Author: Johannes Pikel
Date: 2016.05.24
Description: Final Project

Understanding

My understanding of the program is that I am to write a game that has a series of rooms, that the user/character traverses. There are items that are scattered into different rooms and a certain combination of these items are required to solve a puzzle or successfully complete the goal.

I am to create my own theme for this project.

Theme: I plan to create a theme based around a space station adventure. Specifically, I want the navigation to use all four pointers in each room or space. So I will layout my rooms so that each room could potentially connect to four other rooms. To do this I will need six rooms total to start. I could always add a dead end later. Which I will do something of this sort when one of my rooms disappears and another room takes its place.

Backstory: You are the hero and have been left to tend to the Spherical Space Station while all the scientists take shore leave on Earth. You are to mind all the projects and petri dishes, which frankly you've never been very fond of and of course the housekeeping duties. One morning you are woken up to the awful sound of metallic gnawing and grinding. The only thing you of that could cause this kind of awful sound are ferrous space leaches. The challenge is to wander the Spherical Space Station in search of the items needed to perform a spacewalk and remove the leaches from the hull of the ship. If the time limit runs out the leaches have made too many breaches through the hull and the entire space station depressurizes resulting in your unfortunate death. There will also be an extra challenge to gain access to the end game as well that is a simple puzzle on top of collecting items.

Items: I plan to implement these as a Class so I can have a nice variety and always add new items. Mainly the different derived classes will have different descriptions. This will make it easy to simply declare a new derived item and not need to pass any parameters in. I will create a couple of dud items that can wind up taking up room in the inventory unnecessarily and not be of much use, except waste precious time.

Character: I plan to create a class for the character as well call the Ragdoll. This class will have an inventory that will be an array of pointers to items. I plan to limit it to 4 items total, two of which are specific to the inventory slot they may go into. Slot 0 is the body or worn and slot 1 is in the hands or holstered. The user needs to be able to open the character's inventory inspect items and possibly drop them into a room. Slots 2 and 3 will be considered a backpack and only smaller items will fit into this.

Rooms: I want to create more than three unique rooms. One of my rooms will definitely have a light switch that will not allow the character to do anything until the lights are turned on. Items need to be searched for and do not automatically appear when a character enters a new room. My rooms will have four pointers to other rooms. As well as an array of pointers to Items that will grow and shrink as needed. I do not want to use a stack or queue for this because when a character searches a room, then should be able to find all the possible items in that room instead of just one at a time. It also allows the character to pile up a bunch of items in a room and look at all of them at once instead of just the first or last depending on whether a stack or queue is used. I could use an array but would rather have more control over future implementation so will use an array instead.

Interacting with each room. I plan to include several options that reduce the time left by a set amount or by a variable amount. There will also be one room where a character can gain a few time units by interacting with the room but only a few at a time. But to keep the urgency up this particular room will only be able to be used a few times until the leaches destroy the extra element.

End game: I'd like to write a few different simple endings depending on if the character ran out of time, had the correct items in their inventory or if missing a few. There will only be 1 victory style. Finishing in time and having 4 correct items in their inventory.

Input: I want to have a very simple char input for movement. So the actionMenu will print differently depending on prerequisites such as if the character has searched the room or if the room is dark. The character will be able to move through doors 1, 2, 3, 4. Search the room, open their inventory, examine an item in a room, get an item to transfer it into the character's inventory and interact with the room.

HighScore: I wanted to add something extra in that I wanted to keep high scores for winners between games. But only for the ten best winners. Obviously my simple game most people get the best score possible but It's great practice for reading from and writing to a file. So I will create a couple of functions that read in from a file, display the high scores and then write back to file so the next time the game is run the high scores may be retrieved!

Pre-reflections

MAP!! I am adding this here to help give navigation assistance while wandering the space station. This is the matrix of doors and how they are connected.

