KIVNA'S REVENGE

EDI
EDI

Raj Pannu, Rohan Sharma, Kaelan Croucher, Marcus Caparini

December 9th, 2022

TABLE OF CONTENTS

INTRODUCTION

PROJECT MANAGEMENT

TEAM ROLES

Team Member	Design - Draft	Design – Final	Implementation - Basic	Implementation - Final
Raj	Programmer/Designer	Programmer/Designer	Programmer/Designer	Reporting Lead
Kaelan	Programmer/Designer	Programmer/Designer	Programmer/Designer	Programmer/Designer
Marcus	Programmer/Designer	Programmer/Designer	Programmer/Designer	Programmer/Designer
Rohan	Programmer/Designer	Programmer/Designer	Programmer/Designer	Programmer/Designer

DEVEVLOPMENT PROCESS

CODE REVIEW PROCESS
- Fixed errors when they became apparent in debugging. Routine pipeline checks
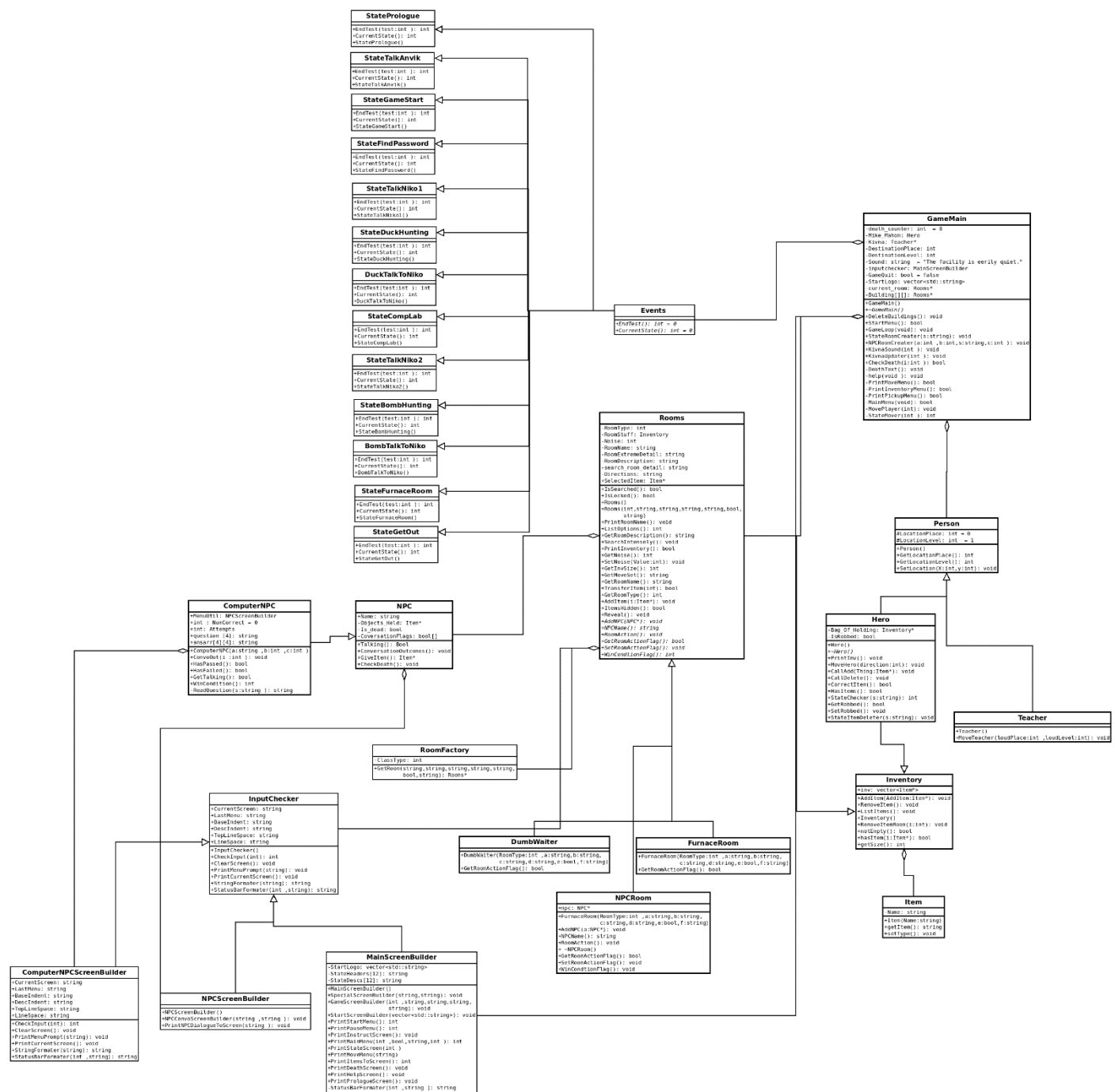
COMMUNICATION TOOLS
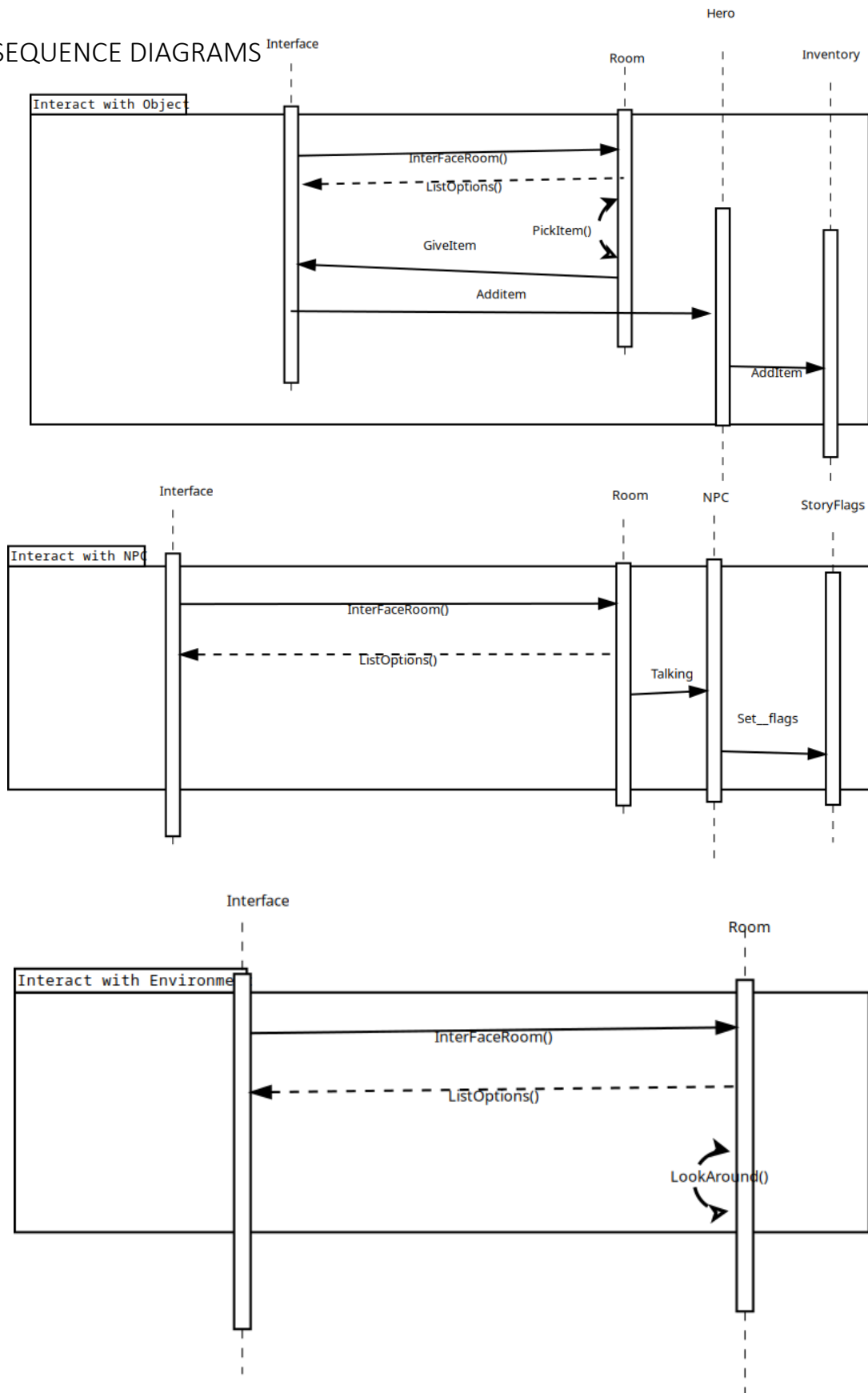- Discord

