**OPSC7311 POE (ASSIGNMENT 1)**

RobERT HURST

2017

**Table of Contents**

Contents

[Table of Contents 1](#_Toc484601781)

[OPSC7311 - Assignment 1 Documentation 2](#_Toc484601782)

[Introduction 2](#_Toc484601783)

[Specifications 2](#_Toc484601784)

[Flowcharts 3](#_Toc484601785)

[UML 6](#_Toc484601786)

[Help File 8](#_Toc484601787)

[Swipe Left 10](#_Toc484601788)

[Swipe Up 11](#_Toc484601789)

[Swipe Down 12](#_Toc484601790)

**OPSC7311 - Assignment 1 Documentation**

**Introduction**

This is the first assignment for 3rd year years students currently Studying the Diploma in Information Technology: Software Development course. This assignment was designed to test a student’s creativity, and application of knowledge based on Android development. The assignment looked at how we could apply the basic knowledge beginner grade UI. This document will go into the various development processes I took to get to my final product, and the analysis of my own work that I produced, and the outcome. This document will also contain flowcharts and various hand drawn diagrams that I used in the development of my application "**Purple Nurple**".

**Specifications**

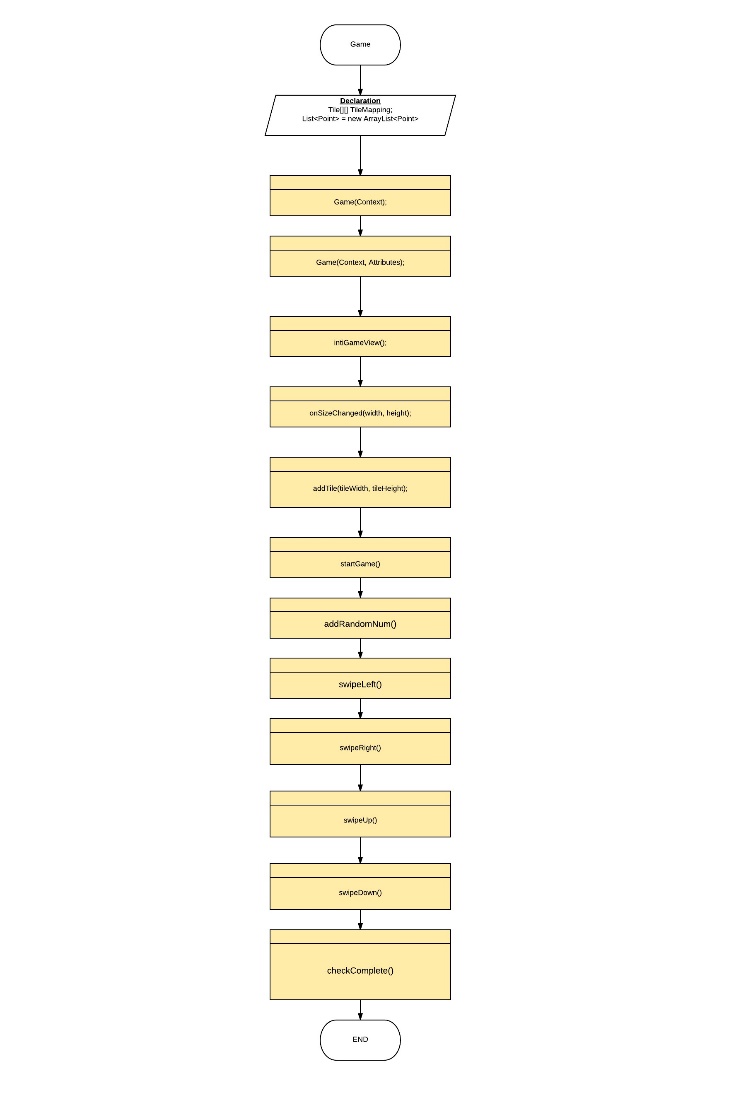
The specification document that was provided stated that the following criteria had to be met:

* Create your own Game.
* Conceptualize and Design your Own Game.
* Provide a Description that Explains the Purpose of your App.
* Provide a Simple How to Use.
* Make use of Methods and Parameter Passing.
* Make use of at least one Input and/or Output.
* A GUI designed using XML.
* Validate All input.

