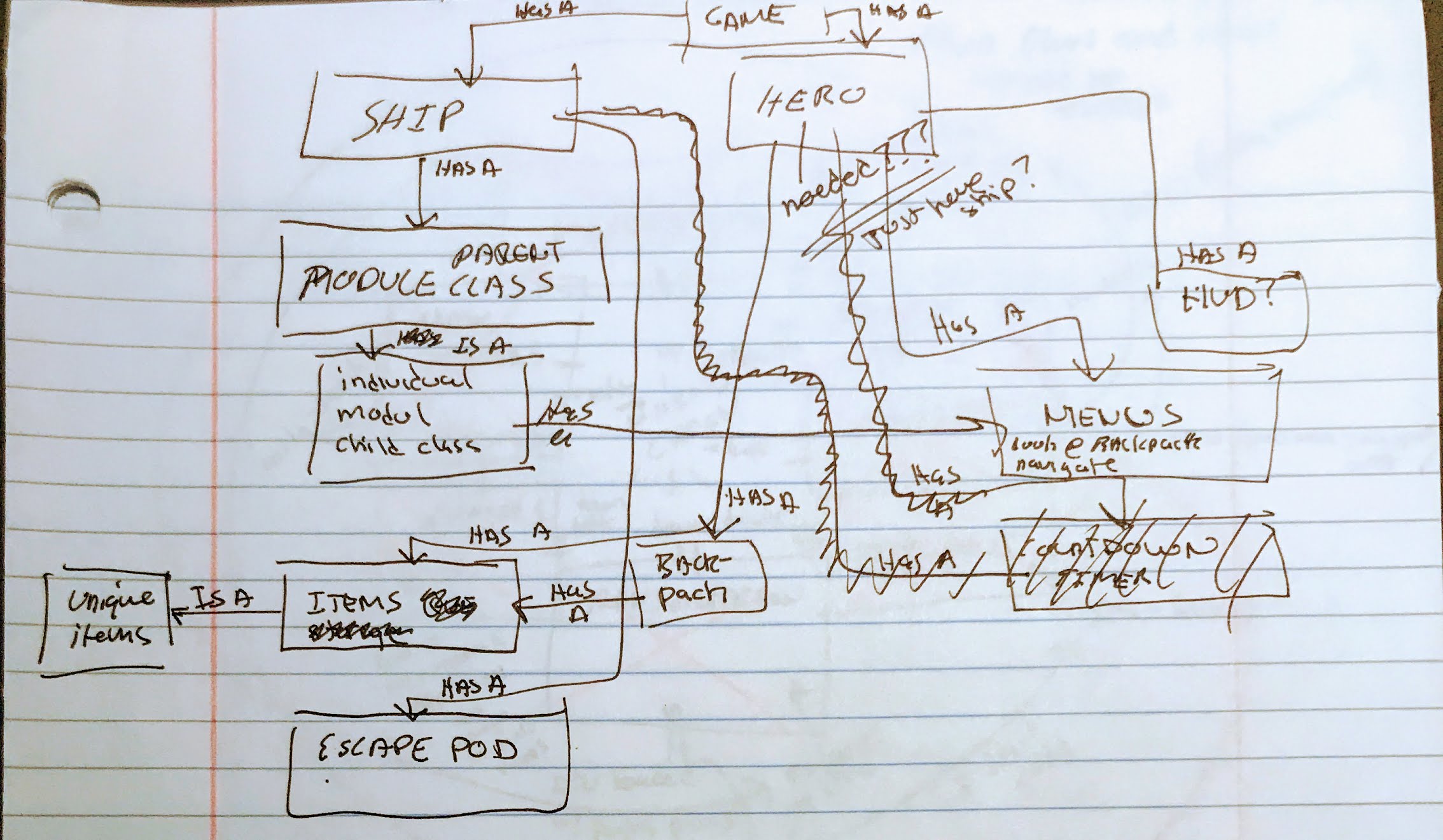


Figure Class interaction

In order to get a better grasp of the layout of the rooms in the ship, and what items would live within each room and where they need to be used in other rooms, I created a sketch of the ship and outlined how items where connected. This also became useful for designing the map in the actual program.

