

MEP - Submission

by Michal Pfeiffer

Submission date: 10-Jan-2025 02:23PM (UTC+0000)

Submission ID: 247960755

File name: 40136_Michal_Pfeiffer_MEP_-_Submission_628113_1559379675.docx (79.66K)

Word count: 476

Character count: 2276

3D OGL Games Programming

Graphics and Games Programming

By Michal Emil Pfeiffer (2204569) - BSc (Game Development)

Introduction	1
Game	1
User guide.....	2
Reference.....	4
Appendix	5

Introduction

In this assignment, the task was to create a Matching Pairs game (memory game) with the use of OpenGL libraries; by basically using the game engine we have created during lectures.

Game

When creating the game using the game engine we have created during lectures, there were many difficulties trying to get the engine to work properly (as discussed in the video), so due to those difficulties of trying to get everything to work, I have ended up creating a basic version of a matching pairs game.

The game is an altered version of source [1], with better comments and more optimised code.

User guide

1. The user will at first be presented with title, how the board looks like and to input the coordinates for their first number

```
Welcome to Memory! Otherwise known as a Matching Pairs Game!

  0  1  2  3  4
--|-----
0  *  *  *  *  *
1  *  *  *  *  *
2  *  *  *  *  *
3  *  *  *  *  *

Flip two cards!

Input row and column for the first card:
|
```

2. The user should input numbers that are within range of the board; row = 0 to 3 and column = 0 to 4

```
Input row and column for the first card:
0
0
```

3. After the correct input, the user is asked to input the coordinates for the next number (follows the same rules as above)

```
Input row and column for the second card:
0
1
```

4. If the numbers are a match, it will display text as such;

```
Match!
```

5. After each selection, it will display what numbers have been selected (so that the user knows what numbers are where for their coordinates)

```
Card 1 is 2 and Card 2 is 2
```

6. As the numbers are a match, they will now be displayed on the board

	0	1	2	3	4
0	2	2	*	*	*
1	*	*	*	*	*
2	*	*	*	*	*
3	*	*	*	*	*

7. This then loops until all pairs have been found

```
Flip two cards!

Input row and column for the first card:
|
```

8. If the cards are not a match, this what is displayed;

```
Input row and column for the first card:
0
2

Input row and column for the second card:
0
3

No match!

Card 1 is 3 and Card 2 is 0
```

9. As the numbers were not a match, the numbers will not be displayed on the board

```
Card 1 is 3 and Card 2 is 0

  0 1 2 3 4
--|-----
0  2 2 * * *
1  * * * * *
2  * * * * *
3  * * * * *
```

10. If one or both inputs are of a number that has already been matched, it will inform the user that the number has been revealed already

```

Input row and column for the first card:
0
0

Input row and column for the second card:
0
2

This card is already flipped!

```

11. If the user tries to enter a number that is out of range, the user is prompted to input a value that is within range (if the input is anything else other than an integer, the code will break)

```

Input row and column for the first card:
12
Please re-enter the number that is within range
|

```

12. Once the user has uncovered the whole board by matching all the pairs

```

You have won!

  0  1  2  3  4
--|-----
0  2  2  3  0  0
1  3  1  1  9  8
2  7  6  8  7  4
3  4  9  5  6  5

```

13. When the board is fully uncovered, the game has ended, and each time the game is run, the board is a different arrangement of numbers

Reference

[1]

DecoratorFawn82, "Classic memory game board (size 4 * 5)," Dec. 02, 2017. [Online] Available:

https://cboard.cprogramming.com/c-programming/175499-classic-memory-game-board-size-4-*5-a.html. [Accessed: Jan. 09, 2025]

