Motivation of Design Choices
––– ––– ––– in 2048 ––– ––– –––

Name: Kieran Rigby
CID: 01067343

EIE1 - Introduction to Computing – Spring Coursework Assignment

Introduction

For this project, good design choices and habits were essential as I quickly learnt after taking long breaks from working on my program and then struggling to understand what a function did or not understanding what a certain variable was doing. Quickly, I decided to set myself a few standards in my code to make mine and other people's lives easier if they wanted to look at my code (I plan to upload my code to GitHub after the assignment has been marked). The following techniques were used throughout my code:

- **Each function must have a comment before it giving a brief description, a description of any parameters and any return values.**

- **Each variable must be clear, concise and meaningful.**

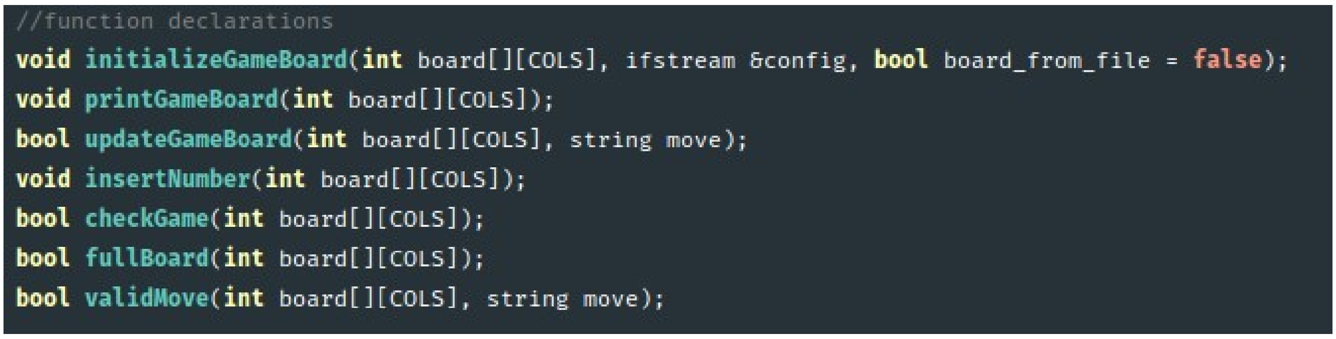
- **Every section of code must have a detailed code which follow the instructions and explain what the code is doing.**

Once I followed these few standards, I found that jumping back into the code was a lot easier.

Declarations

To start, I first defined two preprocessor constants called *"ROWS"* and *"COLS"* which are used everywhere throughout the program. The main choice for declaring these two constants was to give myself the freedom of expanding the program easily to different sized grids if I needed to by just changing these constants and using good programming techniques in my code for compatibility with different sized grids.

Other than this, I had to prototype my functions. Here is a list of the functions that I created:



```
//function declarations
void initializeGameBoard(int board[][COLS], ifstream &config, bool board_from_file = false);
void printGameBoard(int board[][COLS]);
bool updateGameBoard(int board[][COLS], string move);
void insertNumber(int board[][COLS]);
bool checkGame(int board[][COLS]);
bool fullBoard(int board[][COLS]);
bool validMove(int board[][COLS], string move);
```

In the source code, before the actual function, I have written very detailed comments about what the function does and what parameters it takes and what the function returns. If you would like details about what functions do then please refer to the main.cpp file and go to the specific function you would like to read about.

Decisions

In the main function I defined a *“gameboard”* as a two dimensional array of integers. I chose this type because the number of items I need is going to be fixed, the number of rows or columns doesn't change in the middle of the game and so I can take advantage of this and assign the memory I need right from the start. Arrays are very easy to pass into functions compared to vectors too because the name of the array is essentially a pointer to the first element of the array and so by passing this pointer into a function it is automatically passing the array by reference. With vectors you have to pass another vector into the function by reference to edit the vector which would in turn make the code more confusing.

I do utilize vectors in the *“insertNumber”* function. The naïve approach to inserting a number to a random empty space on the board would be to generate a row and column index and check if the element at those indexes are empty and if not, repeat until you find an empty element. This works but there is a small chance it could land on a non-zero element many times. Instead, I use 2 integer vectors to store all the x and y indexes of the empty elements and then pick out from random out of those vectors. Vectors are more suitable here because I don't know how many spaces are going to be empty in the 2D array. I could just set assign a ROWS*COLS amount of space, but that would be wasteful.

I thought that I should make separate function to initialize the gameboard array because it could be initialized in 2 ways, with a file or without a file. It's nice to take away all that initialization code from the main and have a function do it. I also make use of a default parameter, so by default the function will initialize the array to not read from the file.

Conclusion

Overall, I believe that the program I have written is very versatile being able to work with different grid sizes at the change of a constant. I also believe that I documented my code extremely well, used clear names for functions and variables.

However, looking back on my code I could have definitely re-used some of the codes for the shifting. Instead of having one main function to handle all cases(up, left, down & right), I maybe could of write a generic move function which takes in the direction as a parameter and then adjusts the parameters of the for loop accordingly. However I am very pleased with the outcome of this program.