**Introduction**

Title: 2048

This is a number puzzle game. It is played on a 4 x 4 grid which begins with two numbers (of value 2 or 4) placed at arbitrary locations on the grid.

The player can select to move up, down, left, or right. Each move shifts all the tiles on the grid in the chosen direction. If the adjacent tiles have the same value, they will combine and the resulting tile will have a value of the sum of two numbers (or double the number). After the move, a tile of value 2 or 4 will be spawned at a random position on the board.

The objective of the game is to combine equally-valued tiles to reach the 2048 tile without getting stuck. If the grid is full and another tile cannot be spawned, then the game is over.

**Summary**

Project size: 1,038 lines of code

Number of methods: 18

This project was quite challenging to recreate. It took approximately 60 hours to complete. This project includes many of the concepts from chapter 9 through 16 in the book and I also used classes instead of structures. The difficult components of this project included using syntax I was unfamiliar with and having to look up on Google for references, algorithm for moving the tiles in the grid, and converting the Javascript code of the original 2048 to C++ because I don’t have any experience with Javascript.

**Description**

Programming the whole game’s algorithm was the most complicated part of my project.

Refer to flowchart.

**UML**

|  |
| --- |
| BackupGrid |
| - bmaxTile : int  - bscoreRound : int  - bscore : int |
| + BackupGrid() :  + updateTile() : void  + setMaxTile(m : int) : void  + getMaxTile() : int  + void setScoreRound (s: int) : int  + getScoreRound() : int  + getScore(s : int) : void  + getScore() : int; |

|  |
| --- |
| Game |
| - player : Player  - list : vector<Player>  - winValue : int |
| + Game() :  + displayGrid() : void  + displayHelpScreen() : void  + displayWinScreen() : void  + displayLoserScreen() : void  + displayTryAgainScreen(i : int) : char  + savePlayerInfo() : void  + logic() : void  + checkGameOver() : void  + backupGrid() : void  + undo() : void  + setWinValue(i : int) : void  + getWinValue() : int |

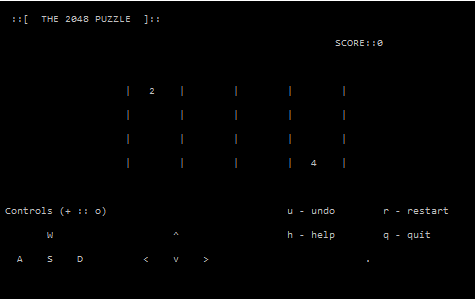
|  |
| --- |
| Grid |
| - maxTile : int  - scoreRound : int  - score : int |
| + Grid() :  + grid[4][4] : Tile  + initializeGrid() : void  + updateGrid() : void  + fillSpace() : void  + spawn() : void  + findGreatestTile() : void  + updateTile() : void  + setMaxTile(m : int) : void  + getMaxTile() : int  + getScore(s : int) : void  + getScore() : int; |

|  |
| --- |
| Player |
| - name : char\*  - pscore : int |
| + Player() :  + setName(char\* ) : void  + setScore(int) : void  + getName() : char\*  + getScore() : int |

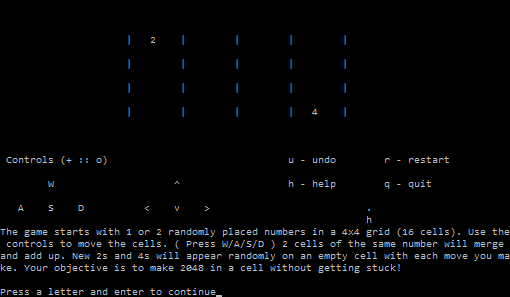
|  |
| --- |
| Tile |
| - value : int |
| + Tile() :  + getValue() : int  + setValue(v : int) : void  + isEmpty() : bool  + updateTile() : virtual void  + randomNum(n : int) : int  + operator+=( t : const Tile&) : Tile  + operator==( t : const Tile&) : bool  + operator+( t : const Tile&) : Tile  + newTile() : void |

**Sample input/output** (screenshots)

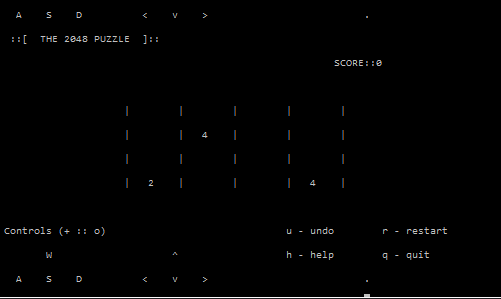
Initial console output:



