** Program name:  Pizza room search – Final Project!

** Author:  Nicole Beecher

** Date:  12/04/2017

## Program Design

Create a class that contains:

1. The room
    a. 6 rooms that are derived from room
    b. 4 pointers to point to next room (up, down, left, right)
    c. Challenge function
    d. Health function
2. Gameplay
    a. Keeps track of user position
    b. Items (pizza ingredients)
    c. User (player) health
    d. Map of house
    e. Next movements

1. The room class (and derived classes)
    a. Gets name of room in the form of a string
        i. "Kitchen", "Bedroom", etc
    b. Challenge function
        i. Each room has a unique challenge
        ii. If challenge is successful, then it returns a value
        iii. This value is used to store a number in the vector of items
        iv. If challenge fails, then returns a null and the user has to try again
        v. Also, if the challenge fails, then the user loses health points (hunger points)
    c. Health depletion function
        i. Returns an integer that is used to subtract from the players health
        ii. Each room has a different value for failure
        iii. If health gets to 0 or below then user loses game
    d. SetUp, down, left, right functions
        i. For each room it sets the rooms up, down, left, or right
        ii. (if the room exists) if not then set to null
    e. GetUp, down, left, right functions
        i. Returns values
2. Gameplay class
    a. Has a room pointer called position
        i. This keeps track of which room the user is in

  - ii. Which also dictates if the user can move up, down, left, or right
- b. Has pointers for each room and sets them to each room
  - i. Each room then sets which room is up, right, down, or left
  - ii. Starting position is kitchen
- c. Has map function
  - i. Prints map and X marks the room the user is in for each movement
- d. Next move function
  - i. Asks user which way they want to move
  - ii. Validates input
  - iii. Moves if possible
  - iv. Otherwise asks for another move
- e. Main game play
  - i. Runs a while loop until user wins, loses, or quits
- f. Show items function
  - i. Prints each item in vector
  - ii. 1 = dough, 2 = sauce, etc
- g. Check health function
  - i. Checks if hunger health is 0 or below
  - ii. If so then game is terminated, and user loses game
  - iii. Otherwise allows user to continue playing

## Test Table

| Test Case | Input Values | Function | Expected Outcomes | Observed Outcomes |
|---|---|---|---|---|
| Check map and starting position | N/A | Gameplay while loop | Starting position is kitchen and X marks the spot | Starting position is kitchen and X marks the spot |
| Input next move from kitchen (up) | Input = 1 | Gameplay while loop | X moves to garage and challenge for garage is initiated | X moves to garage and challenge for garage is initiated |
| Input next move from Garage (Down) | Input = 2 | Gameplay while loop | X moves to kitchen and challenge for kitchen is initiated | X moves to kitchen and challenge for kitchen is initiated |
| Input for kitchen challenge question | Input = n | Gameplay while loop | Prints keep looking and asks for next move | Prints keep looking and asks for next move |
| Input next move from kitchen (left) | Input = 3 | Gameplay while loop | X moves to backyard and challenge for backyard is initiated | X moves to backyard and challenge for backyard is initiated |
| Input next move from backyard (right) | Input = 4 | Gameplay while loop | X moves to kitchen and challenge for kitchen is initiated | X moves to kitchen and challenge for kitchen is initiated |
| Input for kitchen challenge question | Input = y | Gameplay while loop | Prints you do not have all the ingredients and | Prints you do not have all the ingredients and |

| | | | | |
|---|---|---|---|---|
| | | | prints ingredients user currently has – asks for next move | prints ingredients user currently has – asks for next move |
| Input next move from kitchen (down) | Input = 2 | Gameplay while loop | X moves to living room and challenge for living room is initiated | X moves to living room and challenge for living room is initiated |
| Challenge in living room | Input = N | Gameplay while loop | Prints nothing happened – no item is stored in vector – health depletes | Prints nothing happened – no item is stored in vector – health depletes |
| Challenge in living room | Input = Y | Gameplay while loop | User finds cheese under roommate – cheese is stored in the vector | User finds cheese under roommate – cheese is stored in the vector |
| Input next move from living room (right) | Input = 4 | Gameplay while loop | X moves to bedroom and challenge for bedroom is initiated | X moves to bedroom and challenge for bedroom is initiated |
| Challenge in bedroom | Input = A | Gameplay while loop | Wrong choice – no item is stored in vector – health depletes | Wrong choice – no item is stored in vector – health depletes |
| Challenge in bedroom | Input = B | Gameplay while loop | Right choice – basil item is stored in vector | Right choice – basil item is stored in vector |
| Input next move from bedroom (left) | Input = 3 | Gameplay while loop | X moves to living room and challenge for living is skipped bc user already collected cheese | X moves to living room and challenge for living is skipped bc user already collected cheese |
| Input next move from living room (down) | Input = 2 | Gameplay while loop | X moves to bathroom and challenge for bathroom is initiated | X moves to bathroom and challenge for bathroom is initiated |
| Challenge in bathroom | Input = 1 | Gameplay while loop | wrong choice – no item is stored in vector – asks user to try again or quit-if quit then user health is depleted | wrong choice – no item is stored in vector – asks user to try again or quit-if quit then user health is depleted |
| Challenge in bathroom | Input = 3 | Gameplay while loop | right choice – item is stored in vector | right choice – item is stored in vector |
| Input next move from bathroom (up) | Input = 1 | Gameplay while loop | X moves to living room and challenge for living is skipped bc user already collected cheese | X moves to living room and challenge for living is skipped bc user already collected cheese |
| Input next move from living room (up) | Input = 1 | Gameplay while loop | X moves to kitchen and challenge for kitchen is initiated | X moves to kitchen and challenge for kitchen is initiated |
| Challenge in kitchen | Input = Y | Gameplay while loop | Check if all ingredients are in | Check if all ingredients are in |

| | | | | |
|---|---|---|---|---|
| | | | vector – if yes then user wins the game | vector – if yes then user wins the game |
| Quit | Input = 5 | Gameplay while loop | Game ends | Game ends |
| | | | | |

## Reflection

When I first read the assignment requirements I panicked a bit because they were so general. We had free range to basically come up with any game we wanted to. This presented a challenge at first. It was hard for me to sit down and create a game from scratch and create functions from scratch. I was used to having a bit more structure and guidance, which makes starting a project easier.

But as a I got designing I loved having more free will to design my project how I wanted to. Less rules = less headache. I drew out my rooms with which item they would have and what kind of challenge they would each have. Once I got that going everything fell into place.

My main rule for myself for this final project was to keep it simple. I knew if the program got too complicated I would lose focus and end up stressing myself out. So, I hit the basic requirements and kept things manageable.

The hardest part for me was implementing all the pointers especially the up, down, left, or right pointers. I had a hard time visualizing how to set it up. I did some research and googling and realized it was simpler than I was imagining.

Even though the final project in theory is harder because we implemented all the things we learned over the term, I would agree with the professor the first project was still harder in terms of grasping the information and jumping in feet first to pointers and larger projects.

My programming skills have grown 100 percent from CS 161 until now and I feel like a programmer now! Great course. Great projects. I enjoyed writing every game and seeing the rewards on the screen.