Oliver Collins-Cope

2102775@rutc.ac.uk

Learning Aim B and C

Design a computer game to meet client requirements and develop a computer game to meet client requirements

unit 8 computer games devleopment

Assignment 2

Contents

[Introduction 2](#_Toc121733421)

[Purpose of the game 2](#_Toc121733422)

[Target audience 2](#_Toc121733423)

[Requirements 2](#_Toc121733424)

[Programming languages 2](#_Toc121733425)

[Intended platform for delivery 2](#_Toc121733426)

[Game design 2](#_Toc121733427)

[Data dictionary 2](#_Toc121733428)

[Pseudocode 3](#_Toc121733429)

[Storyboard 4](#_Toc121733430)

[Visual styles 4](#_Toc121733431)

[Full motion video 4](#_Toc121733432)

[Assets 4](#_Toc121733433)

[Gameplay features 4](#_Toc121733434)

[External feedback and review 4](#_Toc121733435)

[Justification 4](#_Toc121733436)

[Evaluation of design against client requirements 4](#_Toc121733437)

# Introduction

## Purpose of the game

The purpose of my puzzle game is to entertain my players and to help train their cognitive skills through different puzzles of varying difficulty. They will incrementally get harder and help to identify the players skill level, therefore showing them where they need to improve on themselves.

## Target audience

The target audience for my game is 15+ years old. This is because I do not believe that the game will be interesting for anyone younger than this, and that they also might not have developed the life skills and problem-solving ability to solve the puzzles without outside support, which obviously defeats the purpose of testing the player and encouraging them to improve their skills. Furthermore, it might limit the type of puzzles I can make if the target audience is too young.

## Requirements

The hardware requirements to run my game include 8 GB of RAM on the computer, an i5 intel core. Furthermore, the computer game will be based on WASD input, and a mouse to look around. This means that the player will need both a mouse and keyboard for input in order to play. A monitor will be needed to view the game screen and if the player wants, they can use headphones or speakers for audio.

The software for the game will be the actual download of the game, as well as any additional software required, possibly such as .NET and there is also a need to be running the correct OS.

## Programming languages

The programming language used to develop will be C#, as that is the programming language of Unity and therefore will be used for development.

## Intended platform for delivery

The intended platform for delivery of my game can be adjusted depending on what is needed, however it will initially be developed for windows operating system on PC. It can be built for any OS on PC, however the game will need separate work in the future for it to work on console and other platforms.

# Game design

## Data dictionary

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Data name | | Data type | Scope | Size | Example | Validation |
| Score | Integer | | Class | 4 bytes | private int score; | score >=0 |
| Lives | Float | | Class | 4 bytes | private float lives; | lives >= 0 |
| playerName | String | | Class | Dynamic | private string playerName; | playerName != PLAYERNAME |
|  |  | |  |  |  |  |
|  |  | |  |  |  |  |

## Pseudocode

For every game there is a need to intricately design as much as is needed, and to an extent that means writing out the code for the different scripts that will be used inside of the game. As there are many different features within a game, from a character that is able to move around, to a light switch turning off and on, it allows for lots of room to develop pseudocode in order to smoothly integrate your design into development.

The most important feature of my game will be the ability to interact with the environment, and therefore it is important to ensure that the character has a way to detect what they are looking at and whether they can interact with that item, while the item they are interacting with will decide what the interact action does.

The pseudocode below displays how the character detects if they are looking at something:

make Ray

if Ray Hit == TRUE Then

if Ray Hit Interact == TRUE Then

if Interact Button == TRUE Then

Interact Action

This is a very basic outline, aka pseudocode, of how the game will check if the player is looking at something. This is the interaction pseudocode and determines how the interacted with object will act.

