AC22005 Assignment #1

**Game using C#**

*Group 8: Jokubas Butkus (180017346), Rokas Jankunas (180017115), Justas Labeikis (180012794)*

During our first group meeting we settled on making a game called “2048”. “2048” is a popular mobile game that has lots of variations and clones on the app store. The game is made of a four by four grid (some variations have different grid sizes; our implementation of the game allows changing the grid size in code) with empty tiles and 2 tiles with numbers. The player is supposed to move the tiles (by clicking on one of the edges of the grid or using a keyboard) thereby joining two identical numbers to make a bigger one. For example, joining 2 and 2 together makes a 4, 4 and 4 makes an 8 and so on. The goal is to attain the 2048 tile.

One of the first challenges we faced was taking care of the movement. When a player clicks on the edge of the grid, all tiles move towards that direction. At the same time, tiles with the same number are merged to make a bigger one, and also a new tile is generated. In our implementation of the game we have a 2d integer array that contains a number for each tile. The algorithm we came up with performs a loop on the columns (if the user clicked on the top or bottom edges) or rows (if the user clicked on the left or right edges), traversing the opposite direction of the movement to sum up the values and add them to a new temporary array which is later copied to the appropriate row or column. The hardest part in this was making sure the values end up summed up correctly due to the fact that there are 2 things happening at the same time - tiles have to move and also merge.

For our AI part we are not only checking whether there are empty spaces or neighboring tiles that can merge but we’ve also added what we call a “hard mode” which generates a new tile after each movement in the worst possible place so that it is harder to group the values together before merging. More specifically, the newly generated tile is currently being placed along the opposite edge of the movement right next to the highest value. For example, if the values on the bottom row are 2, 8, 4, 0, the newly generated number 2 tile will be placed right under the 8 value if possible (if not, it will be placed to the second highest value 4 and so on).

If we were to continue this project, most likely we would add functionality to allow to change the grid size during the gameplay or when the program is launched as opposed to in code as it is right now. In addition to that, some animations would be helpful to better recognize which tiles were merged instead of them jumping to a new place. Furthermore, more sound effects would allow the user to immerse better.

Overall what helped our team work efficiently was splitting the game into smaller tasks such as “movement”, “sound effect”, “AI” etc. and using the Github’s new “Board” functionality with columns “Todo”, “In progress” and so on. This allowed us to start and end tasks quicker and have a working product throughout the whole process. Using git and separate branches for each task allowed us to integrate the changes together and work at the same time on different tasks.