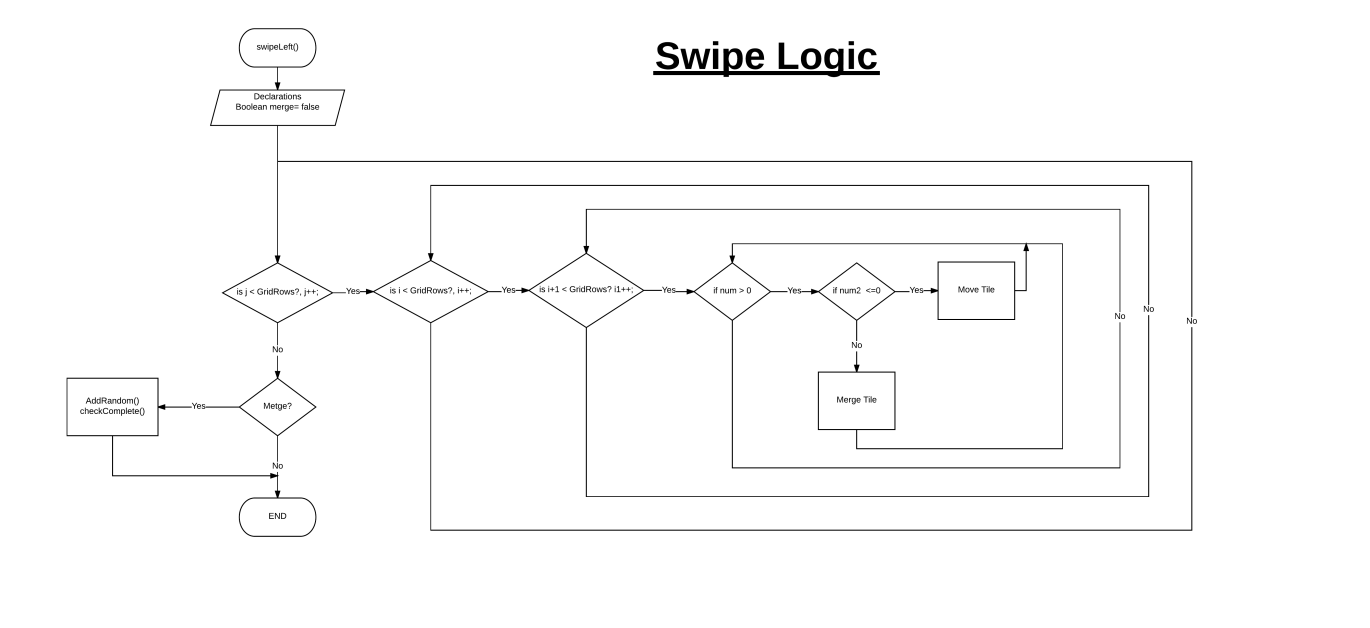
The Application must also make use of:

* Good Programming Practices.
* Good Code Efficiency.
* Make use of Commenting.
* Your Program Complies and Executes.