CHANGE MANAGEMENT
- Gitlab

SOFTWARE DESIGN

DESIGN – CLASS DIAGRAMS

StatePrologue
+EndTest(test:int): int
+CurrentState(): int
+StatePrologue()

StateTalkAnvik
+EndTest(test:int): int
+CurrentState(): int
+StateTalkAnvik()

StateGameStart
+EndTest(test:int): int
+CurrentState(): int
+StateGameStart()

StateFindPassword
+EndTest(test:int): int
+CurrentState(): int
+StateFindPassword()

StateTalkNiko1
+EndTest(test:int): int
+CurrentState(): int
+StateTalkNiko1()

StateDuckHunting
+EndTest(test:int): int
+CurrentState(): int
+StateDuckHunting()

DuckTalkToNiko
+EndTest(test:int): int
+CurrentState(): int
+DuckTalkToNiko()

StateCompLab
+EndTest(test:int): int
+CurrentState(): int
+StateCompLab()

StateTalkNiko2
+EndTest(test:int): int
+CurrentState(): int
+StateTalkNiko2()

StateBombHunting
+EndTest(test:int): int
+CurrentState(): int
+StateBombHunting()

BombTalkToNiko
+EndTest(test:int): int
+CurrentState(): int
+BombTalkToNiko()

StateFurnaceRoom
+EndTest(test:int): int
+CurrentState(): int
+StateFurnaceRoom()

StateGetOut
+EndTest(test:int): int
+CurrentState(): int
+StateGetOut()

Events
+EndTest(): int = 0
+CurrentState(): int = 0

GameMain
-death_counter: int = 0
-Mike Mahon: Hero
 Kivna: Teacher*
-DestinationPlace: int
 DestinationLevel: int
-Sound: string = "The facility is eerily quiet."
-inputchecker: MainScreenBuilder
-GameQuit: bool = false
-StartLogo: vector<std::string>
-current_room: Rooms*
-Building[][]: Rooms*
+GameMain()
+~GameMain()
+DeleteBuildings(): void
+StartMenu(): bool
+GameLoop(void): void
+StateRoomCreater(s:string): void
+NPCRoomCreater(a:int ,b:int,s:string,c:int): void
+KivnaSound(int): void
+KivnaUpdater(int): void
+CheckDeath(i:int): bool
-DeathText(): void
-help(void): void
-PrintMoveMenu(): bool
-PrintInventoryMenu(): bool
-PrintPickupMenu(): bool
 MainMenu(void): bool
+MovePlayer(int): void
-StateMover(int): int

Rooms
-RoomType: int
-RoomStuff: Inventory
-Noise: int
-RoomName: string
-RoomExtremeDetail: string
-RoomDescription: string
-search room detail: string
 Directions: string
+SelectedItem: Item*
+IsSearched(): bool
+IsLocked(): bool
+Rooms()
+Rooms(int,string,string,string,string,bool,
       string)
+PrintRoomName(): void
+ListOptions(): int
+GetRoomDescription(): string
+SearchIntensely(): void
+PrintInventory(): bool
+GetNoise(): int
+SetNoise(Value:int): void
+GetInvSize(): int
+GetMoveSet(): string
+GetRoomName(): string
+TransferItem(int): bool
+GetRoomType(): int
+AddItem(i:Item*): void
+ItemsHidden(): bool
+Reveal(): void
+AddNPC(NPC*): void
+NPCName(): string
+RoomAction(): void
+GetRoomActionFlag(): bool
+SetRoomActionFlag(): void
+WinConditionFlag(): int

Person
#LocationPlace: int = 0
#LocationLevel: int = 1
+Person()
+GetLocationPlace(): int
+GetLocationLevel(): int
+SetLocation(X:int,y:int): void

Hero
-Bag Of Holding: Inventory*
-IsRobbed: bool
+Hero()
+~Hero()
+PrintInv(): void
+MoveHero(direction:int): void
+CallAdd(Thing:Item*): void
+CallDelete(): void
+CorrectItem(): bool
+HasItem(): bool
+StateChecker(s:string): int
+GetRobbed(): bool
+SetRobbed(): void
+StateItemDeleter(s:string): void

