

A dark gray rectangular frame with a thin white border. Inside the frame, the words "Escape Room" are centered in a large, bold, dark blue font. The text is set against a white rectangular background. In the four corners of the dark gray frame, there are white geometric shapes: a diagonal line in the top-left, a circle in the top-right, a circle in the bottom-left, and a triangle in the bottom-right.

Escape Room

Samantha Chu, Elizabeth Zimmerman, Hong Doan

A virtual escape room!

What did we build?

- Build-off of Peer-to-Peer Chat
 - Messaging between players
 - Sending data (unbeknownst to players)
 - Added interactive minigames

Cut to live demo!

Challenges

As C is a procedural language, it is hard for different parts of the program to communicate with each other.

How does Player 1 know the status of Player 2 in the game?

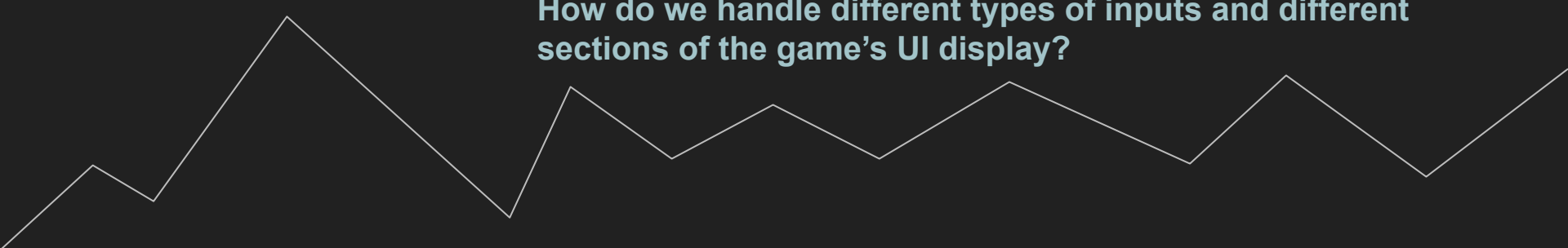
How can we have a set up where Player 1 can only progress through the game if Player 2 reaches a certain point?

How can 2 players collaboratively play a game?

How do the player programs signal the UI to run a game?

How can one input from Player 1 can change the status of the game UI, the narration of Player 1, and the progress of Player 2 synchronously?

How do we handle different types of inputs and different sections of the game's UI display?



We address all these challenges with 3 concepts from CSC-213

Parallelism with Threads

We have so many threads...



Thread Synchronization

We have so many locks...



Networks and Distributed Systems

This one is kind of obvious...

How does it work?

Biggest Challenge: Octopus

(General) Implementation Details

- Reading from files (back to CSC-161!)
- Using forms from ncurses
- Send messages in a struct (username & message)
- Threads:
 - Narrative thread (very important!)
 - 3 continuous threads:
receive_message, timer, and
boss_attack

How does it work?

Third Minigame: Solving a box/Anagram game



Thank you!