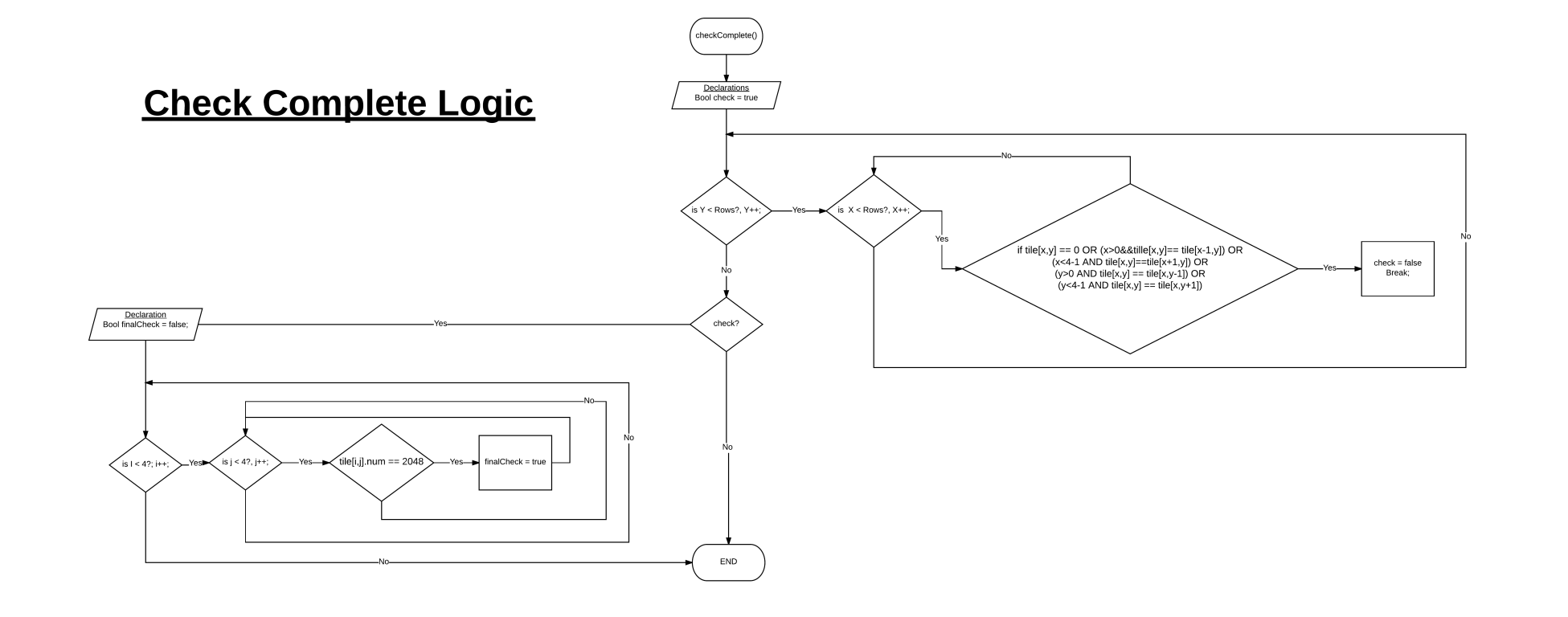
## Flowcharts

(Game Class in Images Folder)

This is the Game class that will contain all the relevant methods that will house the core logic for the game itself. It contains such key logic as the swiping logic that will check for different types of tiles that will be next to a tile in relation to the direction that the user will swipe.

