Nathan Greenlaw

ONID: greenlan

12/6/2016

Final Project

The goal of the project is to create a text-based game using the concepts we have learned this quarter. The game I decided to create is called “The Moose are Loose.” The base idea is that while you are away at work moose have taken up residence in your home and you need to use items found around the house to scare them away. Each moose type is afraid of a specific item and each moose may turn off the lights or lock the door of the room they are in. There are 11 rooms total in the house and 5 different types of moose. There are also 11 items with 6 being keys and 5 being other various objects. The goal is to navigate the house and remove the moose before you run out of time.

The program will follow this pseudocode in order to create this game:

Create the rooms

Create a vector called House filled with room pointers

Ask for the difficulty which is number of moose

Place the number of moose around the house randomly

Place the items in the rooms

Create the player and start the game

Display the starting text along with a “cheat sheet“ for graders

Loop for 90 actions or until all moose have been removed

Ask player to take an action and show if there is a moose in the room

1. Move to another room

Display room choices and then move to the room if door is unlocked

If the door is locked then ask player to try an item, if it is correct then they move to the room, if not nothing happens

2. Search/Remove Moose

Display the which item is on the ground and if there is a moose in the room ask if they want to remove it if the lights are on, otherwise the player tries to turn on the lights with a 70% chance of success.

If they choose to pick up the item it is placed in the bag. If the bag is full then they can choose to swap the item

if they choose to remove the moose they will be asked to try an item. If successful then the number of moose in the house decrements and if not then nothing happens

3. Look at bag

Displays the items in the bag

4. Concede to the Moose

Exits game with the moose winning

Depending on how the game ends, display one of 3 ending scenes.

Program finishes running

My original writing and planning on paper is included at the end as images but the program changed a lot from how it started out. Originally I was going to have a room class, a player class, a moose class and an item class, but I decided early on that I could just have the moose and items be strings of text and that would save on memory and ease the programming process. This left me with only a player class and a room class. I also decided that even though we had been working on dynamic memory allocation all quarter it made more sense to me to have the rooms and player not be dynamically allocated as these objects would exist throughout the entire game and at no point would they be deleted or created during runtime.

The room class holds all the information about the items and moose that are in the game. The items and moose are string members of the room class where a string none is the default nothing is in the room and a string member that keeps what type of key unlocks the door. There are also Boolean members that control whether or not the lights are on, the door is locked and if there is an item or moose in the room. Each room connects to a maximum of 4 rooms labeled right, left, up and down where if one of the rooms didn't have 4 connections, the spots were set to NULL.

The player class holds all the information regarding the actions and game states. It also contains a vector of a bag that can only contain 6 items. It also contains a pointer to the current room the player is in so that it can keep track of where the player is in the house.

I had other people test my game multiple times to see if it worked and how it functioned as well as running through the game myself. I decided on a time limit of 90 actions because it seemed to be above the average amount of time it took for players to finish it on the medium difficulty.

