CS 111 Final Project Self Assessment

Group

Who's in your group?

1. Lauren Bichelmeir
2. Danche Smilkova
3. Jessica Lechuga

Goals

Say a few words about what you wanted the game to be like. Note that if you just wanted to write some code so you could get a good grade on the project, it's fine to admit that.

We wanted to create an Indiana Jones-type of explorer adventure game in which players travel to different environments, converse with mythical creatures and interact with the world around them. The end goal isn't necessarily to win, but to investigate the items around them with a curious mind, look for clues based on the environment and just have fun meandering through all our distinct environments. However- as in any game - trying to win might give a better understanding of what's happening! (We also wanted to write some code to get a good grade on the project, but it was a fun process.)

Lessons learned

What went right?

Our team worked really well together as a group. Despite our 3 different time zones (we each had a 7 hour time difference relative to another: we had one person in Asia, one person in Europe and one person in America), we were able to meet, discuss our code, and then leave the meeting knowing that one person just spent the entire day working on the code and the other person's day was just starting, which meant that they could begin the code with a fresh perspective. We were able to have a clear division of labor by having each person prioritize certain environments, yet we still made sure to collaborate and bounce ideas off each other to make sure the entire code was functioning properly. We are very proud that we were able to figure out how to create and destroy new things within procedures by defining the location with (here) or me as we couldn't figure out for the longest time what location to put for the new-thing we were calling.

What went wrong?

Nothing necessarily went wrong. Sharing our code changes with each other and then updating each of our own files was the most difficult part, especially as more and more changes were created each time.

What do you wish you knew when you started?

We wish that we knew how to create and destroy new things within procedures instead of just using new-props. We also wish we knew our limits in regards to what we could do with the game as the instructions/guidelines were a bit vague and we didn't know at the beginning if our idea was feasible. We wish that we had more structure to refer to and more examples of realistic ideas. We have a lot of procedures that rely on the same logic and structure and we wish we knew earlier whether that was okay. We also wish that we knew earlier how to use Github to share code as it was hard to keep track of it.

Annoying grading bookkeeping

Types

What are the types you added, and what are they for?

1. **(define-struct (creature person)**
   We created a creature subtype of person to serve as our npc characters in the game. We created a mermaid, sorceress, and fairy subtype of creature so that we could have some distinct mythical creatures in the game.
2. **(define-struct (sorceress creature)**
   You are able to communicate with the sorceress for hints, you can also fight with the sorceress. The sorceress can cast spells that weaken the player. The player must kill the sorceress or die trying to end the fight. If the player wins the fight, the sorceress produces a orb
3. **(define-struct (mermaid creature) and (define-struct (fairy creature)**
   The mermaid and fairy are able to communicate with the player and provide useful hints.
4. **(define-struct (orb creature)**
   The orb can talk and increase the player's strength when rubbed.
5. **(define-struct ability (name manacost effects description)**
   The ability type is a new type needed in order to fight the sorceress.
6. **(define-struct (enemy thing)**
   We created an enemy subtype of thing to serve as our enemy objects in the game. This type holds a field to keep track of the 'disenergy' or chaos created by the enemy, which would decrease the strength of the player upon interaction. Cactus and Lava are both subtypes of Enemy.
7. **(define-struct (cactus enemy)**
   The player is able to either touch the cactus which decreases their strength or cut the cactus in order to obtain some water.
8. **(define-struct (lava enemy)**

   Touching the lava severely decreases the player's strength but the player can use a water bucket to destroy the lava

9. **(define-struct (water thing)**

   Water is necessary to fill the bucket and important for the production of oxygen in the game.

10. **(define-struct (oxygen water)**

    Oxygen appears after the procedure (dissociate thing) is called with a specific water and it is used to fill the oxygen tank in order to climb the mountain.

11. **(define-struct (bucket thing)**

    The bucket type was created so that the user could interact with the bucket by filling it up with water. This bucket is also used to destroy the lava (below).

12. **(define-struct (oxygen-tank thing)**

    The oxygen-tank is used as a necessary requirement to be able to climb Mt. Everest mountain. The oxygen-tank can be filled up with oxygen.

13. **(define-struct (leaf thing)**

    The leaf type is created to serve as an inconspicuous flag to place at the top of the mountain after climbing. Once this is done, the game is finished.

14. **(define-struct (mountain thing)**

15. The mountain was created as the last obstacle of the game. It can be climbed but only when certain conditions are met.

