Code Break Lockdown:

An Educational Computer Science Game

Gretta Foxx	*Computer Science*
Yaxin Gu	*Computer Science*
Sarah Ogorzaly	*Computer Science*
Sidney Zweifel	*Computer Science*
Professor Fiske	*Faculty Advisor*

Outline

1. Executive Summary
2. Background
3. Problem Statement
4. Objectives
5. Technical Approach
6. Deliverables
7. Timeline
8. Budget
9. Professional Awareness
10. Scene Manager
11. Level 1 Design
12. Level 2 Design
13. Level 3 Design
14. Level 4 Design
15. Future Work
16. Conclusion

Executive Summary

- Escape room structured gameplay
- Reinforces CIS course content
- Levels based on different concepts
  - Level 1: variables, Level 2: if-else, etc.
- Goal
  - Adaptable and efficient

Source: *A. Zhukov, Behance.net*

Background

- Video games work similarly to how people learn
  - Cycle of learning

- Medium of education video games
  - Experience specially curated toward teaching the player

Problem Statement

- Importance of CS fundamentals
- Many CSU students interested in game development
- Allow CSU to learn, create, and teach

https://www.csuohio.edu/

Objectives

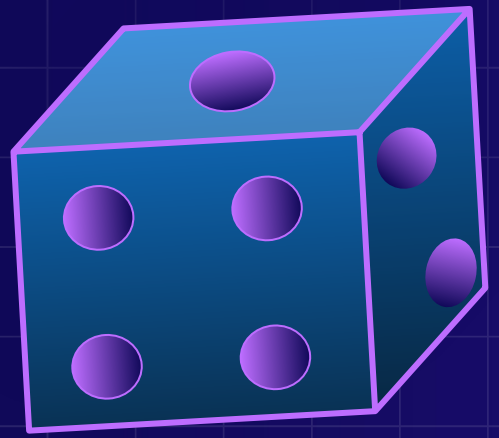
- PC game made with Godot
  - Resource available to students
- Cohesive documentation
  - Make scalability possible
- Replayable and efficient gameplay
  - Understanding > memorization
- Strong team collaboration
  - S.T.E.A.M.

Source: *https://godotengine.org/*

Technical Approach

Identifying Unmet Needs

- Engaging learning experience
- Practice and reinforcement
- Individualized learning
- Accessibility to CSU



Technical Approach

Determining Design Constraints

- Limited experience with Godot
- Lack of version control software
  - GitHub repository
- Implement randomization
  - JSON
  - Godot Random Number Generators

https://github.com/

Technical Approach

Defining Technical Specifications

- Create our game using the Godot game engine
- Create “escape room”-esque levels that provide Comp-Sci learning experiences
- Levels each have some aspect of replayability
- Include documentation for how to create new levels

Technical Approach

Enumerating Design Concepts

- Have one group member tackle one Comp-Sci concept per level
- Each puzzle has some aspect of replayability
- Each puzzle has hints to help the player
- Gradually implement more advanced concepts as the levels progress