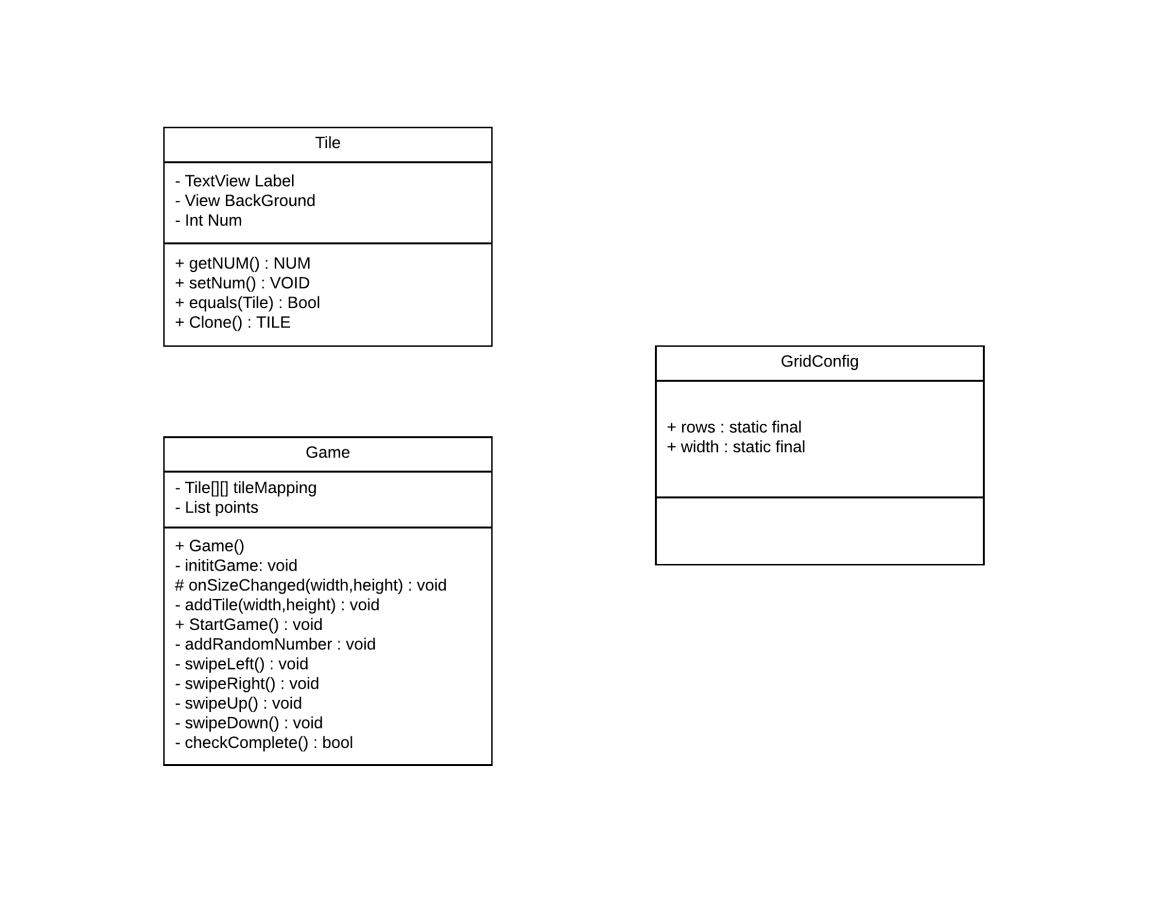


In this flow diagram I loop through the x and y tiles in the grid I essentially check if the number of the tile that is next to the tile is the same number as the current tile then perform the merge tile method, which will in turn iterate the loop further.



 In this flow chart I have two checks that will look through a whole bunch of conditions that essentially check if there are no more moves to be made within the grid itself. So checking if all tiles are filled with a number, and that no numbers can merge or shift into each other. The other check will loop through all the tiles to see if there is the winning tile i.e. ‘2048’ and provide a winning case.

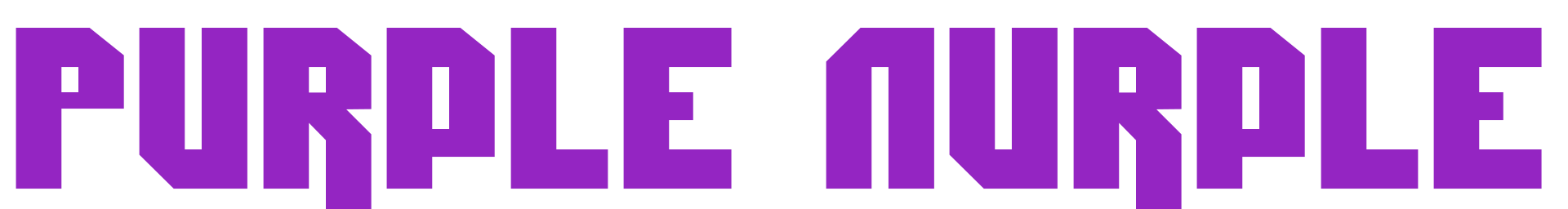
## UML



This is a UML representation of the 3 main classes that I made use of in Purple Nurple. In the tile class I defined very basic methods that will get and see the numbers that will represent the Tile itself. I also added GUI elements to the class to

manipulate the Tile itself. In the game class lies the core logic in my application. The swiping and check logic. Without these methods, my game would not function. I also added a basic class called GridConfig, which will just hold the final values of the size of my grid. I kept it at the uniform grid size of 4 x 4. But at some stage I can increase the size of the grid, by just changing this class.

## Help File



Welcome to Purple Nurple. The Grid based color game, that provides the user with hours and hours of fun for all ages.