Teacher
+Teacher()
+MoveTeacher(loudPlace:int ,loudLevel:int): void

NPC
+Name: string
-Objects Held: Item*
-Is_dead: bool
-ConversationFlags: bool[]
+Talking(): Bool
+ConversationOutcomes(): void
+GiveItem(): Item*
+CheckDeath(): void

ComputerNPC
-MenuUtil: NPCScreenBuilder
+int : NumCorrect = 0
+int: Attempts
+question [4]: string
+ansarr[4][4]: string
+ComputerNPC(a:string ,b:int ,c:int)
+ConvoOut(i :int : void
+HasPassed(): bool
+HasFailed(): bool
+GetTalking(): bool
+WinCondition(): int
-ReadQuestion(s:string]: string

RoomFactory
-ClassType: int
+GetRoom(string,string,string,string,string,
         bool,string): Rooms*

InputChecker
+CurrentScreen: string
+LastMenu: string
-BaseIndent: string
-DescIndent: string
+TopLineSpace: string
+LineSpace: string
+InputChecker()
+CheckInput(int): int
+ClearScreen(): void
+PrintMenuPrompt(string): void
+PrintCurrentScreen(): void
+StringFormater(string): string
+StatusBarFormater(int ,string): string

DumbWaiter
+DumbWaiter(RoomType:int ,a:string,b:string,
            c:string,d:string,e:bool,f:string)
+GetRoomActionFlag(): bool

FurnaceRoom
+FurnaceRoom(RoomType:int ,a:string,b:string,
             c:string,d:string,e:bool,f:string)
+GetRoomActionFlag(): bool

NPCRoom
-npc: NPC*
+FurnaceRoom(RoomType:int ,a:string,b:string,
             c:string,d:string,e:bool,f:string)
+AddNPC(a:NPC*): void
+NPCName(): string
+RoomAction(): void
+~NPCRoom()
+GetRoomActionFlag(): bool
+SetRoomActionFlag(): void
+WinConditionFlag(): bool

MainScreenBuilder
-StartLogo: vector<std::string>
-StateHeaders[12]: string
-StateDescs[12]: string
+MainScreenBuilder()
+SpecialScreenBuilder(string,string): void
+GameScreenBuilder(int ,string,string,string,
                   string): void
+StartScreenBuilder(vector<std::string>): void
+PrintStartMenu(): int
+PrintPauseMenu(): int
+PrintInstructScreen(): void
+PrintMainMenu(int ,bool,string,int): int
+PrintStateScreen(int)
+PrintMoveMenu(string)
+PrintItemsToScreen(): int
+PrintDeathScreen(): void
+PrintHelpScreen(): void
+PrintPrologueScreen(): void
-StatusBarFormater(int ,string): string

Inventory
+inv: vector<Item>
+AddItem(AddItem:Item*): void
+RemoveItem(): void
+ListItems(): void
+Inventory()
+RemoveItemRoom(i:int): void
+notEmpty(): bool
+hasItem(i:Item*): bool
+getSize(): int

Item
 Name: string
+Item(Name:string)
+getItem(): string
+setType(): void

ComputerNPCScreenBuilder
+CurrentScreen: string
+LastMenu: string
+BaseIndent: string
+DescIndent: string
+TopLineSpace: string
+LineSpace: string
+CheckInput(int): int
+ClearScreen(): void
+PrintMenuPrompt(): void
+PrintCurrentScreen(): void
+StringFormater(string): string
+StatusBarFormater(int ,string): string

NPCScreenBuilder
+NPCScreenBuilder()
+NPCConvoScreenBuilder(string ,string): void
+PrintNPCDialogueToScreen(string): void

DESIGN – SEQUENCE DIAGRAMS

Interface

Hero

Room

Inventory

Interact with Object

InterFaceRoom()

ListOptions()

PickItem()

GiveItem

AddItem

AddItem

Interface

Room

NPC

StoryFlags

Interact with NPC

InterFaceRoom()

ListOptions()

Talking

Set_flags

Interface

Room

Interact with Environment

InterFaceRoom()

ListOptions()

LookAround()

CLASS DESCRIPTIONS

GameMain

Class that runs main game loop and controls interactions between player, Kivna, Rooms, NPCs, and items.

