# ZORK





#8 secret room contains: piles of gold

#6 vault room contains: 3 walking skeletons

#5 dining room contains: dust, empty box

#7 parlor contains: treasure chest

#4 kitchen contains: bats

#3 library contains: spiders

#1 foyer contains: dead scorpion

#2 front room contains: piano

Set current room value, Increment counter(s), tell what’s in the room , return string indicating directions (or room #???)

|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
| **Two**, User inputs choice of where they want to go | **Three**, get response … if NOT quit the go to room chosen | **One**, Start in Foyer (#1) tell user what’s in that room and which direction they can go or (Q)uit |
|  |  |  |

|  |
| --- |
| +Room |
| -roomNumber:int |
| -roomName:String |
| -roomContents:ArrayList |
| -roomDoors:ArrayList  -visitCount:int |
| +setRoomNumber(int):void |
| +setRoomName(String):void |
| +setRoomContents(String):void |
| +setRoomDoors(String):void  +setVisitCount(int):void  +getRoomNumber():int  +getRoomName():String  +getRoomContents():String (coma delimited)  +getRoomDoors():ArrayList<String>  +getVisitCount():int |

**room\_names HashMap design**

|  |  |
| --- | --- |
| **Key (int)** | **Value (String)** |
| 1 | foyer |
| 2 | front room |
| 3 | library |
| 4 | kitchen |
| 5 | dining room |
| 6 | vault |
| 7 | parlor |
| 8 | secret room |

**room\_contents HashMap Design**

|  |  |  |
| --- | --- | --- |
| **Key (int)** | **Value1 (Substring in String)** | **Value2 (Substring in String)** |
| 1 | dead scorpion |  |
| 2 | piano |  |
| 3 | spiders |  |
| 4 | bats |  |
| 5 | dust | empty box |
| 6 | 3 walking skeletons |  |
| 7 | treasure chest |  |
| 8 | piles of gold |  |

**room\_doors HashMap Design**

|  |  |  |  |
| --- | --- | --- | --- |
| **Key (int)** | **Value1 (Substring in String)** | **Value2 (Substring in String)** | **Value3 (Substring in String)** |
| 1 | N2 |  |  |
| 2 | S1 | W3 | E4 |
| 3 | E2 | N5 |  |
| 4 | W2 | N7 |  |
| 5 | S3 |  |  |
| 6 | E7 | E8 (only 25% of finding) |  |
| 7 | W6 | S4 |  |
| 8 | W6 |  |  |

Main Program

Initialize HashMap<Integer,Strings> of room\_names

Initialize HashMap<Integer,String> of room\_contents

Initialize HashMap<Integer,String> of room\_doors

Initialize all\_room\_visits = 0

Initialize Room[] rooms\_array = new Room[8]

Declare current\_room

Declare previous\_room

Initialize done = false

Declare msg\_info\_array = new String[5]

Initialize found\_secret = false

Initalize err\_msg = “”

Invoke the populate\_maps method // this assigns values to HashMaps which need to be passed

Invoke fill\_rooms method passing the HashMaps (use getters and setters) //

previous\_room = 0

current\_room = 1

REPEAT

to\_rooms(current\_room, all\_room\_visits, rooms\_array, msg\_info\_array, found\_secret)

all\_room\_visit = msg\_info\_array[3]

found\_secret = msg\_info\_array[4]

Display message “You are in the “ + msg\_info\_array[0]

Display message “This room has ” + msg\_info\_array[1]

Display message “What direction would you like to take?