Technical Approach

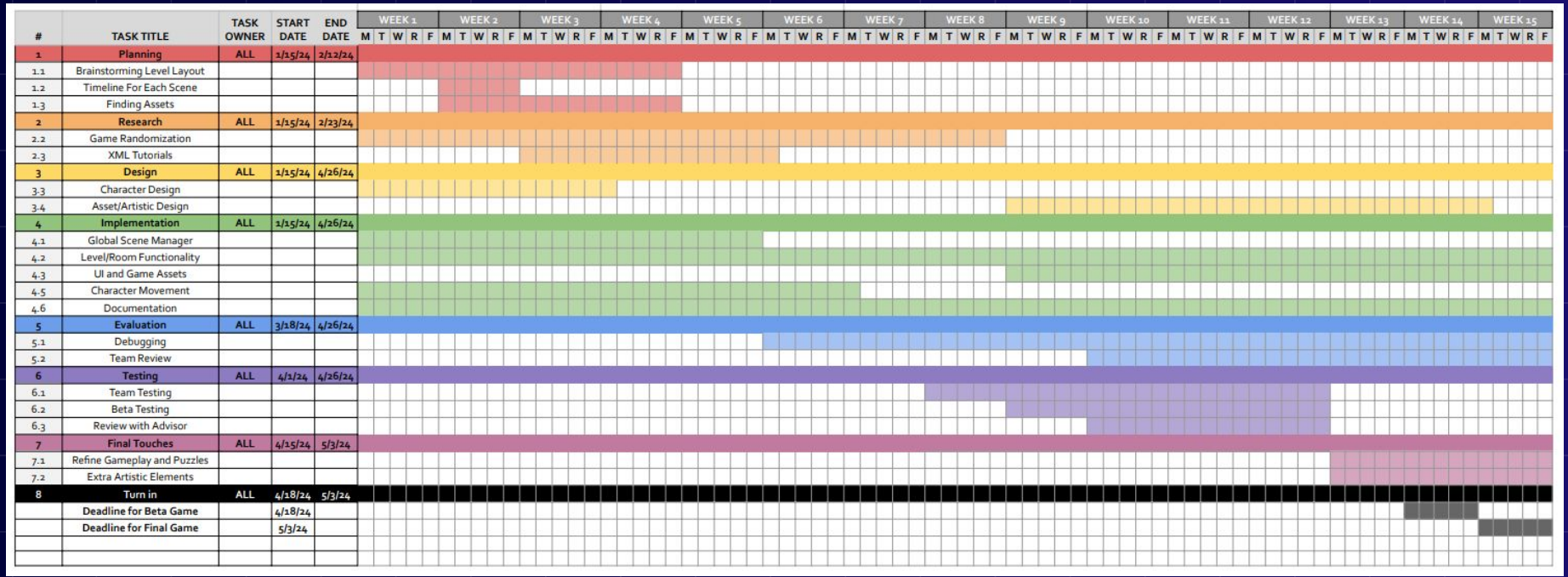
Standards Compliance

- ISO/IEC 25010
  - Functionality, performance, compatibility, usability, security, maintainability, portability
- Copyright law
  - Protect original works of authorship

Deliverables

- Incremental progress showcased in weekly meetings and pushed to group members' personal GitHub branch
- A reference document that includes links and descriptions for each resource used
- Comprehensive document detailing how to create new levels
- Fully functional and polished game

Spring 2024 Timeline



#	TASK TITLE	TASK OWNER	START DATE	END DATE
1	**Planning**	**ALL**	1/15/24	2/12/24
1.1	Brainstorming Level Layout			
1.2	Timeline For Each Scene			
1.3	Finding Assets			
2	**Research**	**ALL**	1/15/24	2/23/24
2.2	Game Randomization			
2.3	XML Tutorials			
3	**Design**	**ALL**	1/15/24	4/26/24
3.3	Character Design			
3.4	Asset/Artistic Design			
4	**Implementation**	**ALL**	1/15/24	4/26/24
4.1	Global Scene Manager			
4.2	Level/Room Functionality			
4.3	UI and Game Assets			
4.5	Character Movement			
4.6	Documentation			
5	**Evaluation**	**ALL**	3/18/24	4/26/24
5.1	Debugging			
5.2	Team Review			
6	**Testing**	**ALL**	4/1/24	4/26/24
6.1	Team Testing			
6.2	Beta Testing			
6.3	Review with Advisor			
7	**Final Touches**	**ALL**	4/15/24	5/3/24
7.1	Refine Gameplay and Puzzles			
7.2	Extra Artistic Elements			
8	**Turn in**	**ALL**	4/18/24	5/3/24
	Deadline for Beta Game		4/18/24	
	Deadline for Final Game		5/3/24	