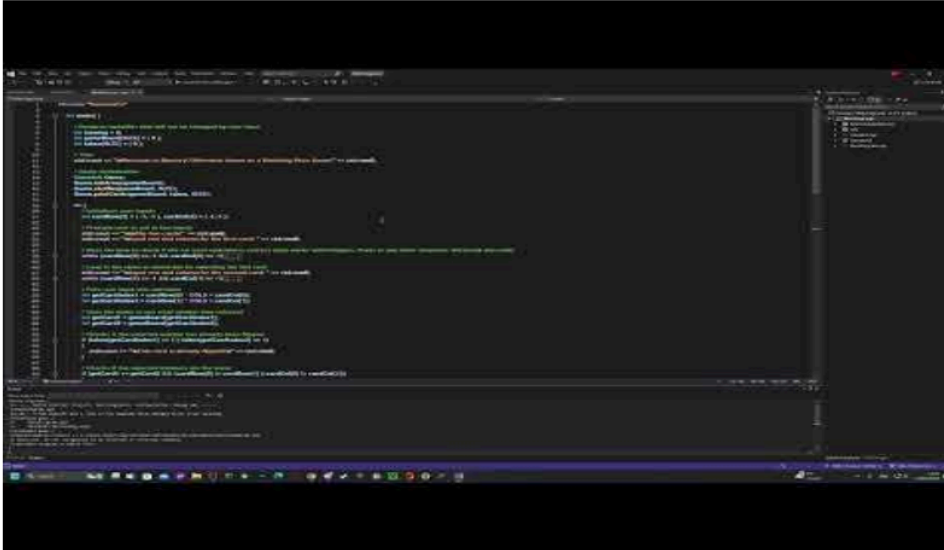
[2]

"C++ How To Generate a Random Number," N/A. [Online] Available:

https://www.w3schools.com/cpp/cpp_howto_random_number.asp. [Accessed: Jan. 09, 2025]

Appendix

Video: <https://youtu.be/u2vXHD4uKsQ>



The game would've been made using the game engine we have created, reasoning as to why I have not used it is in the video above.

Code used has been uploaded to Moodle.

MEP - Submission

GRADEMARK REPORT

FINAL GRADE

50/100

GENERAL COMMENTS

Comments:

I just wanted to say, you weren't provided a game engine to use, what we developed in class was a test harness to demonstrate the core functionality of OpenGL and yes you were free to use it however you wish but you were in no way limited to that implementation, in fact, it would have been better to not. I know in your video you mentioned that your opinion is that a pairs game in 3d was pointless, but the idea is to demonstrate your knowledge, of which I know you are very knowledgeable in the area.

In the video you demonstrated an error with the textures while showing a uniform for colours, the two aren't linked and it was probably your texture implementation having trouble locating the file required and I would be interested in seeing that fixed.

I am unfortunately limited to marking your 2d implementation, in the future **provide** your attempts even if they aren't working as your efforts would have been rewarded. See my provided breakdown of marks and you would have at least touched upon a few points.

Game Design (40 points)

Develop a game based on the matching pairs premise:
10/10 points

Use of OOP: 5/5 points

Include a mechanism to cover and uncover pairs of objects:5/**5 points**

Implement object picking (keyboard, collision, or god solution, or another innovative approach):3/**5 points**

Allow players to match objects; mismatched objects are covered again :10/**10 points**

Maintain high scores:0/**5 point (local file storage or db solution)**

Graphics and Rendering (40 points)

Use C++ with relevant APIs/SDKs:
0/5 **points (OPENGGL/GLFW/GLEW/GLM/IMGUI)**

Camera system: 0/**5 points**
(Orthographic/Projection)

Implement a lighting system:0/**5 points**

Create and import appropriate textures: 0/**5 points**

Apply textures to primitives: 0/**5 points**

Import game objects using a consistent file format: 0/**3 points**

Implement depth testing: 0/**5 points**

Apply surface culling: 0/2 **points**

Batching: 0/**5 points**

Documentation/Report (20 points)

Write a report detailing the structure of the program using appropriate notation: 6/**8 points**

Explain the important algorithms and data structures used, justifying their use: 6/**7 points**

Evaluate the application by testing and validating it: 5/**5 points**

Alternatively a video discussing the same topics as the report while highlighting the key areas of the code and operation of the system

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5