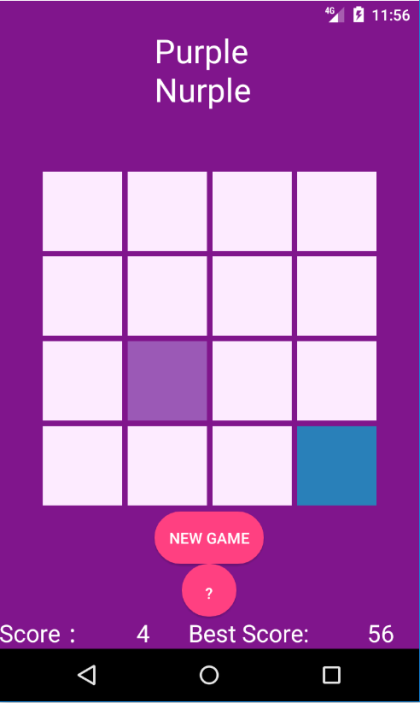
This help file will guide you through the process of playing the very simple and easy to use game of **Purple Nurple.**

Game Start



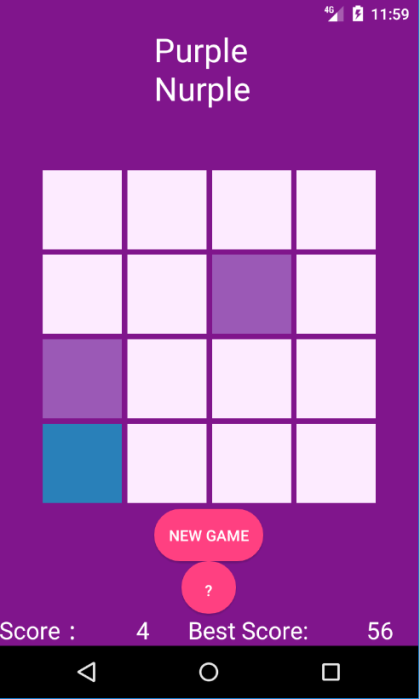
This is the start screen where the user is presented with the centered grid which will allow the user to swipe in 4 directions, Up, Down, Left, and Right. Each time the user swipes in a direction the tiles will shift in the direction of the desired swipe. An example of this is when we swipe right, we shall see the far-left tile move from the left to the right and merge into a new color. This will also result in a new tile spawning into the grid. This will happen every time there is a new swipe in the grid.

Swipe Right



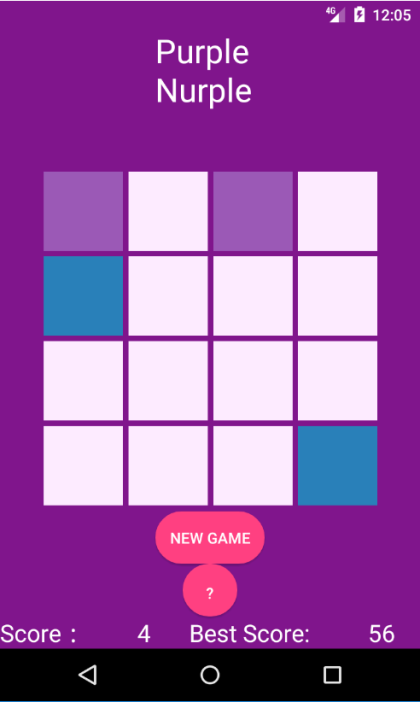
The next example will be when swiping left, so show the tiles shifting to the left side of the screen.

### Swipe Left

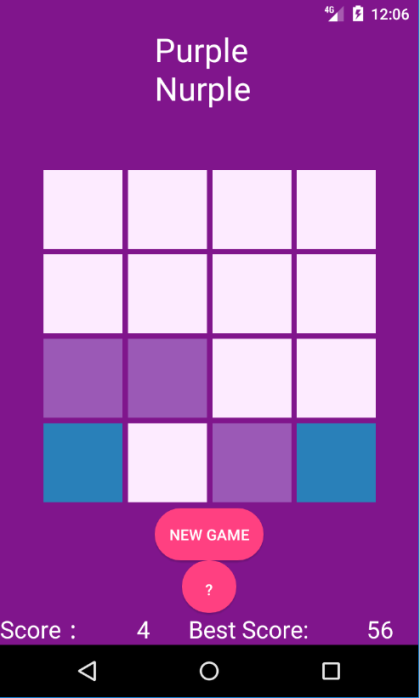


Notice the tiles shifting to the left of the screen, and a new Tile Spawning. This will be evident when the user Swipes Both Up and Down.