Function Interact Action

Object (I.e. box, door, light switch)

Function Door

If Key Active == TRUE Then

Door Unlocked == True

If Door Open == False AND Door Unlocked == TRUE Then

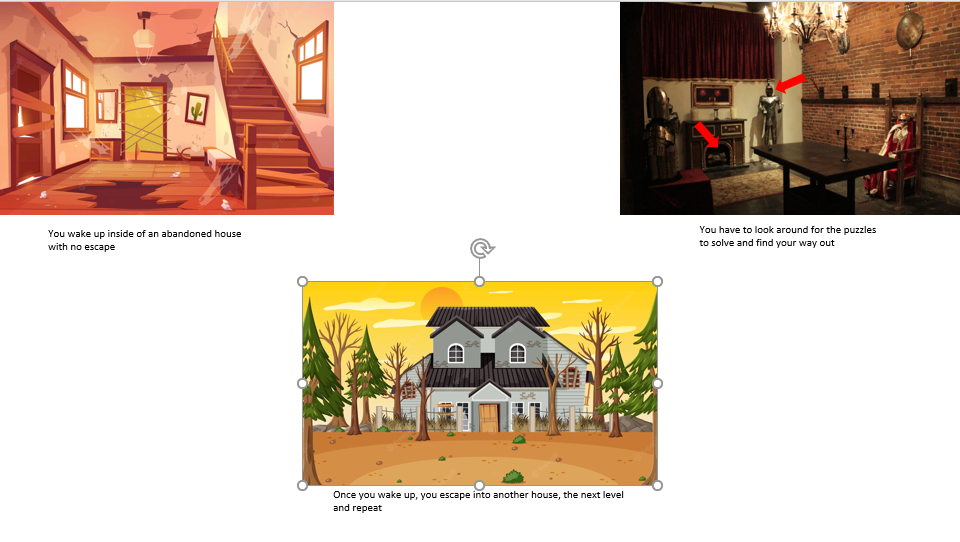
Play Animation Open Door

Else if Door Open == TRUE AND Door Unlocked == TRUE Then

Play Animation Close Door

This is another very basic version of the code that will be used for the doors in the game and will be expanded on in development.

## Storyboard



While extremely vague, the storyboard gives a basic insight into the game and how it will be play out and the general style. The game will have a few different levels and offer a score and life system based on completing and failing the puzzle respectively. Finally, the story board also helps to understand a little about how the game takes place, however as story is not a focus of my game it will not be covered deeply both inside and outside of development.

## Visual styles

The visual styles in the game will help to bring it to life and create depth for the player to explore and enjoy it. The world will be in 3D, with most of the textures being plain white or other solid colours, while interactable objects will be a different colour in order to help differentiate the differences between what is part of a puzzle and what is not. Eventually, the plan would be to implement textures and hide the interactable objects in the environment to make the game harder.

There will be only the player character and likely no non-playable characters, though this is subject to change in the future. The perspective of the game will be based in first person with WASD movement and mouse view.

## Assets

I will be using mostly my own assets, with the exception of a player controller and potentially some buildings or puzzles. This is because I will not be using any textures so I will not be using any other sources for this, therefore greatly limiting the need for external assets. I will also be using audio assets for different features, such as button pressing and losing a life, as I am not confident that I can create high quality assets for audio. Other than this, I do not plan to use any assets.

## Gameplay features

There are many gameplay features within my game which will be covered below.

**Jump**

The character will be able to jump around which will help to solve the puzzles in certain scenarios. This will be executed with a spacebar.

**Movement**

The character will be able to move around with the WASD keys on the keyboard and this will allow them to play the game.

**Interact**

The player will be able to interact with the interactable game objects by looking at the with the mouse key and pressing the interact key, which is currently E.

**Score**

The player is able to gain score by completing the puzzles in the game and progressing levels.

**Lives**

The player will have a limited amount of lives meaning that they have to complete the puzzle correctly within limited time and lives, giving them a chance to lose.

**Collision detection**

My game will also feature collision detection to prevent the player from walking through walls and enabling them to interact with game objects within the match.

**Keypad**

There will be a puzzle featuring a keypad which the player will be able interact with. It will function by validating input that the player inputs and checking if it is right or wrong.

**Rotational puzzle**

There will be a puzzle in my game that the player will be able to solve by interacting with the object and turning it until it is completed and validating.

# External feedback and review

# Justification

Justify design decisions

# Evaluation of design against client requirements

Levels

Lives

Collision detection

Score

Adding Score

Deducting Lives