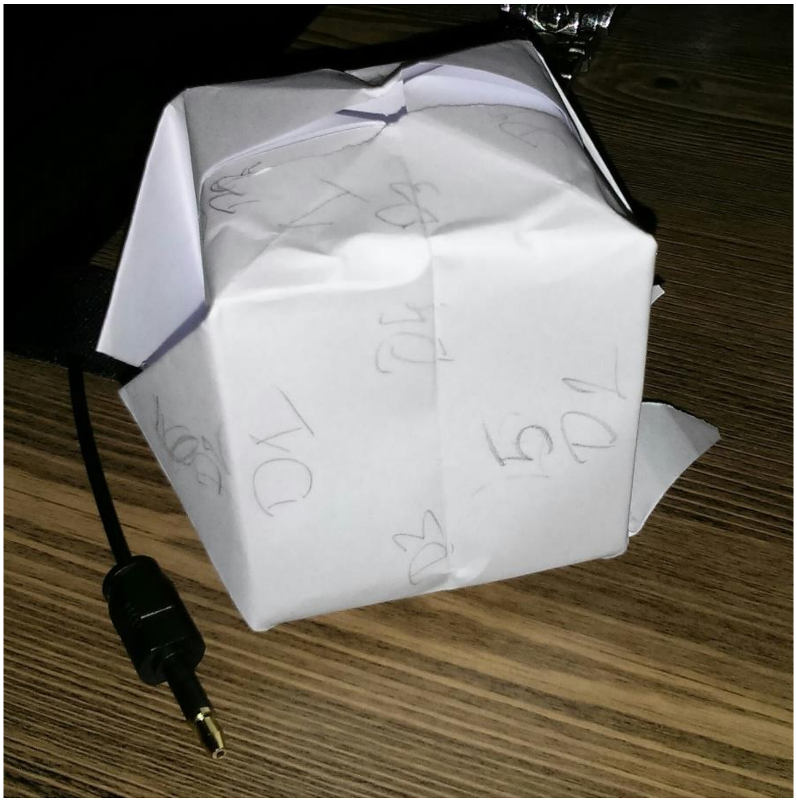
Space	Door 1	Door 2	Door 3	Door 4
Sleeping Quarters	Laboratory	Truss / NOWHERE	Cargo Bay	Equipment Bay
Laboratory	Bridge	Sleeping Quarters	Equipment Bay	Truss/Airlock
Bridge	Equipment Bay	Laboratory	Truss/Airlock	Cargo Bay
Equipment Bay	Cargo Bay	Bridge	Sleeping Quarters	Laboratory
Cargo Bay	Truss/Airlock	Equipment Bay	Bridge	Sleeping Quarters
Truss	Sleeping Quarters	Cargo Bay	Laboratory	Bridge
Airlock	NOWHERE	Cargo Bay	Laboratory	Bridge

** is the extra room that will replace the truss eventually when the timer gets to a certain point. Also the character won't be able to be in that room obviously. But I will state that if Items are left in the Truss when it disappears those items are lost to the vacuum of space! Also once the new room gets added. Sleeping quarters door2 will no longer function and Airlock door1 will no longer function. This because door1 of airlock has to exit to space!

Generally, Door1 leads through the station in a straight line, door2 return back to where you came from

If you go through Door3 go back through Door4. If you go through Door4 you can go back through door3. I even had to create an origami cube to write on to figure this one out ☺