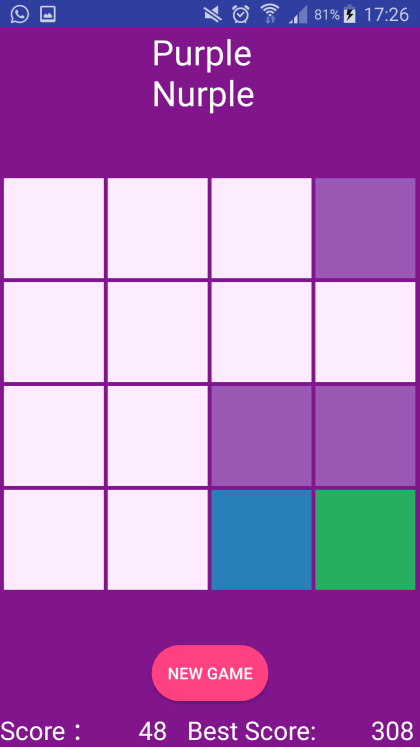
### Swipe Up



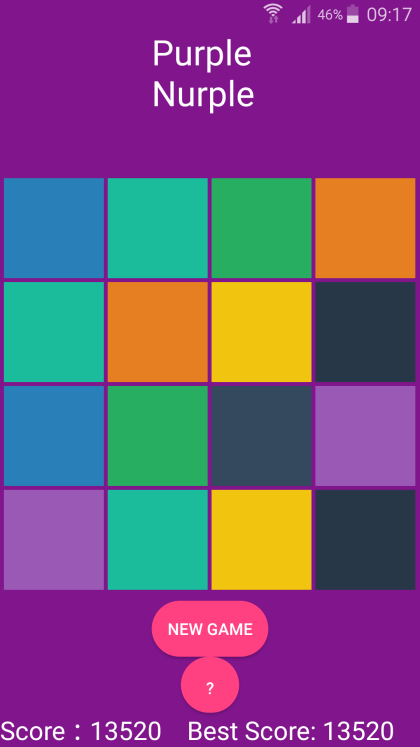
### Swipe Down



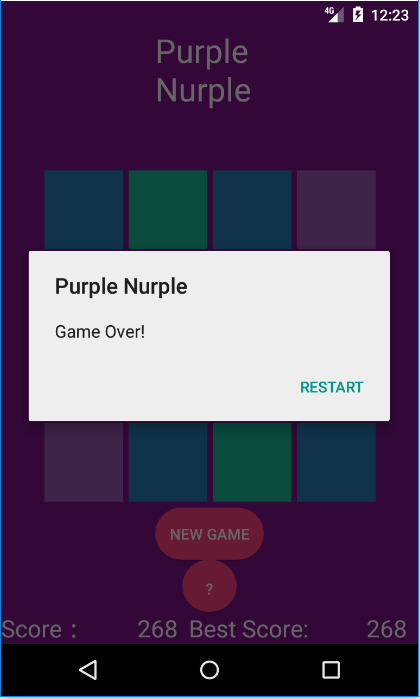
## Screenshots & Test Data



In the screen shot, I started testing if the score, and the best score would increase where the tiles would merge. The total would increase by the number that each color represented.



In this screen shot I looked to see if the Best score would increase, which in turn it did, and to see if the check game complete condition was indeed working. At this stage of the development it was not working.



In this screenshot, I got the game to a point of failure that it would indeed indicate if the game is over.

## Conclusion

In conclusion, I am happy with the current state of my application. I think it meets the criteria of the Assignment quite well and provides something that is both fun, but simple and elegant in its design. The application could use some work to possibly become something that can be place on possible app stores. I could have planned various aspects of the application better before starting to code, such as the whole winning cases and checks. But upon reflection of the app I am happy with my progress as a programmer and developer. I am learning more and more.