Budget

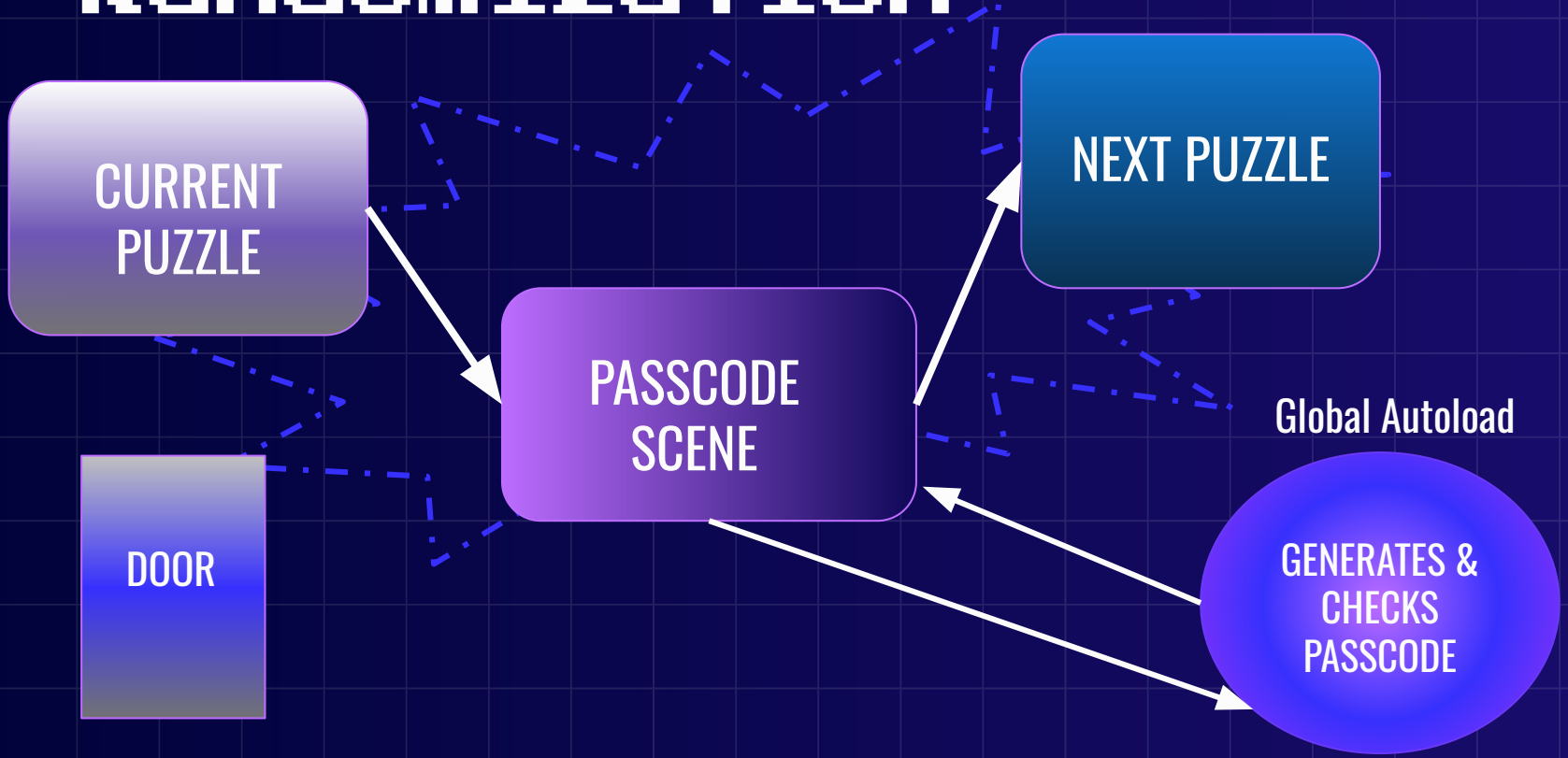
- Open-source or original assets only used

Item	Supplier	Quantity	Unit Price	Total
Game Assets	Godot Marketplace	1	~$150	**$150**

Professional Awareness

- Source or references
- Educational game must provide accurate information
- User understands that documentation serves as a supplement to various other game development and computer science resources
- Lifelong learning

