Jason Lim

CS 162

Project 5: Sushi Adventures Reflection

Game Description:

The objective of the game is simple. Collect the three types of sushi: Tuna, Salmon, and Unagi. You must collect all Tuna first then get to the end and put all Tuna down on the plate. Then you can get all the Salmon and put it on the plate. Then you must get the final sushi the Unagi sushi and put it on the plate. Throughout your journey you will encounter puzzles that test your physical & mental strength along with your luck. You must successfully win these puzzles or games and receive the sushis. If you are successful and win the game, then you get to return back to your village with your new sushi skills and will be able to start your own sushi restaurant!

Premise:

Sushi is a rare and scarce commodity to the land. In fact, it has been so sacred that the local government has banned sushi forever from all villages to keep to themselves. You are on an adventure to gather up sushi throughout many trials that test your physical strength, mental strength, and luck. Goal: Gather up the required sushi: tuna, salmon, and unagi, and you will be known as the sushi hero and will be able to go back to your land and start your very own Sushi restaurant at your village.

Design

1. Space Class
    a. Create 4 pointers for the final project specs for right, left, top and bottom
    b. Create an integer value for the room number to store which room we are in
    c. Create a sushitype enum to determine which sushi type is in the room
    d. Create constructor which takes room number as a parameter
    e. Set the pointer based on pointer to space, left and right
        i. Linear structure for sushi adventures game
    f. Go inside virtual function to grab sushi and put into storage
    g. Get left and get right function to get the pointer for left or right and determining which room we are in
    h. Find sushi to get sushi or find it in room
    i. Space deconstructor
2. Sushi struct
    a. Enums for Tuna, Salmon, and Unagi sushi rolls
    b. Create sushi type based on enums so that the sushi can have a value of Tuna, Salmon, or Unagi
    c. Have sushi constructor which takes value of sushi type and room number to put sushi there
3. Derived classes for Space class
    a. Tuna Room

        i. The tuna room will have a constructor for the room number or which room the tuna is in
        ii. Go inside function which will take the sushi in the room when the mini-game is complete
        iii. Mini game to match rock paper scissors with Sushi master
    b. Salmon Room
        i. The Salmon room will have a constructor for the room number or which room the salmon is in
        ii. Go inside function which will take the sushi in the room when the mini-game is complete
        iii. Random mini game to jump to cliff y/n to continue
    c. Unagi Room
        i. Final room
        ii. Has a Boolean value for if all tuna, all salmon, all unagi received
        iii. Boolean value for if win game as well
        iv. The Unagi room will have a constructor for the room number or which room the unagi is in
        v. Go inside function which will take the sushi in the room when the mini-game is complete
        vi. Mini game with sushi wizard multiplication game
4. Game class
    a. Int value for total number of steps the player has taken
    b. Sushido space for main character or pointer to space
    c. Creating storage vector which is a pointer to a list of sushis
    d. Boolean for if the game has been won or not
    e. Game constructor to create game and set all attributes
    f. playGame to actually play the game and go through all the rooms and steps
        i. Provides left or right movement capability and # steps cannot exceed 18
    g. Game destructor
5. Menu class
    a. A display menu function to display first menu
    b. chooseFromMenu function to choose the option on menu
    c. DisplayObjective function to display the Sushi Adventures objectives
6. inputValidation class
    a. Yes Or No input for choosing yes or no
    b. Integer input and integer input for sushi (3 options vs 2 options)
    c. Choose size to let user choose a integer value

Sushi Adventures Game Test Plan

Test Case	Input by User	Expected Output	Actual Output
Input Validation	Validate input for characters Y/N and integer values as well as menu	Incorrect values give error message	Incorrect values give error messages