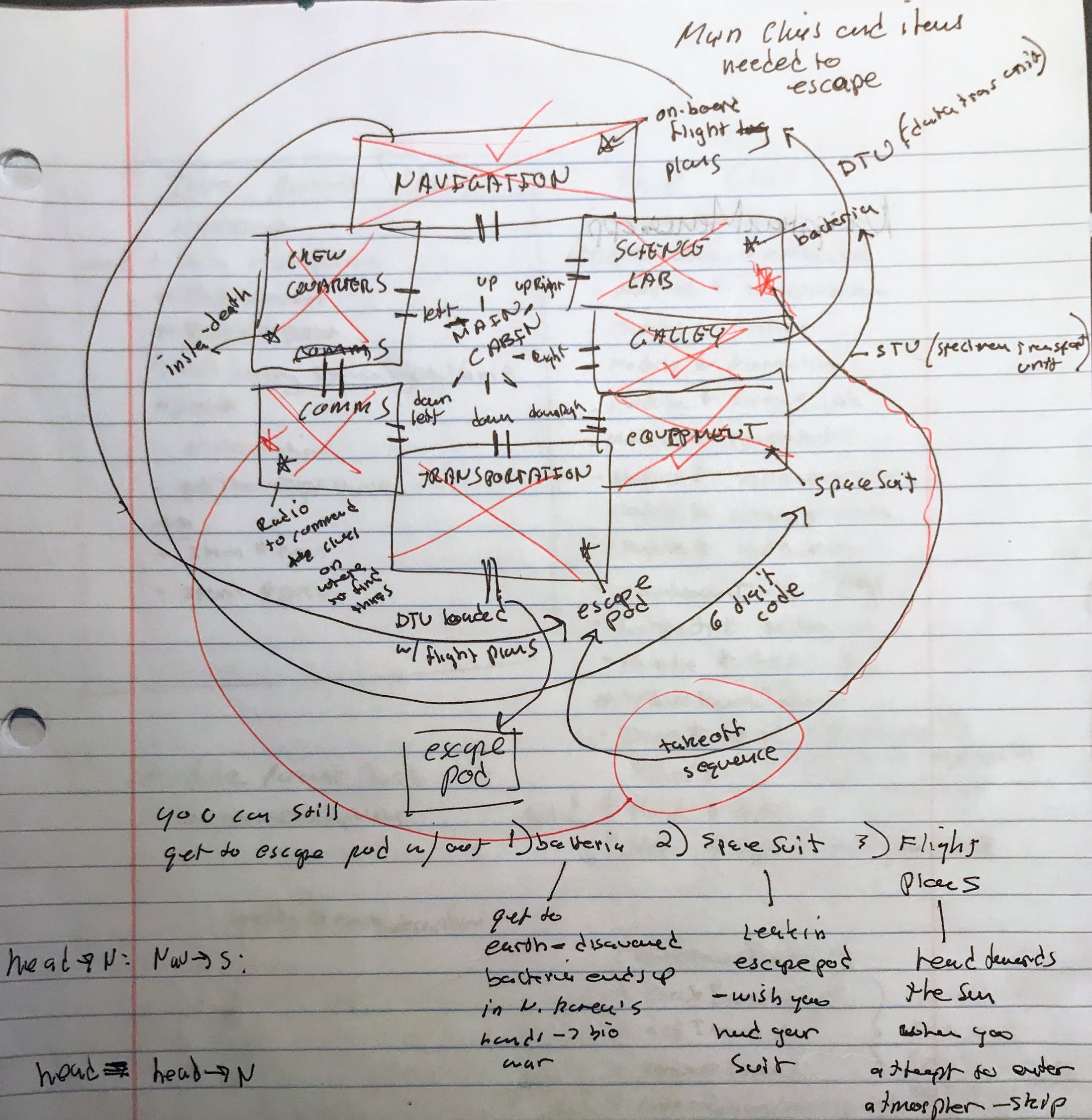


Figure Ship map and item interactions

Now that I had a good layout of classes, items and how they all work together I developed the Spaceship class’s UML – as this would be the main class that handles most of the game’s flow. This early rendition was a great starting point, but it changed many times throughout development as I identified more efficient ways to program the game.