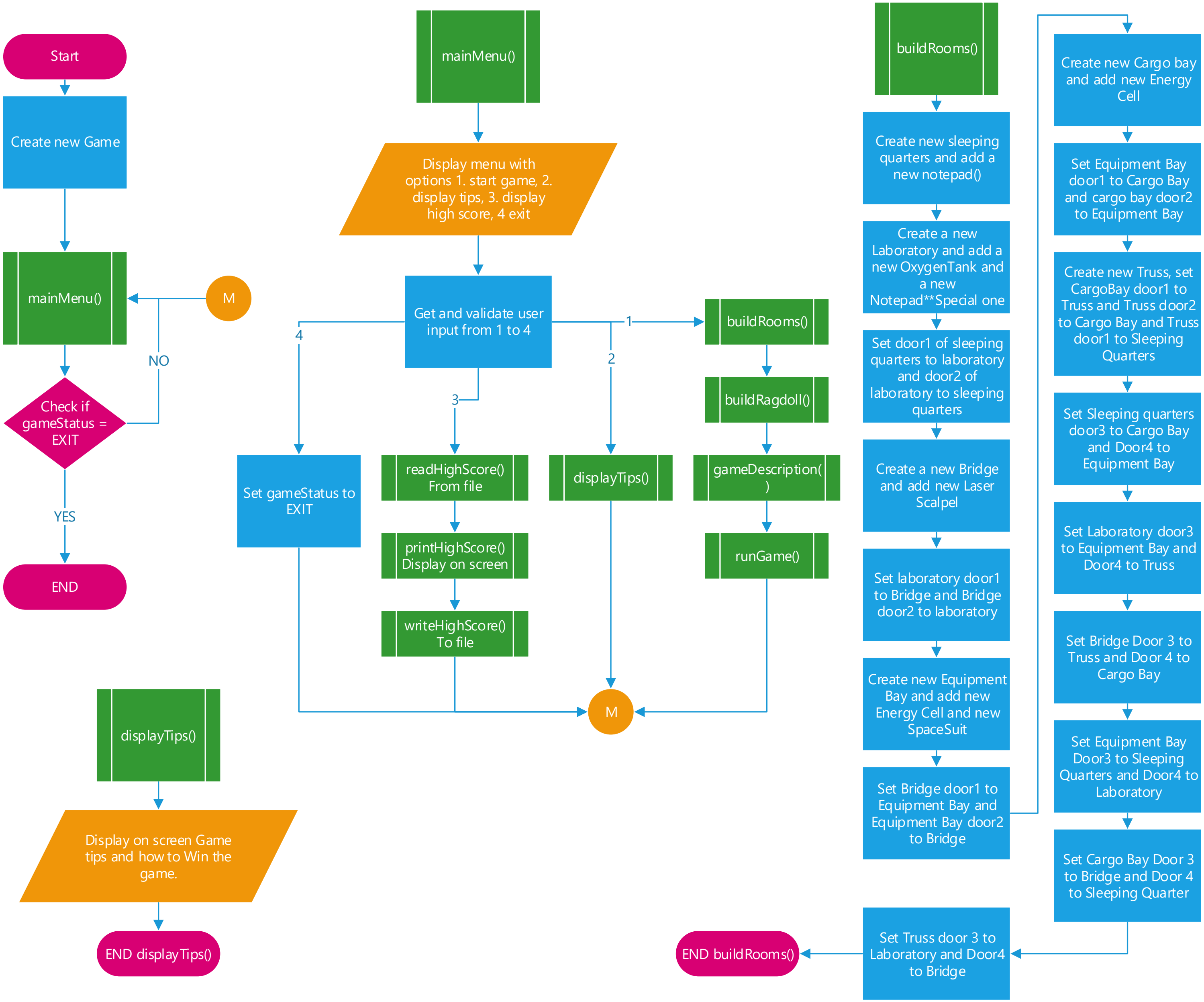
Origami project:



Author: Johannes Pikel
Date: 2016.05.24

Description: Final Project

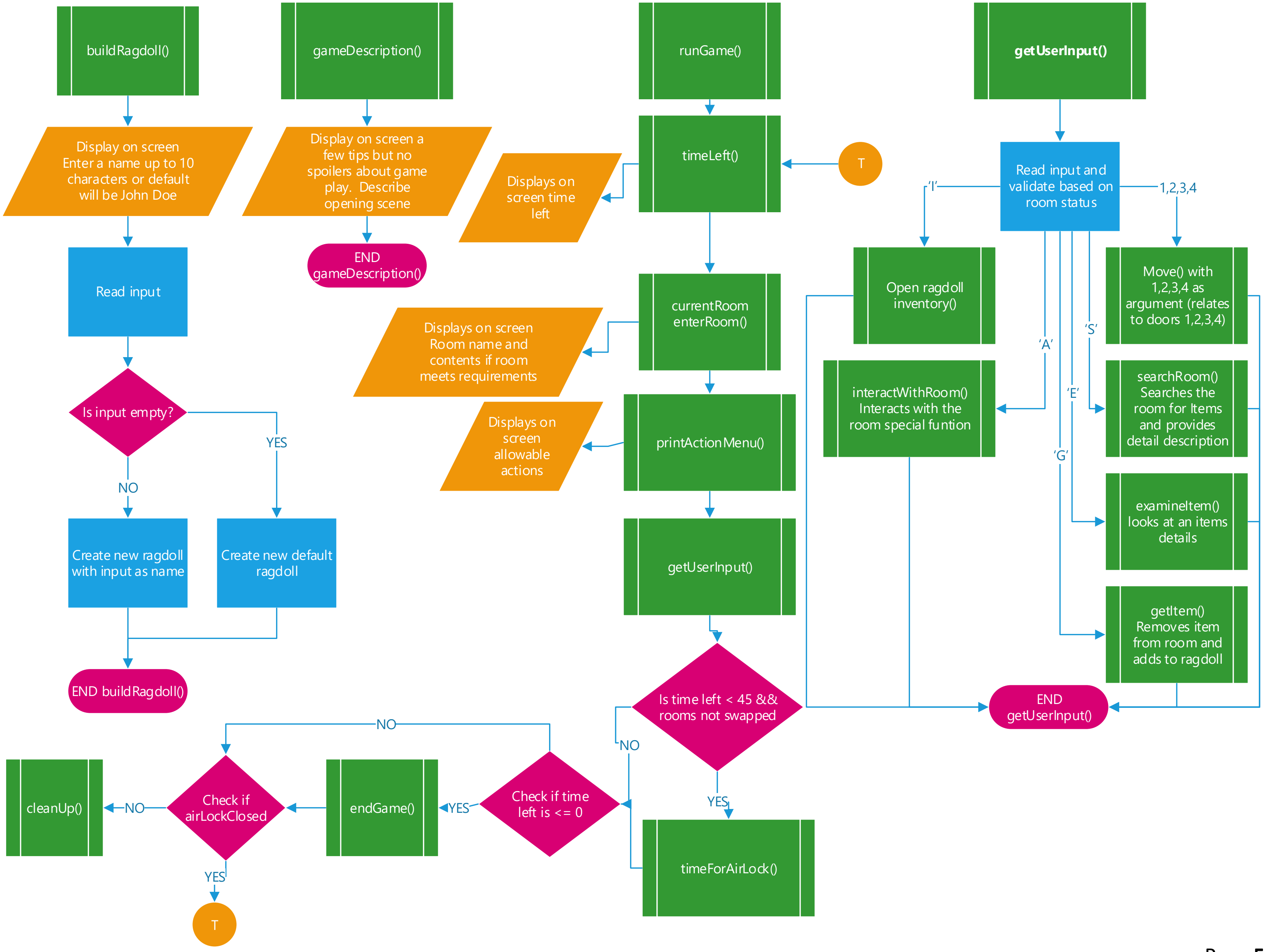
Flowchart


Start
Create new Game
mainMenu()
M
NO
Check if gameStatus = EXIT
YES
END
mainMenu()
Display menu with options 1. start game, 2. display tips, 3. display high score, 4 exit
Get and validate user input from 1 to 4
4
3
2
1
Set gameStatus to EXIT
readHighScore() From file
printHighScore() Display on screen
writeHighScore() To file
displayTips()
buildRooms()
buildRagdoll()
gameDescription( )
runGame()
M
displayTips()
Display on screen Game tips and how to Win the game.
END displayTips()
buildRooms()
Create new sleeping quarters and add a new notepad()
Create a new Laboratory and add a new OxygenTank and a new Notepad**Special one
Set door1 of sleeping quarters to laboratory and door2 of laboratory to sleeping quarters
Create a new Bridge and add new Laser Scalpel
Set laboratory door1 to Bridge and Bridge door2 to laboratory
Create new Equipment Bay and add new Energy Cell and new SpaceSuit
Set Bridge door1 to Equipment Bay and Equipment Bay door2 to Bridge
Create new Cargo bay and add new Energy Cell
Set Equipment Bay door1 to Cargo Bay and cargo bay door2 to Equipment Bay
Create new Truss, set CargoBay door1 to Truss and Truss door2 to Cargo Bay and Truss door1 to Sleeping Quarters
Set Sleeping quarters door3 to Cargo Bay and Door4 to Equipment Bay
Set Laboratory door3 to Equipment Bay and Door4 to Truss
Set Bridge Door 3 to Truss and Door 4 to Cargo Bay
Set Equipment Bay Door3 to Sleeping Quarters and Door4 to Laboratory
Set Cargo Bay Door 3 to Bridge and Door 4 to Sleeping Quarter
Set Truss door 3 to Laboratory and Door4 to Bridge
END buildRooms()