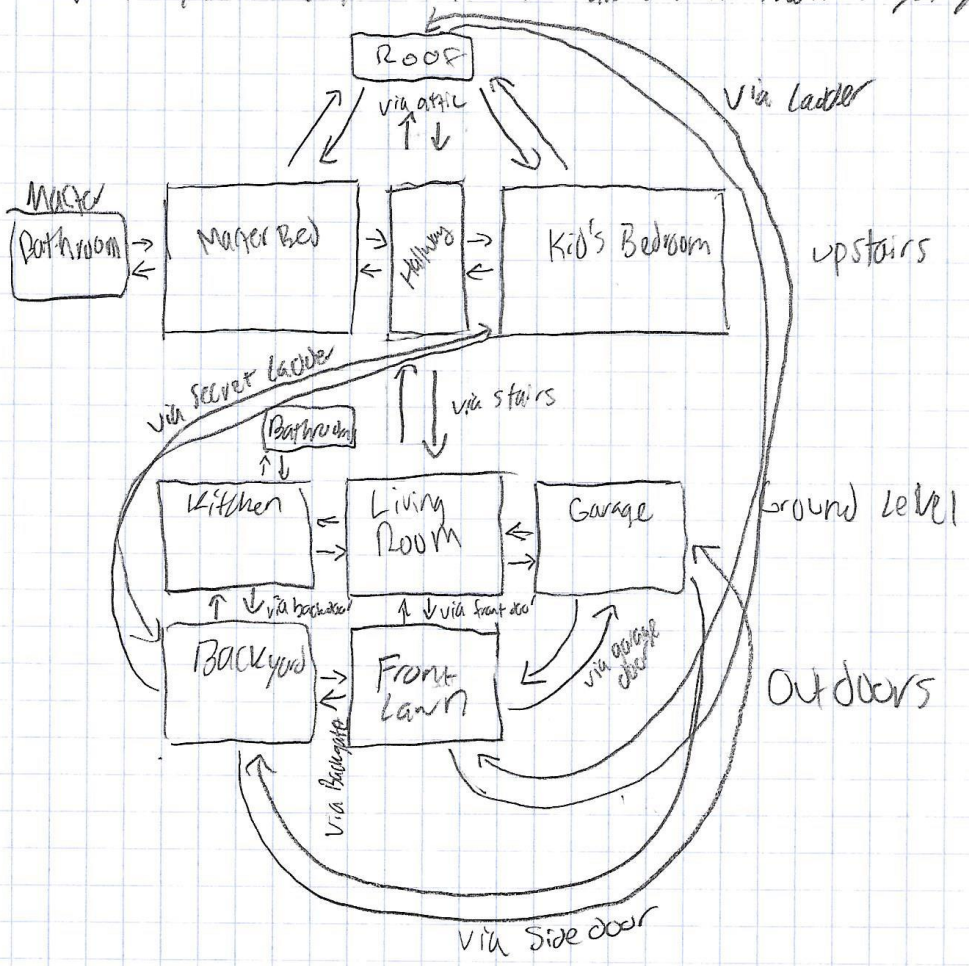
Reflection

There was one major problem that I encountered that I think I fixed as I was unable to replicate it after testing the program numerous times but when assigning the items to the rooms somehow there was an occasional mod by 0 which caused there to be an error and dump the core. But even before I altered it the bug only happened occasionally so I am not positive that the bug will not come up again, so if it does then just re-try it and it should work. After the alteration I tested it about 15 times with no error which before it would give an error every 5 or so, which makes me feel more confident about it. In terms of everything else, I think it was a smart decision for me to transition away from having moose and item objects which is how I started. The program itself started to get unnecessarily complicated and the items and moose themselves each had about 2 members and 2 functions that were easily moved. Below is my original on paper planning including a map of the house such that the graders can have an easier time using the program.

Scans of Original Plan

The Moose are LOOSE:

Your neighbor says moose go into your house...
Your house has been invaded by Moose, but luckily you have all the necessary materials since this is a rather common occurrence. Search the house for these materials and use them to get your house back.



- There are different types of moose, all need a specific item to scare them away. Moose hide in plain sight
- The bag can store up to 5 things, must discard to make room
  - Randomly place 2-4 moose around the house of abstract class Moose
- Randomly place items all around house, if bag full then ask to swap

W N S E

Abstract Class: Room

	Derived classes	Pointers: Right	Left	UP	Down
Outdoors	Front Lawn	Garage	Backyard	Living Room	Roof
	Backyard	Front Lawn	Kid's Bedroom	Kitchen	Garage
Ground level	Garage	Backyard	Living Room	NULL	Front Lawn
	Living Room	Garage	Kitchen	Hallway	Front Lawn
	Kitchen	Living Room	NULL	Bathroom	Backyard
	Bathroom	NULL	NULL	NULL	Kitchen
Upstairs	Kid's Bedroom	Backyard	Hallway	Roof	NULL
	Hallway	Kid's Bed	Master Bed	Roof	Living Room
	Master Bed	Hallway	Master Bath	Roof	NULL
	Master Bath	Master Bed	NULL	NULL	NULL
	Roof	Kid's Bed	Master Bed	Front Lawn	Hallway

Return Items/Moose

Members

- Item in Room?, Item
- Moose in Room?, Moose
- Pointers: Up, Down, Left, Right
- Type of Room (Name)
- Locked door (Yes or No)
- Lights on (Yes or no)

Functions

- Action → Move, Search, Lights, Remove Moose, Unlock door
- Search → Look for item if lights are on
- Move → Menu for moving, moves if door not Locked
- get/set functions for members
- Remove Moose → Try an item in bag to Remove Moose

Player functions

Abstract Class: Moose

Derived classes

Members

- turns off lights?
- Locks door?
- Afraid of
- Room location
- Successful removal (string)
- Failed removal (string)

Functions

- get/set functions
- LockDoors → if it locks doors Then change Room member
- lights → if it turns of light Then "

- If the moose is presented with the item it is afraid of the moose runs away and the item is consumed

---

Type	Lights off?	Locks Doors	Afraid of
Smelly Moose	No	Yes	Air freshener Spray
Sleepy Moose	Yes	Yes	Alarm Clock
Vegetarian Moose	No	No	Meat
Posh Moose	NO	NO	Garden Gnomes
Copycat Moose	Yes	No	Mirror

Abstract Class: Item

Members

- Type

Functions

- Get / Set

do while (vector.(at) ⇒ (item in Room != "none")
+ for keys ↳ and random type != room type

Items are placed Randomly around the house and unlock doors or remove mouse

---

	Type	function
Keys	Living Room	unlocks the door to the type
	Garage	"
	Bathroom	"
	Kid's Bed	"
	M. Bed	"
	M. Bath	"
Mouse Removal	Air fresh	Removes the appropriate mouse (Smelly)
	Alarm Clock	" (Sleepy)
	Meat	" (veggie)
	Garden gnomes	" (Posh)
	Mirror	" (copycat)

Class Player

members
- Bag
- CurrentRoom

functions
- Action :
  - Move → Locked door
  - Search → Remove Move or look for item
    - Lights

STEPS

Prep Game work:

- Create Player
- Create all Rooms
- Create house (array of Rooms)
- Set pointers for all Rooms
- Determine where moose go (Random number)
- Set moose in Rooms
- Determine where items go
- Set items in Rooms (all items)

Start Game:

- Display text for starting game
- Loop for some amount of time (or Until All moose are gone
- Allow player to take action
- execute action
- resolve action
- increment counter