Test ASCII art	Test ascii art to make sure it shows up in sushi adventures title and the actual sushi	Sushi adventures title and sushi ascii art show up	Sushi adventures title and sushi ascii art show up
Rock Paper Scissors Tuna Room	Use rock (1) and try to get a match	Matches with sushi master's rock and my rock, ability to proceed	Matches with sushi master's rock and my rock, ability to proceed
Can say No to a room and proceed	Type N for skipping room	Skips room	Skips room
Movement left and right of rooms	Moving right of rooms moves to the next room, moving left of the room moves to previous room	Moving right and moving left different rooms	Moving right and moving left different rooms
Jump cliff Salmon Room	Wins 50% of the time based on ready to jump	Wins 50% of the time after jump	Wins 50% of the time after jump
Math game sushi wizard Unagi Room	Math game with random values from 1-10 multiplying by the same random value	Correct value allows to proceed to get unagi	Correct value allows to proceed to get unagi
>= 18 steps	Do 18 steps or more and forces game to end	Game lost	Game lost
Able to obtain sushi	Take sushi allows us to put sushi in bag	Takes sushi and puts in bag	Takes sushi and puts in bag
Taking same sushi not allowed	Not allowed to take same sushi	Error message, sushi already taken	Error message, sushi already taken
Taking more than 3 sushi in storage	Not allowed to have more than 3 sushi in storage at same time	Error message, bag full	Error message, bag full
Win Game	Win game by collecting all tuna then putting on plate, collect all salmon put on plate, then unagi and put on plate	Wins game and gives story ending	Wins game and gives story ending
If not have the correct number of sushis	Putting sushi on plate but not all same	Message to go back and gather all same sushi	Message to go back and gather all same sushi
Valgrind test	Valgrind test through program (win game)	No mem leak	No mem leak
Valgrind test	Valgrind test through program (lose game)	No mem leak	No mem leak

Reflection Sushi Adventures

I decided to do my design as well as think of all the rooms in the game first and then proceed on working on the project. I thought that this was going to be a large project and one that was bigger than project 3 or project 4 so I needed ample time from start to finish. I started off with the basic elements and built on top of that so I created the Space class first and then thought of what derived classes should be coming from the Space class. I also had to determine what the functions of those Space class and its derived classes would be that would be inherited. The spaces would be provided so that

the player could walk through them and go into different rooms. I decided that the sushi would be placed into each room so that the player could go ahead and collect these sushi in order to place them on the plate at the end of the game and win. The rules of the game were simple, collect all tuna first then place on place, then collect all salmon (2 salmon only available in the game) then put it on the plate, and then finally beat the Sushi Wizard and put the unagi sushi on the plate. It wasn't until the end that I realized that I needed to enforce these rules somehow and put them in my instructions.

The biggest challenge was coming up with the linked list that would determine the overall map structure of the game. It was complicated to use all the four pointers for up down left and right so I just decided to go with a linear structure to simplify things. Because of this design choice, I was able to avoid top and bottom pointers that would overall increase the difficulty of the game since you must navigate more rooms and it was hard to see the linked list that way.

There were issues in the description of my game as well as some parts where my mini-games would continuously loop or not make sense. I had to fix these bugs by writing clearer descriptions and looping only when needed. I also had to fix a design choice where I initially chose too few total steps to be taken until the game ended causing the game to end halfway through. I did valgrind testing to make sure there were no memory leaks. There was no memory leak when you win the game and also no memory leaks when you lose the game. I also made sure that the ascii art fit onto my screen and was good enough to present. I had to make sure to choose the right ascii art for my sushi and my title Sushi adventures.

I created the Sushi class and decided on building enums for the types of sushi that my game would have. Instead of using an existing string or other data type, I decided to create my own.

In my Game class, I had game constructor construct all the variables and items needed for the game to start. Then I had a play game function which would play the game and would allow the user movement across the game's rooms. There was also a Boolean I decided to implement so that we knew when the game was won.

Finally, I added all the dialogue and rules of the game as well as the premise at the beginning and situations when we lose or win the game. The story had a lot of storytelling and as I played the game, it made sense all the mini-games and the plot and it helped me figure out what to do next. I hope this game is easy to play and fun for my audience!