Brenna Smith
CS 162 – Final Project
Due 3/20/2018

Final Project – Alien: The C++ Game

Design Plan – PsuedoCode

Space Class

Private

- Pointer to space up
- Pointer to space down
- Pointer to space left
- Pointer to space right
- Name of the room

Public

- getUp() – returns up space
- setup(Space*) – sets up space
- getDown() – returns down space
- setdown(Space*) – sets up space
- getleft() – returns left space
- setleft(Space*) – sets up space
- getright() – returns right space
- setright(Space*) – sets up space
- getName()
- (virtual) enter room (makes whatever special event happen based on room type)

ItemSpace

- EnterRoom(array of items)
  - Push prompt You found an item
  - Choose random item from item array
  - Array is not full
    - Add the item
  - Else
    - Ask if they would like to remove an item
    - Use menu to ask which item
    - Replace chosen item with new item
  - Return 0 damagee ---- (All enter rooms return health effect)

AlienSpace:

- EnterRoom(array items)
  - Push prompt alien attack

If flamethrower empty or if they don't want to use it
Damage 10-15
Else
Damage 0-5

EventSpace

enterRoom(array items)
Choose prompt randomly
Drop an item randomly or
Eat an apple +3 health or
Face hugger fight or
Nothing happens or
Find pills for health +5

AlienGame

While health > 0 and timer > 0 and they haven't won yet:
Display ship map
Ask what room they want to enter
Enter room for that space type
Add result to health
Check health and timer is fine
Ask if user would like to useItem() at the end of each room

Brenna Smith
CS 162 – Final Project
Due 3/20/2018

Test Plan

Test Case	Input Values	Driver Functions	Expected Outcomes	Observed Outcome
If all directions are available to move		move()	Allows user to choose from up down left or right on menu	Allows user to choose from up down left or right on menu
If some spaces are null		move ()	Only shows available options in menu	Only shows available options in menu
User enters an item space	Moves random room	enterRoom()	Randomly selects a required or helpful item that is added to backpack	Randomly selects a required or helpful item that is added to backpack
User enters an alien space	Moves random room	EnterRoom()	Randomly selects an alien encounter to deal with	Randomly selects an alien encounter to deal with
User wants to use flamethrower	Selects yes	enterRoom()	If not empty, give user the option. If yes, deals less overall damage.	If not empty, give user the option. If yes, deals less overall damage.
User enters an item space	Moves random room	enterRoom()	Randomly selects a required or helpful item that is added to backpack	Randomly selects a required or helpful item that is added to backpack
User decides to quit	Press 2 to quit	main()	Game stops playing	Game stops playing
User enters an event space	Moves random room	enterRoom()	Randomly selects an event to happen. Can be good, bad or neutral	Randomly selects an event to happen. Can be good, bad or neutral
User uses valid item	Chooses yes when prompted	leaveRoom()	Complete effect for that item, then remove from backpack	Complete effect for that item, then remove from backpack
User uses invalid (required) item	Chooses item from menu that is required to win	leaveRoom()	Shows user an error and states why	Shows user an error and states why
User dies or runs out of time	Health < 0 or the timer <=0	leaveRoom()	User gets message and game ends	User gets message and game ends

Brenna Smith
CS 162 – Final Project
Due 3/20/2018

Reflection

I had a fun time designing and writing this program. I've always been a huge Alien fan, and for some reason it was on my mind when reading the requirements for this project. Maybe all the talk about space? Either way, I thought this would be interested game to write and to play.

Most of the aspects of gameplay came together easily as I read through the requirements. Ridley has a bag and has to get certain items to take off before the ship explodes or before the Alien kills her, just like the movie! Some stretches had to made for the sake of development, but that made it all more interesting. The spaces aren't anything special, other than they are themed well. You get items in some space, monster fights in some, or random events in others. They all kind of borrow code from each other, despite their being content. Planning the items was difficult since everything happens in the virtual functions, they all had to have the same parameters and return types if I wanted to avoid using explicit type tracking in my main program. So I decided all space types would receive the bag and return and damage done or health gained.

The one major aspect I added to this game was a 4^{th} space called the escape space. This is a special space that allows the user to win the game if they are in the spot and have all the required items. I tried to make this like any other space where it could have a randomized type, but it had it's own mechanics since it was a requirement to make it there.

I thought scope creep would be a huge issue, but I was able to break down my goals into manageable, programmable chunks. I look forward to work on more exciting projects like this in the future.