Author: Johannes Pikel
Date: 2016.05.24

Description: Final Project

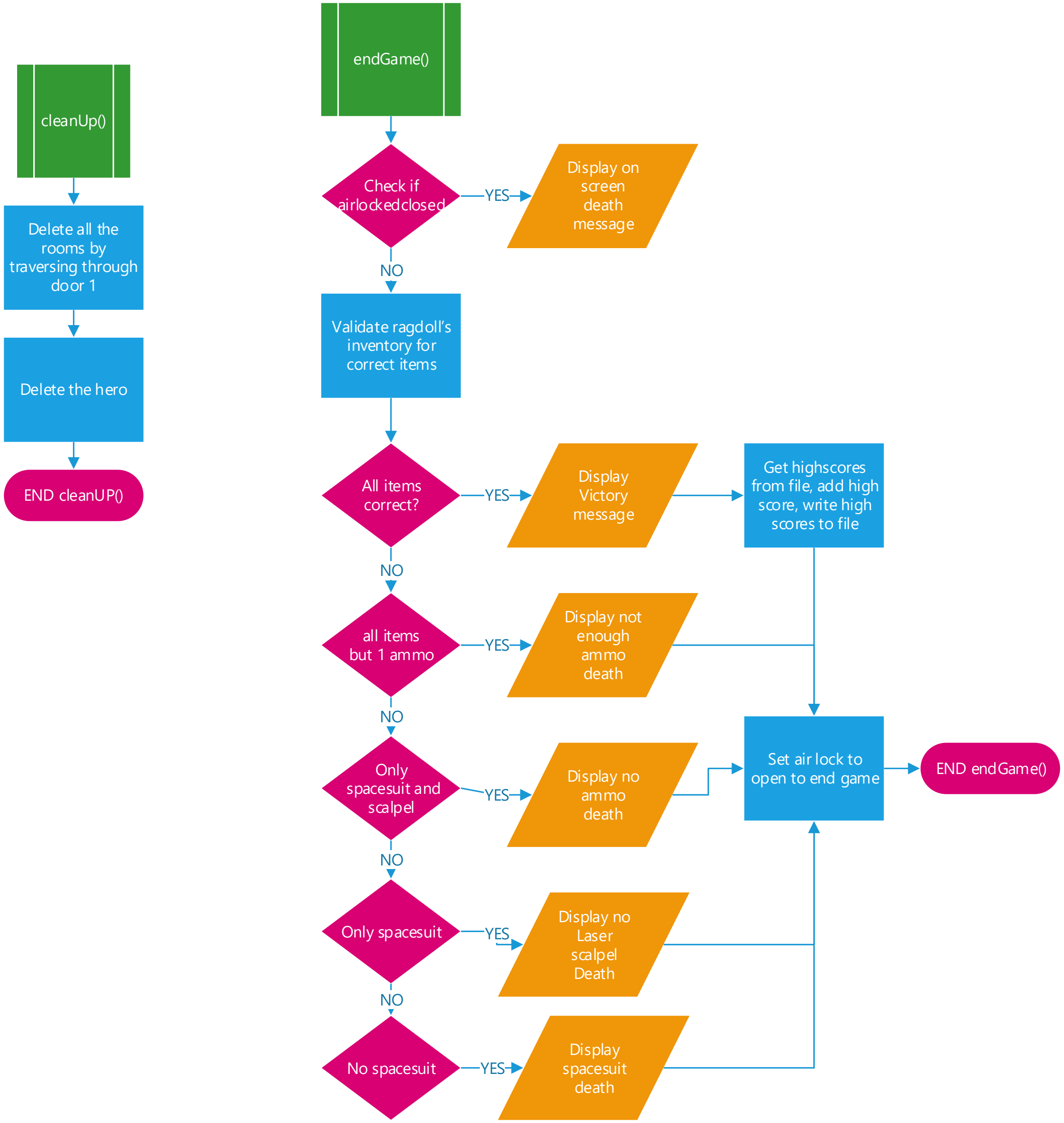
Flowchart


buildRagdoll()
Display on screen Enter a name up to 10 characters or default will be John Doe
Read input
Is input empty?
YES
NO
Create new ragdoll with input as name
Create new default ragdoll
END buildRagdoll()
gameDescription()
Display on screen a few tips but no spoilers about game play. Describe opening scene
END gameDescription()
runGame()
timeLeft()
T
Displays on screen time left
currentRoom enterRoom()
Displays on screen Room name and contents if room meets requirements
printActionMenu()
Displays on screen allowable actions
getUserInput()
Is time left < 45 && rooms not swapped
NO
YES
timeForAirLock()
Check if time left is <= 0
NO
YES
endGame()
Check if airLockClosed
NO
cleanUp()
YES
T
getUserInput()
Read input and validate based on room status
'I'
1,2,3,4
Open ragdoll inventory()
Move() with 1,2,3,4 as argument (relates to doors 1,2,3,4)
'A'
'S'
'E'
'G'
interactWithRoom() Interacts with the room special funtion
searchRoom() Searches the room for Items and provides detail description
examineItem() looks at an items details
getItem() Removes item from room and adds to ragdoll
END getUserInput()