16. **(define-struct (shell thing)**

    The shell produces a new bracelet object after two calls are made to the shell, wipe and sing-to!.

17. **(define-struct (good thing)**
    The good thing subtypes are items that increases the players strength

18. **(define-struct (bracelet good)**
    The bracelet appears after the shell has been sung to. It increases the strength of the player if worn.

19. **(define-struct (flask good)**

    The flask type is used to create a 'potion of invincibility' using the mushroom ingredient. Drinking the flask/potion increases the player's strength

20. **(define-struct (mushroom thing)**

    A mushroom type was created to serve as an ingredient for the flask to make a potion.

21. **(define-struct (food thing)**

    Food serves as an energy booster when eaten.

22. **(define-struct (coconut food) and (define-struct (berry food)**

    Both the coconut and the berry boost the energy field by their own pre-determined specified amount when consumed.

23. **(define-struct (room environment)**

    This is a subtype of environment and is for smaller or more specific spaces within environments.

Fields

What are the fields you added, what types did you add them to, and what are they for?

1. **Fields of person subtype: (define-struct (person thing) (energy oxygen strength)**

a. **energy**: describes the energy condition of the person, a parameter needed to survive/win the game

b. **strength**: describes the strength condition of the person, a parameter needed to survive/win the game

c. **oxygen**: describes the oxygen levels of the person, a parameter needed to climb the mountain and survive/win the game

2. **energy-value**
   (added to food type): Keeps track of the energy values of the food items
3. **disenergy**
   (added to enemy type): Keeps track of the disenergy (or 'chaos') values of the enemy types that will decrease the player's strength.
4. **power**

   (added to good type): Used to increase the strength of the player with certain "things" in the game.

5. **song**

   (added to shell type): The shell can be sung to (sing-to! thing song) using its specified song in the song field: "The Shell Song"

6. **ingredient**

   (added to flask type): to make a potion (craft-with flask thing), the flask requires the specified ingredient in its ingredient field. (craft-with flask thing) checks to see if we have the ingredient before creating the potion

   in start-game: (new-flask (new-mushroom forest-env) forest-env)

7. **water**

   (added to bucket type): the bucket can be filled with water using the bucket-water field. (fill bucket thing) checks if the player has the specified water in the bucket-water field before filling up the bucket

   in stat-game: (new-bucket (new-water '("sea") sea-env) volcano-env)

8. **leaves**

   (added to mountain type): one of the requirements for climbing the mountain is having a flag/leaf (climb mountain) checks if the player has the specified flag in the mountain-leaves field

   in start-game: (new-mountain (new-leaf forest-env) mountain-env)

9. **oxygen-value**

   (added to oxygen-tank type): Used to change the oxygen levels of the person.

10. **name**

    (added to ability type): the name of the specified ability

11. **manacost**

(added to ability type): the cost of an ability/spell, specified in mana; used for fighting the sorceress

12. **effects**

    (added to ability type): specifies how each spell affects the opponent (by how much is the health of the sorceress decreased)

13. **description**

    (added to ability type): description of how the attack is conducted

14. **hp**

    (added to sorceress type): describes the health condition of the sorceress, a parameter needed to survive/win the fight

15. **abilities**

    (added to sorceress type): keeps track of the abilities of the sorceress

16. **mana**

    (added to sorceress type): resource you need to cast abilities

Procedures

What are the procedures you added or significantly modified from their original form, and what are they for?

1. **(dissociate thing)**

   Takes water and dissociates the molecules to create the oxygen needed to increase oxygen levels.

2. **(sing-to! thing song)**

   Sings the song you specify to the shell.

3. **(wear! thing)**

   Allows the player to wear the bracelet which will give them strength.

4. **(take-it-off!)**

   Takes the bracelet off and decreases the player's energy.

5. **(is-a-food? thing)**

   checks if the specified input is a food from a given list of different types of food in the game

6. **(all-food-list)**

   Procedure with the list of things that are food.

7. **(eat thing)**

   Procedure that eats the food, boosts your energy, and destroys the given input if it is a food.

8. **(all-enemy-list)**

   Contains list of enemy structs

9. **(is-not-an-enemy? thing)**

   Checks if the specified input is an enemy from a given list of enemies in the game.

10. **(touch thing)**

    Procedure that goes to void if the input is not an enemy, otherwise it decreases the strength of the player.