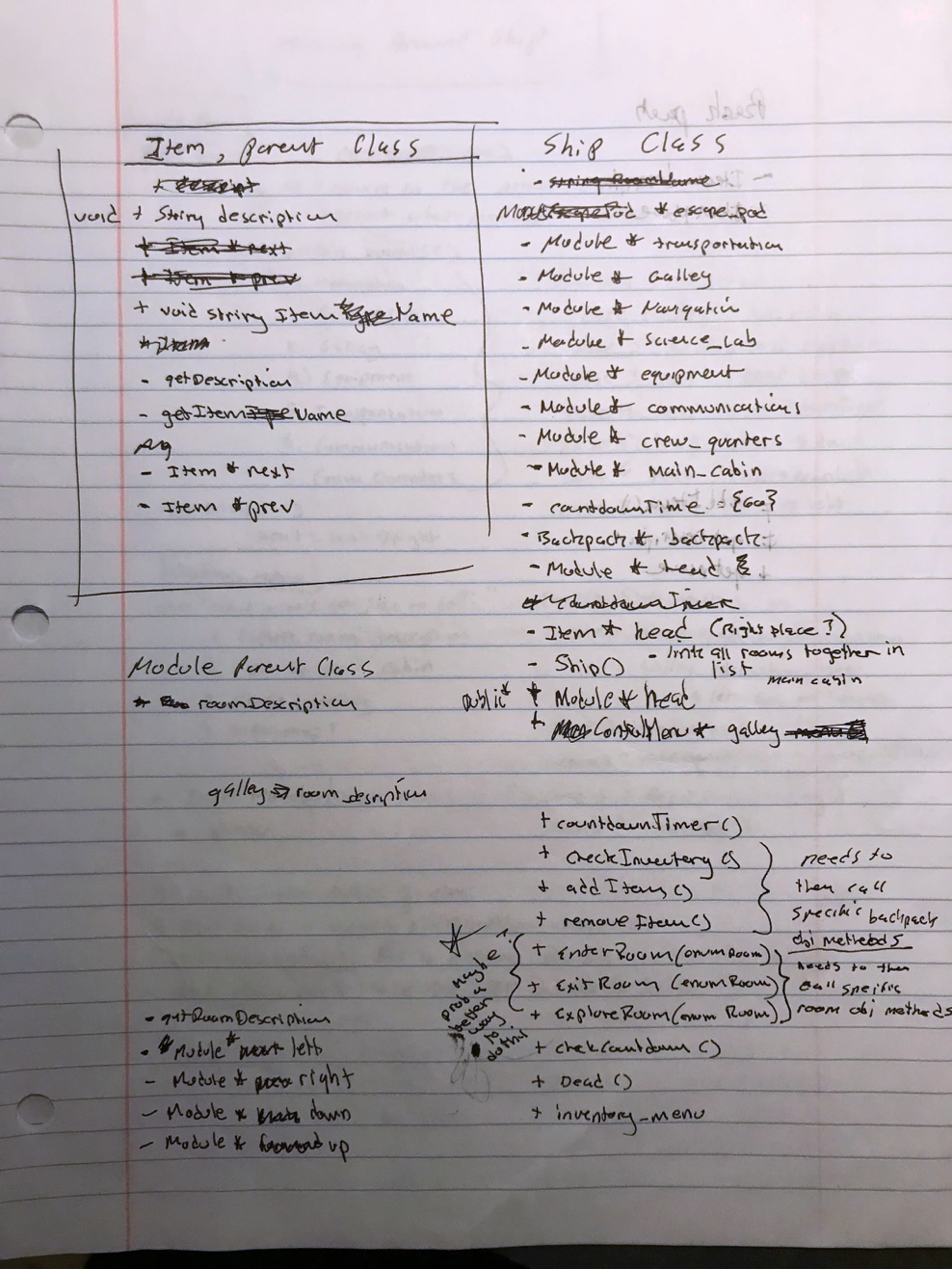
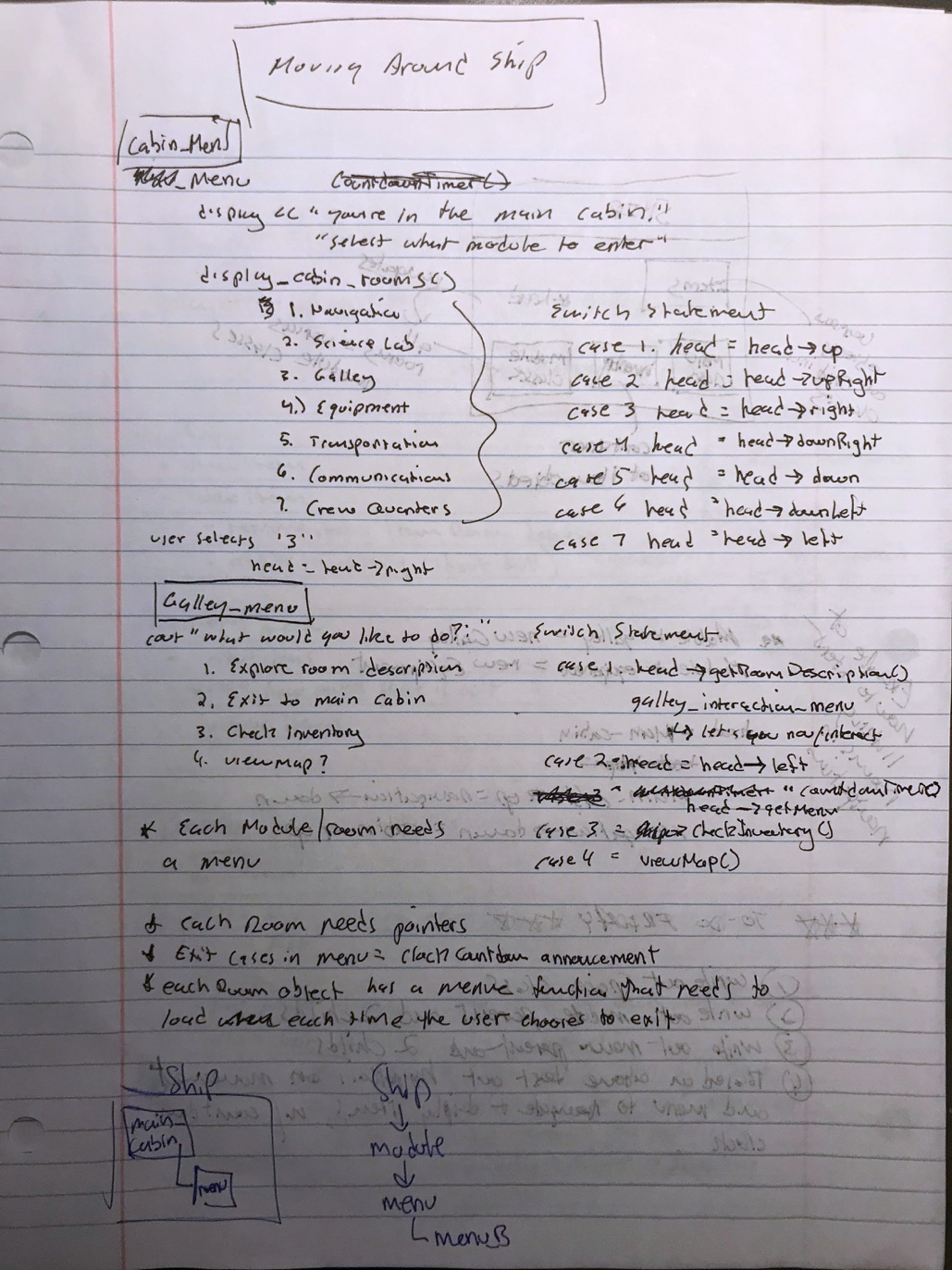