Author: Johannes Pikel
Date: 2016.05.24

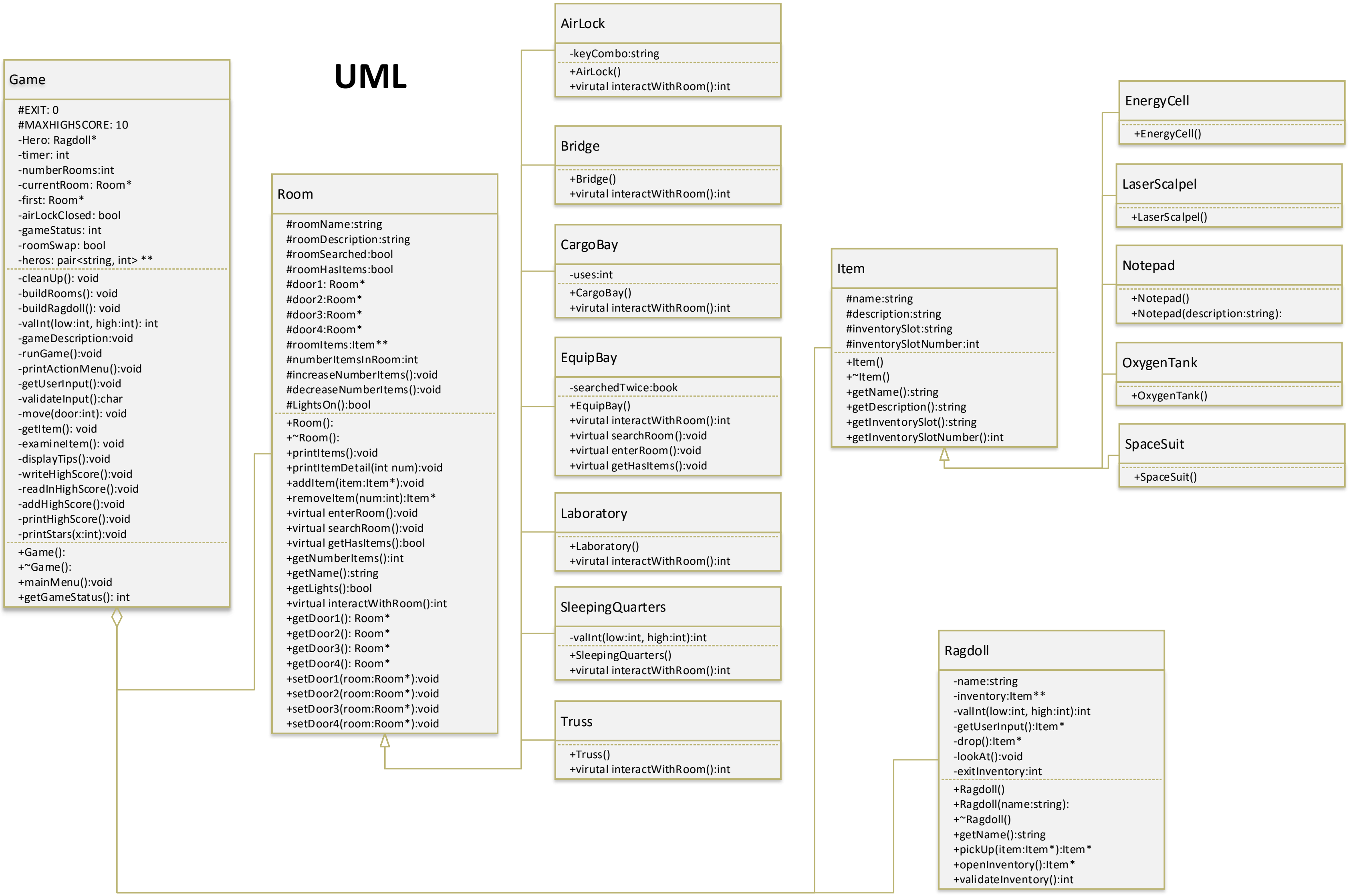
Description: Final Project

Flowchart


cleanUp()
Delete all the rooms by traversing through door 1
Delete the hero
END cleanUP()
endGame()
Check if airlockedclosed
YES
Display on screen death message
NO
Validate ragdoll's inventory for correct items
All items correct?
YES
Display Victory message
Get highscores from file, add high score, write high scores to file
NO
all items but 1 ammo
YES
Display not enough ammo death
NO
Only spacesuit and scalpel
YES
Display no ammo death
Set air lock to open to end game
END endGame()
NO
Only spacesuit
YES
Display no Laser scalpel Death
NO
No spacesuit
YES
Display spacesuit death

Author: Johannes Pikel
Date: 2016.05.24
Description: Final Project

UML



Game
#EXIT: 0
#MAXHIGHSCORE: 10
-Hero: Ragdoll*
-timer: int
-numberRooms:int
-currentRoom: Room*
-first: Room*
-airLockClosed: bool
-gameStatus: int
-roomSwap: bool
-heros: pair<string, int> **
-cleanUp(): void
-buildRooms(): void
-buildRagdoll(): void
-valInt(low:int, high:int): int
-gameDescription:void
-runGame():void
-printActionMenu():void
-getUserInput():void
-validateInput():char
-move(door:int): void
-getItem(): void
-examineItem(): void
-displayTips():void
-writeHighScore():void
-readInHighScore():void
-addHighScore():void
-printHighScore():void
-printStars(x:int):void
+Game():
+~Game():
+mainMenu():void
+getGameStatus(): int

Room
#roomName:string
#roomDescription:string
#roomSearched:bool
#roomHasItems:bool
#door1: Room*