11. **(process-spell name)**

    Emulates a dictionary, converting an ability name string into an ability object.

12. **(prompt-spell target)**

    Prompts the player to cast an ability. After the user inputs the ability name, it will cast it on the target.

13. **(initialize-sorceress! enemy)**

    Wrapper which calls the parent initializer.

14. **(handle-sorc-combat-victory sorc)**

    Called upon winning against the sorceress. Dumps the orb in the player's inventory.

15. **(attack! sorc)**

    Initializes combat with the sorceress. Recursive. Guarantees all conditions for combat hold true (sorceress health is sufficient, sorceress ability resource is sufficient and the player has enough power). Exits combat when one of these conditions is broken.

16. **(player-reduce-health-by! value)**

    Reduces the power of the player by a value.

Initialization Procedures

All of the following procedures initialize the specified type.

17. **(new-coconut location)**
18. **(new-berry location)**
19. **(new-water adjectives location)**
20. **(new-cactus location)**
21. **(new-lava location)**
22. **(new-bracelet location)**
23. **(new-shell location)**
24. **(new-mushroom location)**
25. **(new-flask ingredient location)**
26. **(new-bucket water location)**

27. **(new-oxygen-tank location)**
28. **(new-oxygen location)**
29. **(new-leaf location)**
30. **(new-mountain leaves location)**
31. **(new-orb vocab adjectives location)**
32. **(new-ability name costs effects desc)**
33. **(new-mermaid vocab adjectives location)**
34. **(new-fairy vocab adjectives location)**
35. **(new-sorceress adjectives location vocab abilities mana)**
36. **(new-room adjectives)**

Methods

What are the methods you added or significantly modified from their original form, what types were they added to, and what are they for? Note that if you have three different methods for the same generic procedure, list each one separately.

Person Type Methods

1. **(change-energy who what)**
   Changes the energy of the player after eating food
2. **(change-oxygen who what)**
   Changes the oxygen level of the player
3. **(increase-strength who what)**
   Increases the strength of the player when interacting with the "good" objects
4. **(decrease-strength who what)**
   Decreases the strength of the player when interacting with the "enemy" objects
5. **(energy-oxygen-strength who)**
   Procedure that checks the current state of the person and prints it
6. **(go thing)**
   Method that prevents the player from "going" into objects

7. **(cut cactus)**
   Allows you to cut the cactus
8. **(destroy lava thing)**
   Allows the player to destroy the lava with a water bucket
9. **(wipe shell)**
   Lets the player wipe the dirty shell to reveal a message
10. **(examine shell)**
    Allows user to examine the shell

11. **(examine water)**
    Allows user to examine the water
12. **(rub orb)**
    Allows the player to rub the orb, increasing the player's strength. The orb is then destroyed
13. **(cast! abi person caster)**
    Handles casting an ability by a caster onto a person. Applies the effects of the ability on the target and outputs what occurred.
14. **(apply-effects! abi person)**
    Applies all of the effects of an ability onto a person
15. **(dec-hp! sorc amount)**
    Decreases the health of the sorceress by an amount
16. **(let-cast! enemy target)**
    Handles the sorceress casting onto a target. Makes sure the sorceress has sufficient resources to cast and casts a random ability out of the abilities known by the sorceress. Finally, it subtracts the sorceress' current spell resource by the appropriate value
17. **(craft-with flask thing)**
    Allows the player to create a potion with the flask and one other necessary thing and destroys the thing after the potion has been created.
18. **(drink flask)**
    Allows the player to drink the potion in the flask. After drinking the flask, the player's strength increases and the flask is destroyed
19. **(fill bucket thing)**
    Allows the player to fill the bucket up with sea water.
20. **(fill-up oxygen-tank thing)**
    Allows the player to fill up their oxygen tank with oxygen before they are allowed to climb Mt.Everest
21. **(plant leaf)**
    Allows the leaf to be planted as a flag on the top of the mountain.
22. **(climb mountain)**
    Allows the player to climb to the top of Mt.Everest only if they have the leaf to plant and sufficient oxygen.
23. **(prepare-to-remove! container thing)**
    We created immovable things in the game by calling the things with the adjective "static" or checking if the input is a specific type and adding an if statement in these procedures to create an error if the user tried to move the thing.

Total stuff we built

Write the total number of items listed above.

Types: 23

Fields: 18

Procedures/Methods: 59

Total stuff we built: **100** (which is why we deserve an 100 ;))