Scene Manager & Global Randomization

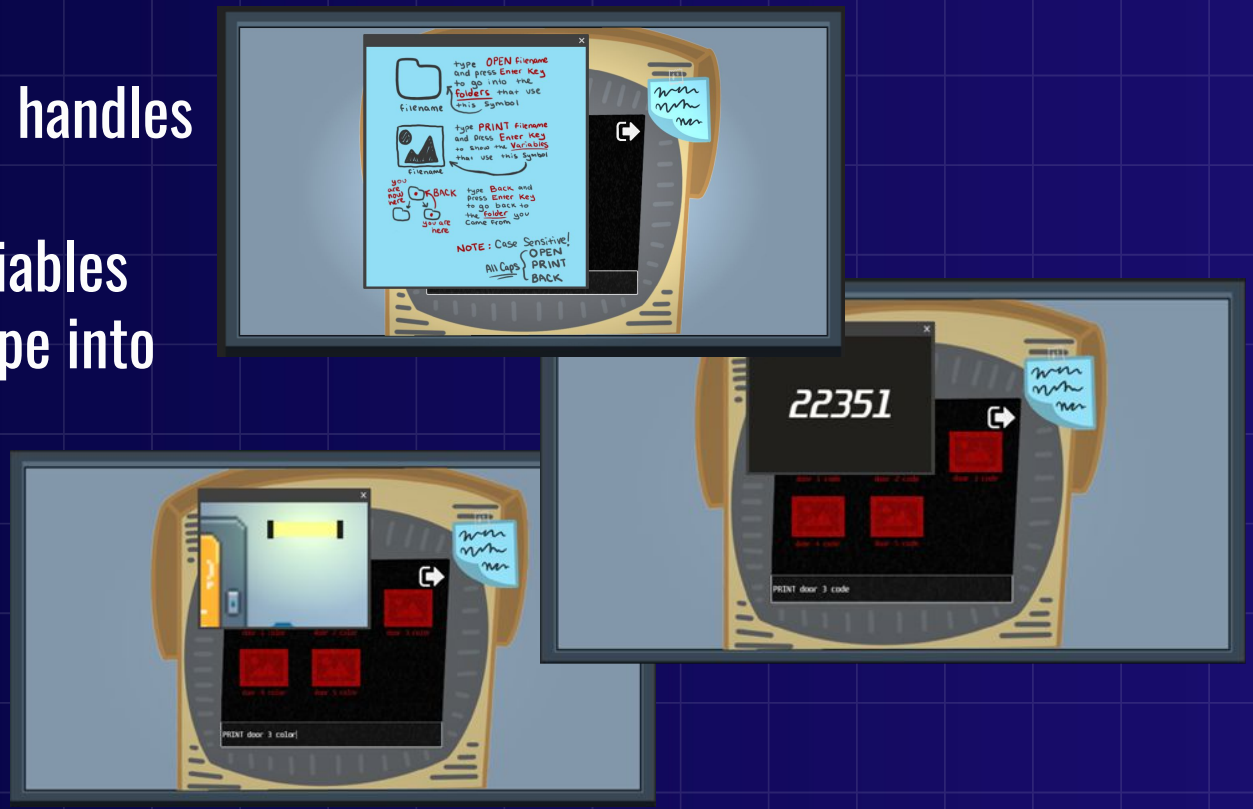


Level 1 Design

- Variable printing & traversing through a mock file system
- Global Autoload
  - Door codes generation
  - Door light generation
  - Door number/light pairs
- Door light randomly generated each time level is played
- Door number and code also different each time

Level 1 Design

- Local
  - Terminal system handles user input
  - Displays the variables and files they type into the terminal



Level 2 Design

- If-Else Puzzle
- Global Autoload
  - Generate passcode
- Local
  - Display passcode in conditional statements