#door2:Room*
#door3:Room*
#door4:Room*
#roomItems:Item**
#numberItemsInRoom:int
#increaseNumberItems():void
#decreaseNumberItems():void
#LightsOn():bool
+Room():
+~Room():
+printItems():void
+printItemDetail(int num):void
+addItem(item:Item*):void
+removeItem(num:int):Item*
+virtual enterRoom():void
+virtual searchRoom():void
+virtual getHasItems():bool
+getNumberItems():int
+getName():string
+getLights():bool
+virtual interactWithRoom():int
+getDoor1(): Room*
+getDoor2(): Room*
+getDoor3(): Room*
+getDoor4(): Room*
+setDoor1(room:Room*):void
+setDoor2(room:Room*):void
+setDoor3(room:Room*):void
+setDoor4(room:Room*):void

AirLock
-keyCombo:string
+AirLock()
+virutal interactWithRoom():int

Bridge
+Bridge()
+virutal interactWithRoom():int

CargoBay
-uses:int
+CargoBay()
+virutal interactWithRoom():int

EquipBay
-searchedTwice:book
+EquipBay()
+virutal interactWithRoom():int
+virtual searchRoom():void
+virtual enterRoom():void
+virtual getHasItems():void

Laboratory
+Laboratory()
+virutal interactWithRoom():int

SleepingQuarters
-valInt(low:int, high:int):int
+SleepingQuarters()
+virutal interactWithRoom():int

Truss
+Truss()
+virutal interactWithRoom():int

Item
#name:string
#description:string
#inventorySlot:string
#inventorySlotNumber:int
+Item()
+~Item()
+getName():string
+getDescription():string
+getInventorySlot():string
+getInventorySlotNumber():int

EnergyCell
+EnergyCell()

LaserScalpel
+LaserScalpel()

Notepad
+Notepad()
+Notepad(description:string):

OxygenTank
+OxygenTank()

SpaceSuit
+SpaceSuit()