ConcreteState

A class that represents the different stages of the game. Each with an associated integer value that is incremented with the progression of the game

ComputerNPC

Specialized NPC subclass that contains the debugging quiz.

ComputerNPCScreenBuilder

A class that helps format the printing of computerNPC content to the screen

Dumbwaiter.

A room subclass that has the special functionality of being linked to other dumbwaiter objects

FurnaceRoom

A room subclass that has a special breaker Boolean that can be set on/off or received with getters and setters

Hero

A person subclass that contains an inventory which can be accessed and altered.

InputChecker

A class that validates user input depending on the number of cases for a given situation, passed as an int

Inventory

A class that holds object items. Essentially a vector with a layer of abstraction

Item

An object that represents an object in the gameworld essentially a fancy string

MainScreenBuilder

A class that controls the UI of the main game loop

NPC

A class that reads from dialogue files that increment with player interaction. Works extensively with GameMain

NPCSCreenBuilder

A class that helps format the printing of NPC dialogue

Person

A class that abstracts the locations of the player and teacher(Kivna)

RoomFactory

A class that generates rooms according to type (standard, furnace, NPC)

Rooms

A class that contains all the functionalities of rooms such as searching

Teacher.h

A class that contains the movement algorithm of the antagonist

APPENDICES

APPENDIX A: FIGURES AND TABLES

Things to note:

1. The Game is hard. You have to be very careful with your searching, and use tight windows to bait Kivna to a certain position in order to get certain objectives.
- Searching alerts Kivna to your location if he is on your floor or one floor away. He will begin following the same paths you can (save for the fact that he cannot take the dumbwaiters) to navigate to the spot in which the last search was.
- Doing anything on the middle floor is treacherous as if he gets on the middle floor, you are doomed. Acquiring any items on the middle floor must be done with thought in mind.

The start to every winning order:

    i) Talk to Anvik (right all the way, up to the top floor, and left until Anvik's lab)

    ii) Flip the breaker (right from Anvik's lab, down to the basement, and one left [search the room to access the breaker])

    iii) Get access to Nicole's room
1. Attempt to get into Nicole's room (2 rooms left of the furnace room)
2. Go to the middle floor and one left to the living room
3. Search the living room to unlock Nicole's room
4. Go back to Nicole's room and talk to her.

    iv) Go right to the storage room and acquire the Rohan duck

Here is a winning order for winning by debugging (low margin of error):

    v) In order to grab Kaelan's duck (in the kitchen), you must ensure Kivna is at least 3 moves from the staircases. You can do so by
1. Leading him to the middle floor if he's not (do this by searching on the middle floor)
   - Going downstairs/upstairs and searching in the room next to either dumbwaiter (Anvik's lab or Nicole's room)
   - This should attract Kivna to the farthest he can be from the kitchen
   - Take the dumbwaiter up/down and navigate to the kitchen to grab Kaelan's duck. You should have a small window ahead of Kivna.

    vi) Obtain the Marcus and Raj ducks in the Auxiliary and Anvik labs.
1. may have to lead Kivna away if you did step 2) in the top floor

    vii) Go talk to Nicole
1. may have to lead Kivna away if you did step 2) in the bottom floor

    viii) Get to Anvik's lab
1. Solve the debug puzzle
   - The answers are 20, 20 10, gh, and Amazonian Pygmy Owl

Here is a winning order for winning by bomb (No margin of error):

1. Die to Kivna
2. Talk to Nicole
3. Go right to the storage room and acquire the rope
4. Lead Kivna by searching in the kitchen and going downstairs
   a. No margin of error, he will be right above you when on the staircase
5. Take the dumbwaiter up and navigate right twice to Anvik's quarters. Pick up the two items
6. Move left to Anvik's lab and pick up the centrifuge
7. Take the dumbwaiter down and go talk to Nicole. Then go into the furnace room and plant the bomb
8. Go out the front door (middle floor, all the way left) in 6 turns. Do not waste any turns.
   a. Ensure that Kivna is not on the middle floor or you will die.

The 3 possible ways to lose are:

- Get caught by Kivna (twice)

- Fail the debug puzzle
- Get blown up by the bomb