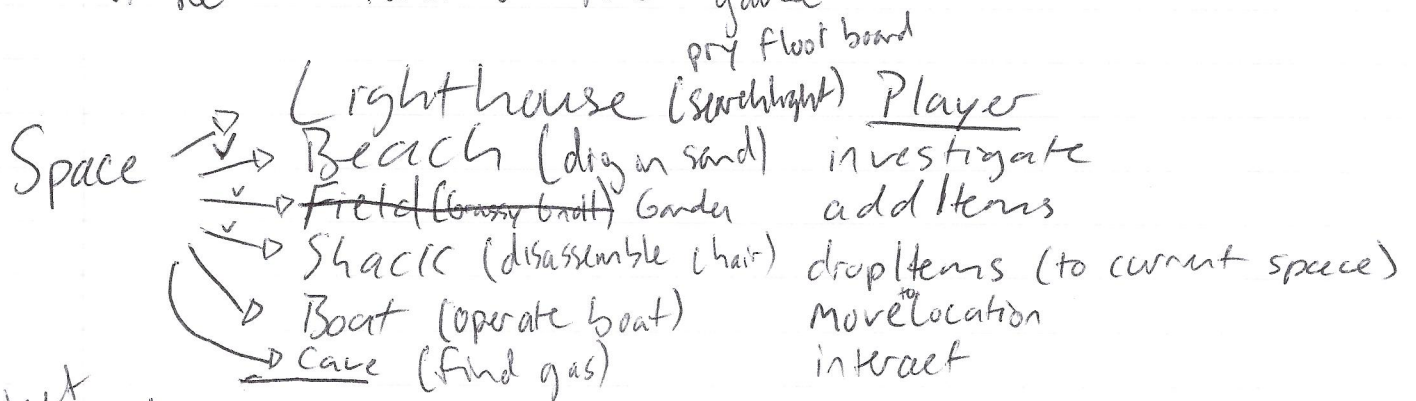


Final Project

Requirements:

- 5 different spaces of at least 3 different types (derived Space classes)
- Each space will have 4+ pointer variables that link to other spaces
- Space will be an abstract class w/function Special being a pure virtual
 - each type of space will have a special ~~function~~ ^{action}
- Programmer defined goal
- Must keep track of where the player is. Map?
- Player will have a backpack (~~array~~ ^{vector}) with a limited amount of space
 - one or more stored items must be needed for the solution to the game
- Must allow the player to interact w/environment besides picking up items
- Provide a menu option to play the game or see the rules of the game



Pocket
at start
of game

Items (strings)

- gasoline (in cave)
- map/ bandages (in shack)
- key (beach) to the lighthouse
- ~~metal pole (in field)~~ garden
- Note (boat)
- Note2 (lighthouse)
- ~~flashlight (shack)~~

Back pack storage
space = 4

Time limit will
be water/thirst

Player
Private:

vector<string> backpack; map<spaces> playerMap;
bool interaction; investigation; string droptItem;

Public:

void addItem (String newItem); // first if - else statement,
if backpack.size >= 3 cout the backpack ~~is~~ is
full, drop an item first, else the passed through
string item gets pushed back to the backpack

void displayBackpack (); // loops through and
counts the item(s) currently in the backpack
string droptItem (); // makes a call to displayBackpack
and then asks users what item they'd like
to remove, then removes said item

bool interact (); // this will ~~return~~ change interaction
to true and return it

bool investigate (); this will change investigation to
true and return it

void moveToLocation (); ~~prompt the user to~~
check where user currently is. Prompt user
to select where location to go to. If
user is in lighthouse and they have the key
then they have the option to go in the
~~lighthouse~~ ^{game}. If they are in the game they
can go in the lighthouse; In order to move
about use the map STL player map

current location pointer pointing to currently occupied
space, update it to the new space
void setCurrentLocation (Space #05); 3
Space getCurrentLocation();

661747	697	772
610669	508	505
574617	529	689

Put
in game
class

Space
Protected:

Player pobj; vector <String> itemStorage;
String uniqueItem1, uniqueItem2;

Public:

Space(); constructor

~Space(); destructor

virtual void describePlace(); this function will be called
each time the user enters the space and

will give a general description of the environment
virtual void revealInvestigation(player obj); this function
first tests and checks if the player investigate

object returns true. If it returns true then
it will give a description of areas that are
able to be interacted with (special) and what
items are available to be picked up

virtual void displaySpaceItems(); iterates through
the vector to display items available in that space

virtual void addOrRemoveItems(player obj); Takes a
player object. If player adds an item then the
itemStorage for this location will look its item. If
the player drops an item then it will add that
item to itemStorage

virtual void special() = 0;

1 Shack

2 Garden

3 Beach

4 Lighthouse

5 Boat

Shack, garden, lighthouse, boat, beach

```

Game
Private:
    Player obj;    Space *obj;
int userSelection;
    bool winGame, looseGame;
Public
    Game(); constructor
    ~Game(); destructor
void Menu(); menu item for determining whether
the user picks up or drops items, investigates
the area, checks their backpack, uses special
or moves locations
void gameIntro(); this will display describe the opening
scene for the player
    bool gameWon(); returns true once player wins
    bool gameLost(); returns true if player loses

```

Initialize all of the space objects in Game,
then set the pointers in the game constructor