Ragdoll
-name:string
-inventory:Item**
-valInt(low:int, high:int):int
-getUserInput():Item*
-drop():Item*
-lookAt():void
-exitInventory:int
+Ragdoll()
+Ragdoll(name:string):
+~Ragdoll()
+getName():string
+pickUp(item:Item*):Item*
+openInventory():Item*
+validateInventory():int

Testing

For this program I did a significant amount of unit testing. First I created my Item class and derived and then worked through declaring Items and accessing all the functions. Then I did so with the Rooms and their specific functions like the addItem and removeItem functions. I did all of this directly from Main. Once I got to a point that these smaller pieces worked well. I developed the Ragdoll and did similar testing with each function.

Lastly I built the Game class and put all the interaction between the different Classes together. The thing that took me the longest was actually implementing the high score features, reading from a file and writing to a file. I kept getting odd errors where my function would write to the file and it would be empty. I did finally come up with a solution and think it works fairly well. When I read in from the file I store all the names and scores in a dynamic array. There is no need to keep that memory around so once I take the appropriate actions with the array whether it is displaying it to screen or adding a new hero that won to it. I then delete the highscore file, recreate and write to it. I think something was no going correctly between closing the file for reading and opening it again for writing. This works well and once I'm done writing out to the file I delete the array, to free memory so the system is not constantly holding on to the memory while other things are happening like playing the game.

CASE	INPUT	FUNCTION	EXPECTED	ACTUAL
Move to each room according to the matrix	1,2,3,4 in each room	getUserInput() and move()	Move to appropriate Room per matrix	As expected
Each Room should be able to hold any number of items	Add as many Items as in game to the room	Room::addItem()	Adds items to the room and no errors	As expected
Pickup an item from a room	If a room has an item then pick up the correct item per user input	Room:removeItem() and Ragdoll::addItem()	Removes item from room and adds item to inventory of ragdoll	As expected
Run out of time	Lose and die	endgame()	Lose and die ☹	As expected
Test each of the possible win or lose scenarios	Win only if correct items in inventory, and get appropriate description per loss	Endgame()	Get correct display on screen	As expected
Interact with each rooms special function	'a'	getUserInput()	Each room has a special feature and show work correctly	As expected
Run the game appropriately!	Play game	mainMenu()	Run the game from start to finish without errors	As expected

Author: Johannes Pikel
Date: 2016.05.24
Description: Final Project

Reflections

I had a blast with the assignment!! At first it was a bit daunting and I definitely had some ideas that were out of this world and I never thought I could accomplish, until I toned it down a bit. I tried my best to be creative with what happens and how.

Even though my game is the same each time, I think it fits the requirements because this allows for easy testing and makes sure the appropriate items are in the game so it can be won.

I had the hardest time with the writing and reading from the file for the highscore. My final solution was to remove the file and then recreate it and write to. Which seems to work well now.

I also had some trouble with the way I implemented the Equipment Room because for this room the lights are off initially, so I had to add some additional criteria to how I printed the actionMenu and validated userInput so that it was not possible to get and item or examine an item, before the lights were turned on and the room search. I tested this space quite a bit and have hopefully worked out all the bugs.

It is possible to get a higher score than when the new airlock appears to if time and room interactions are used wisely. I left the exact details of what my rooms do a bit shrouded in mystery, so that way I do not give away the details. But generally they interact with the amount of time that is left on the clock. The Equipment Room does not and instead it has the lights off which is an obstacle, so turn the lights on first.

It took a bit of working out the details to get the actionMenu() to only print the possible options when they are available. For instance, interacting with a room, getting an item, or examining an item are only possible after the room has been searched and the lights are on. In the case of the Equipment room the lights need to be turned on first and then the room searched. So it actually needs to be searched twice.

I hope you enjoy the space adventure and get a high score! It's possible if the random number generator scores high to get and even higher score than what's already in the list ☺ I've played it many times, trying to find a way to break it and so far have been able to fix all my bugs. Hopefully it works well for you as well!

Thank you, Andy for being an awesome TA this class. You have been great to work with and always responded fast to any of my questions. I cannot say enough about how excellent your feedback has been in a constructive manner. Thank you very much for all your patience and help along the way. Best of you luck to you in your last terms! Perhaps we'll see each other down the road at a company!