Display message “These are your choices…” + msg\_info\_array[2]

Enter first letter of direction and room # (i.e. N1) ”

Display message “ or type ‘Q’ to quit

Prompt user giving them a choice of direction or ‘Q’ to quit

IF user inputs ‘Q’

THEN

done = true

ELSE

IF user inputs something other than ‘Q’

THEN

Set previous\_room to current\_room

Set current\_room to user input

err\_msg = “The direction entered is invalid. Please try again

or type Q to quit”

String valid\_string\_test = “”

String[] = valid\_array\_test = null

valid\_ string\_test = room\_doors.get(current\_room)

IF valid\_ string\_test = empty string

THEN

Display err\_msg

ELSE

IF valid\_ string\_test does NOT contain a comma

THEN

IF user input = valid\_string\_test

THEN

to\_rooms(current\_room,

all\_room\_visits, rooms\_array,

msg\_info\_array, found\_secret)

ELSE

IF user input != valid\_string\_test

THEN

Display err\_msg

END

ENDIF

ELSE

IF valid\_test\_string contains comma

THEN

Set each valid\_array\_test element

to one of the substrings in valid\_string\_test

Initialize boolean value found to false

FOR index = 1 to last element in

valid\_array\_test

IF user input = valid\_string\_test at

index

THEN

found= true

ENDIF

ENDFOR

IF found

THEN

to\_rooms(current\_room,

all\_room\_visits, rooms\_array,

msg\_info\_array, found\_secret)

ELSE

IF NOT found

THEN

Display err\_msg

ENDIF

ENDIF

ENDIF

ENDIF

ENDIF

ENDIF

ENDIF

UNTIL (done)

## populate\_hashMaps Method

populate\_hashMaps(){

room\_names.put(1,foyer)

room\_names.put(2,front room)

room\_names.put(3,library)

room\_names.put(4,kitchen)

room\_names.put(5,dining room)

room\_names.put(6,vault)

room\_names.put(7,parlor)

room\_names.put(8,secret room)

room\_contents.put(1,”dead scorpion”)

room\_contents.put(2,”piano”)

room\_contents.put(3,”spiders”)

room\_contents.put(4,”kitchen”)

room\_contents.put(5,”dust , empty box”)

room\_contents.put(6, 3 walking skeletons)

room\_contents.put(7,piles of gold)

room\_contents.put(8,treasure chest)

room\_doors.put(1,N2)

room\_doors.put(2,S1,W3,E4)

room\_doors.put(3,E2,N5)

room\_doors.put(4,W2,N7)

room\_doors.put(5,S3)

room\_doors.put(6,E7,E8)

room\_doors.put(7,W6,S4)

room\_doors.put(8,W6)

}

## fill\_rooms method

fill\_rooms(room\_names, room\_contents,room\_doors){

Initialize room\_number = 0

Initialize name\_of\_room = “”

Initialize contents\_in\_room

Initialize doors\_in\_room

Initialize visit\_count = 0

Initialize roomArrayIndx = 0;

FOR index =1 to 8

Set room\_number to index value

Set name\_of\_room to value pulled from room\_names HashMap using index value as key

Set contents\_in\_room to value pulled from room\_contents HashMap using index value

as key

Set doors\_in\_room to the value pulled from room\_doors HashMap using index value as

Key

Create an instance of a Room class

Set roomNumber value to room\_number for the instance of the Room class

Set roomName value to name\_of\_room for the instance of the Room class

Set roomContents value to contents\_in\_room for the instance of the Room class

Set roomDoors value to doors\_in\_room for the instance of the Room class

Set visitCount value to 0 for the instance of the Room class

Set rooms\_array[roomArrayIndx] to instance of the Room class

Increment roomArrayIndx by 1

ENDFOR

}

## to\_rooms method

to\_rooms() {

Increment all\_room\_visits by 1

Increment visitCount by 1

Set msg\_info\_array[0] to current room value

Set msg\_info\_array[1] to contents of the room

IF (current\_room != 6 OR found\_secret)

THEN

Set msg\_info\_array[2] to directional locations of doors leading to other rooms

ELSE

IF (current\_room == 6 && ! found\_secret)

THEN

IF visitCount X 4 < all\_room\_visits

THEN

Set msg\_info\_array[2] to directional locations of doors leading to

other rooms with the exception of the secret room

ELSE

IF visitCount X 4 >= all\_room\_visits

THEN

found\_secret = true

Set msg\_info\_array[2] to directional locations of doors

leading to other rooms

ENDIF

ENDIF

ENDIF

Return msg\_info\_array

}