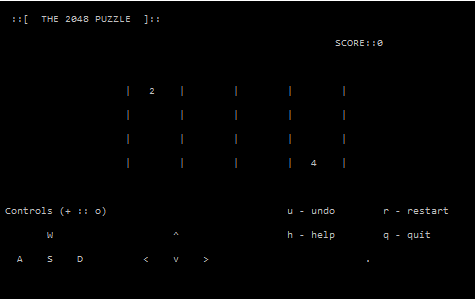
When user enters ‘h’ for help:



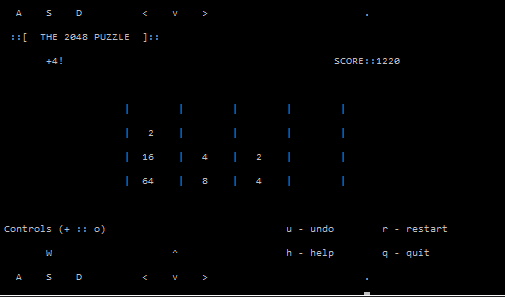
When user enters ‘s’ for down and a new tiles is spawned:



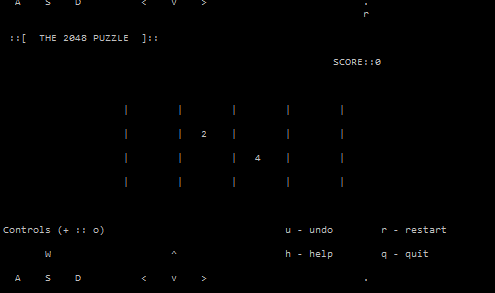
When user enters ‘u’ for undo:



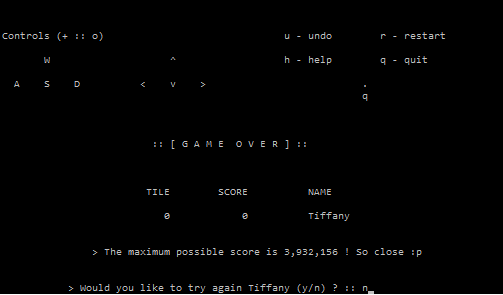
After several moves:



When user enters ‘r’ for restart:



When user enters ‘q’, enters name which is written to binary file, and ‘n’ for try again:



**Pseudocode**

*Initialize grid*

*Display output screen*

*While game has not ended*

*If ‘w’ is pressed*

*Move tiles up*

*Else if ‘a’ is pressed*

*Else if ‘s’ is pressed*

*Else if ‘d’ is pressed*

*Else if ‘q’ is pressed*

*Else if ‘r’ is pressed*

*Else if ‘h’ is pressed*

*Else if ‘u’ is pressed*

*Spawn new tile at random location that is empty*

*Output grid*

*Score of Round = log2(tileValue-1) \* tileValue*

*Update grid*

*Combine equal adjacent tiles*

*Check for game over*

*If grid is full && cannot spawn new tile || maxTile = 2048*

*Input player info*

*exit*

**Variables**

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Variable Name | Description | Location |
| int | playerScore | Score of player | Player Class |
| char[] | name[10] | Name of player | Player Class |
| bool | Response | Response to quit by user | Grid Class |
| bool | tileDestroyed | Is tile destroyed | Grid Class |
| char | control | Input by user | Grid Class |
| int | maxTile | Maximum tile | Grid Class |
| int | winValue | Value of tile to win | Grid Class |
| int | scoreRound | Score of the current round | Grid Class |
| int | score | Running total of score | Grid Class |
| Tiles [][] | tiles[4][4] | Grid that is 4x4 matrix | Grid Class |
| Tiles [][] | btiles[4][4] | Backup grid | Grid Class |
| char | option | Option input by user | Game Class |
| Player | player | Player object | Game Class |
| Game | Exec | Game object | int main() |

**Concepts**

|  |  |  |
| --- | --- | --- |
| Chapter | Concept | Location |
| 9 | Pointers | Main line 21; Tile line 56, 62 |
| 10 | Characters | Player.h line 24, main line 33 |
|  | Strings | Game line 167 |
| 12 | File | Game line 168, 243 |
| 13 | Classes | BackupGrid, Game, Grid, Player, Tile |
| 14 | Static member variable | Game.h line 43 |
|  | Copy constructor | BackupGrid line 17; Grid line 19; Player line 15; Tile line 15 |
|  | Overload operators | Tile line 53,59,65 |
| 15 | Inheritance | Game line 17 |
|  | Polymorphism | Grid and BackupGrid extends Tile |
|  | Virtual functions | Tile.h line 19; BackupGrid line 27; Grid line 131 |
| 16 | Exception | Game line 170 |
|  | Vectors | Game.h line 42 |

**References**

1. The original 2048 (javascript): <https://github.com/gabrielecirulli/2048>
2. Starting out with C++ by Gaddis

**My code:** https://github.com/teepann/PanTiffany\_CSC17a\_48983/tree/master/Project%202