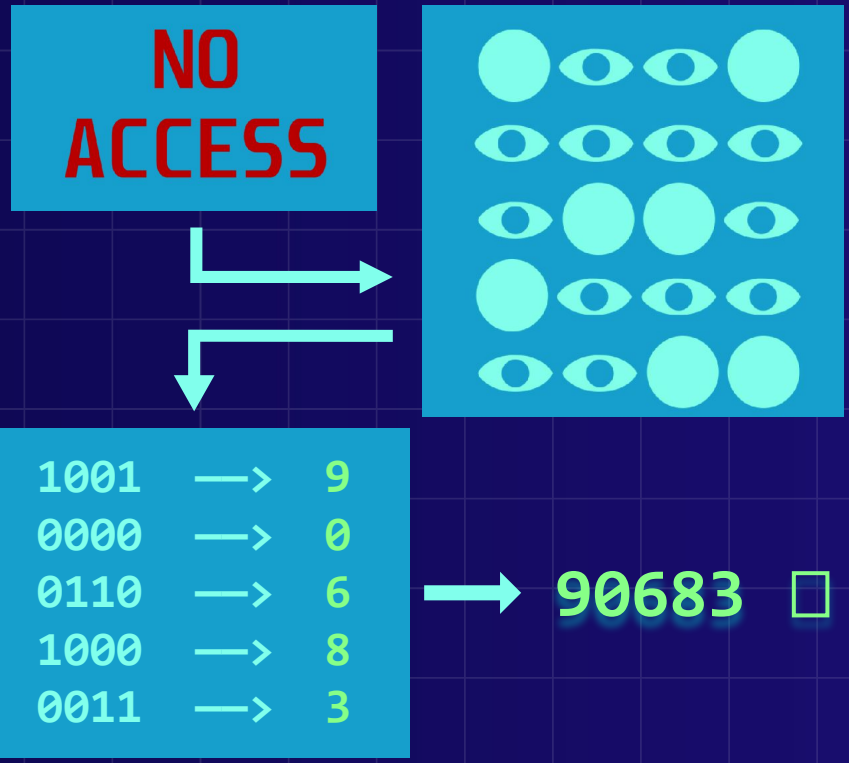
Level 3 Design

- Bitwise Operation Puzzle
- Global Autoload
  - Problem generation
  - Solving operation
- Local
  - Handles user input
  - Compare to global


```
if get_logic_question() == solve_logic_question()
    display_binary_puzzle()
```

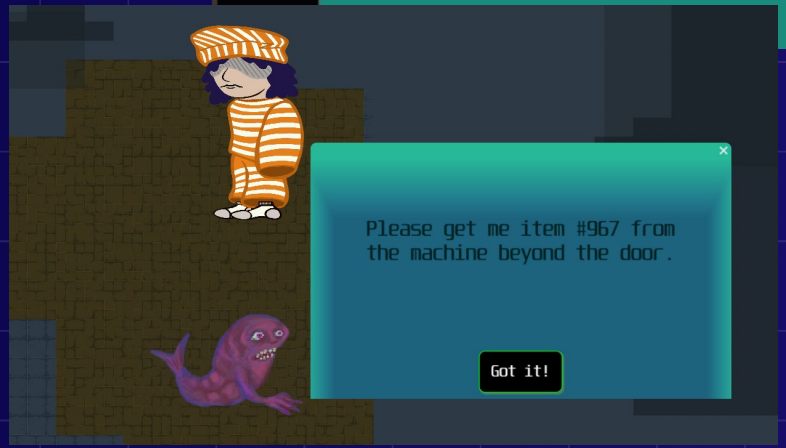
Level 3 Design

- Binary Conversion Puzzle
- Global Autoload
  - 5 digit-code generation
- Local
  - Converts code to binary
  - Binary in symbols



Level 4 Design

- Loops Puzzle
- Local
  - Instanstianes inmates
  - Instantiates door for each inmate
- Puzzle 4 Autoload
  - Saves user entered info
  - Controls inmate spawning



Level 4 Design

- Loops Puzzle
- Local
  - Controls puzzle generation
- Puzzle 4 Autoload
  - Loop counter
  - User-entered indexes
- Global Autoload
  - Override passcode generation

What Could Have Been Done Differently?

- Two main pitfalls were productivity and performance
- Productivity
  - Collaboratively developing reusable assets
  - Spent more time learning GitHub
- Performance
  - Prioritize the performance of the game
  - Enforce best practices when implementing game elements

Conclusion

- Comp-Sci can be hard, so it is good to have a resource that can teach it in a fun way
- A game lets students be eased into hard concepts
- We want inspire people to stick with Comp-Sci

QUESTIONS?