FluffyTurtles
Augie Murphy, Maximilian Korsun, Arpita Nag
Period 3
Choose your own adventure

CYOA

Our Idea:

In its simplest form, this choose your own adventure is in line with games such as monkey island - a cross between a puzzler and an RPG with branching story paths.

**** How it works: ****

The user interacts with various elements in the world to complete the mission. They will be given quests and will need to interact with the environment. The world has different people who will give out (~ 2 main quests) and (~ 5 max side quests), ranging from retrieving an item to solving a puzzle/playing a minigame. The user will find different objects to help in the quest. However, be aware - the way you interact with the world can affect your ending (branching paths). *We will make a backstory*

Our Implementation:

**** How to Play: ****

1. The game starts off with a ***backstory*** to set the scene. Once the user has seen it, they're instructed to ***click*** anywhere to view the next screen. (for MVP, just an image and the word "world")
2. Next, we see the starting area. The player starts off with an empty **inventory**, and a brief ***tutorial*** explains the features of the game and how the user interacts with the world, and guides the user to the first quest.(tutorial will be in this intro sequence - jut shows a quick how to play the game with text)
3. Clicking on the quest or any area of the world will bring up some sort of description(the description may be implemented later/not at all, it depends) and 2 options, thus leading to the "branching" pathways and different outcomes.
4. If there is another ***person*** to interact with, they may initiate a sidequest or continue on the quest.
    a. If they continue the quest, it will pop the current message and lead you to the next one.
    b. If they initiate a side quest, it will pop onto the stack. If you talk to them again, it will be removed. The way you interact with people can also change the way your story branches, even if you do not realize it.

**** Possible Design: ****

1. **CYOA** is the driver file of our game. It's ***setup*** by going through a backstory screen and a brief tutorial. It contains a **world**, a **player**, and ends when the **quest** is solved(or something terrible happens to the player). Each draw refreshes the screen displayed--which shifts when a **Feature<***Examinable***>** is ***examined***.
2. The **World** is the setting of the game, it contains many uniquely designed instances of **Feature<***Examinable***>** , such as Ocean and person. Each of these has its own unique interaction choices, and some may only work if you havve a certain item on you.
3. Each **Player** *has-a* name and an **inventory**, which holds the **tools** they can use to interact with other **features**. They ***click*** on **features** to ***examine*** them and their objective is to ***finish the quest.*** If the feature that's clicked is not *Examinable*, nothing happens.
4. The *Examinable* Interface is implemented by all **features** of the **world** relevant to finishing the quest. This distinguishes embellishments in display and design from clues and forces the user to really examine the situation.[1] All implementing classes must provide implementations of ***examine***. This makes displaying them easier and allows each feature to specify its own abilities and secrets.
5. The **Inventory** can utilize our study of data structures this year. It holds the **tools** that the **player** examines. It can be ***displayed***.
6. Each **Tool** is-a **Feature<***Examinable***>** that is stored in the **Inventory** and has a specific function. For example, a lock will only open if the correct key is used.
7. Each **Feature<***Examinable***>** *has-a* description and provides an implementation for ***examine***. It can be ***displayed***. It can be ***solved*** by using the right tool, clicking on a secret compartment, or typing in the correct password (depending on the feature, each one is unique). The puzzles contained within these can use binary trees, stacks, and other cool data structures!

Additions for greenlighting

The entire world and the cuurent placement of the character will be stored in sperate trees. One will be a tree system of keeing track of location, and the other tree will keep track of which story option the player has made - different actions have different consequences.

The questing system will run off a stack. When the stack is depleted or the player has met an unfortunate end, the game is over.

---

[1] If we have time, we could have the mouse change shape when it passes over a **Feature<***Examinable***>**.