Figure Spaceship class UML

It was really helpful for me to first sketch out how the Spaceship class would handle the linking of Spaces to create a somewhat type of circular list. Each Space has 8 points of directional reference (like a compass).



Once I had tested the player moving through the ship correctly, along with all the other design work, I was ready to jump into the programming.

**TESTING PLANS AND TABLE**

For testing I implemented the following tools:

* Outline of test cases (inputs from requirements)
* dbg GNU debugging within the terminal
* Testing table to keep track of all variables (input/output)

This initial list was my outline of test cases I would need to think about when pulling together my design, as well as to test for once the program was written and compiling on Flip.

Testing for this project was extremely iterative, as I worked very modularly. For example, I ran about 20 tests to just build out a test case for moving around the ship with the different linked lists. Not until that was proven to work correctly would I work to add elements into those rooms to interact with. For each Item I created, I ran a test to make sure it was working correctly. For reach method within each class that was coded, I ran tests to make sure it worked perfectly before moving on. By the time I got to the end of the program I was in such good shape that there were only some minor testst that needed to be run – including memory leaks.

**Test Cases:**

I designed out a comprehensive testing table to track all the events above against expected and actual results – noting any solutions to errors that arose.



**Reflection:**

I had a blast coming up with the game and pulling it all together with everything we learned. I was able to draw upon previous projects and labs, as well as some of my own Google investigation, but I felt in control and confident throughout the entire process.

In the future I would still try and focus on fully designing out flowcharts for all class interactions before coding. That was something I skipped this time to dive right into the coding – as I knew this was going to be a big endeavor for me. I would also handle memory leaks as I went along with my iterative testing. This was something that I waited until the end and it did cause me to spend unnecessary extra time doing.

This really showed me I’m capable of taking requirements and using whatever tools I have to bring to life a working program that meets the deliverables of the client (professor).