DESIGN

First, I decide to write a story for that would make sense for a player to go in and out of 6 spaces. So, I wrote a paragraph as to what the objective could be, and included a story. Then, I used that narrative to write the game design below. This document includes my key, the base classes and their derived classes, and the objectives in each class. Then, I started programming by writing a base space class, and a base item class. I then built one derived space or item at a time, and tested them in my game flow. I took an interactive approach to designing my application. I focused on designing a space class with only one derived space and then used the test space to link together several instances to get the transitions sorted out. I focused on ordering my allocations and deallocations on these initial classes so that subsequent memory allocations could still be caught by the superclass destructor. Whenever I felt that a chunk of the application was working properly, I then moved on to designing a new space, including the theme of the space, and how it would challenge the player. I did sometimes try to link two spaces that were far apart together in some fashion. For example, you must travel through a linear sequence of spaces to get the astronaut suit, which is the trigger to open the planet in a far away space.

GAME DESIGN

SPACES
BASE: SPACE
DERIVED (IN ORDER):
1. OUTERSPACE_SPACE
2. STARS_SPACE_SPACE
3. BLACKHOLE_SPACE
4. ROCKETSHIP_SPACE
5. PLANET_SPACE
6. ALIEN_SPACE

BASE: ITEM

DERIVED:	KEY
1. PLAYER	&
2. ALIEN_BOX	=
3. ALIEN_SECRETS	?
4. ROCKETSHIP_KEY	$
5. ASTRONAUT_SUIT	8
6. END OF SPACE	|
7. STARS	*
8. BLACKHOLE	@
9. PLANET	o

10. ROCKETSHIP	^
11. ALIEN THANKS	>
12. FORCE FIELD	x
13. FUEL CAN	#

INVENTORY
1. ROCKETSHIP_KEY
2. ALIEN BOX
3. ALIEN SECRETS
4. ASTRONAUT SUIT
5. ALIEN THANKS

****The player is given fuel at the beginning of the game. If they run out of fuel, they lose. However, there are fuel cans scattered throughout each space which allow the player to refuel if they reach them****

FLOW THROUGH SPACES
in OUTERSPACE_SPACE 1. Must collect the ALIEN BOX to unlock STARS_SPACE. There is a force field in your way that cannot be broken until youve collected the box. There are many fake boxes to try and trick you.
in STARS_SPACE 2. Must collect ALIEN_SECRETS in your alien_box to unlock BLACKHOLE_SPACE. Must dodge the stars to get to the next space
in BLACKHOLE_SPACE 3. Must collect the ROCKETSHIP_KEY to unlock the ROCKETSHIP_SPACE. Must dodge invisible obstacles.
in ROCKETSHIP_SPACE 4. Must collect the ASTRONAUT SUIT to unlock the PLANET SPACE. Must travel through force field maze to get the suit. Once they collect the suit, they must travel back to OUTERSPACE to reach PLANET_SPACE
in PLANET_SPACE 5. Planet space is unlocked only if player has all previous items. THen, they travel into the planet
in ALIEN_SPACE 6. Must bring aliens their secrets. Then, must collect their thanks to be teleported home.

REFLECTION
Upon reflection, I'd like to implement a game where the objects in the spaces possible move, creating more difficulty for the player. Also, I encountered many memory leaks at the end of testing which I should have solved earlier. I had some problems wrangling my class hierarchies. Does it make sense for a player to be an Item? Maybe not conceptually, but it certainly makes places things on the game board easier. I also spent a lot of time making sure my code was as concise as possible, because when a function would grow too large (like the code that allows the player to move), it would become unmanageable. I think also, despite my best efforts, memory leaks were still an issue because it's very hard to understand the order and timing of

deletes in my program. Sometimes delete would be called twice, which was good in the sense that I was being very thorough about deleting memory, but bad in the sense that it resulted in undefined behavior ending in a segfault. The last part I added was extra challenges and better dialog. I think it's really important to get something working before you make it better. My game worked with all of the spaces defined and all of the item interactions placed out, but with empty rooms. I only added the challenges after everything else was ready to go.

TEST

INPUT	**TEST CASE**	**TEST FUNCTION**	**EXPECTED OUTCOME**	**OUTCOME**
1	**Start game**	**Menu**	**Start game screen**	**Start game screen**
2	**instructions**	**Menu**	**Instructions screen**	**Instructions screen**
3	**credits**	**Menu**	**Credits screen**	**Credits screen**
k	**Invalid char input**	**Menu**	**Please enter a valid integer**	**Please enter a valid integer**
-1	**Invalid num input**	**Menu**	**Please enter a valid integer**	**Please enter a valid integer**
5	**Pick up alien box**	**Outerspace interact**	**Added alien box to inventory**	**Added alien box to inventory**
5	**Pick up alien secrets**	**Stars interact**	**Added alien secrets to inventory**	**Added alien secrets to inventory**
5	**Pick up rocketship key**	**Blackhole interact**	**Added rocketship key to inventory**	**Added rocketship key to inventory**
5	**Pick up astronaut suit**	**Rocketship interact**	**Added astronaut suit to inventory**	**Added astronaut suit to inventory**
5	**Pick up alien thanks**	**Alien space interact**	**Added alien thanks to inventory, end screen prints**	**Added alien thanks to inventory, end screen prints**

5	Pick up fuel can	Fuel addition	Fuel increase by 100	Fuel increase by 100
1	Left	Player movement	Player move left, fuel decremented	Player move left, fuel decremented
2	Down	Player movement	Player move right, fuel decremented	Player move right, fuel decremented
3	up	Player movement	Player move right, fuel decremented	Player move right, fuel decremented
4	right	Player movement	Player move right, fuel decremented	Player move right, fuel decremented