For the purpose of this writing and my program, a grid is a rectangle space.

The biggest change I have made to the original design is I created a **class Grid** in my program, which I did not have in my design. The reason is because the **game is just a bunch of grids of the same size put together**. Their only **two differences** are each one has a **different location** than the other one and each one **might** have **a different display** than the other one. Thus, the class should have **local variables** that are going to hold **location information** and to **control** what the grid should **display**.
Therefore, we know that the class needs variables to do 3 things.

First, to determine the size of the grid. Since the size of the grids is the same, their width **gridW** and height **gridH** should be **global variables**,

Second, to indicate x and y location of the grid. A grid's **location is unique** because each one has unique column **(colN)** and row (**rowN**) numbers. Considering **rectMode(CORNER);** , colN being in the range of **0 to nCol-1** and rowN **0 to nRow-1**, the x position is determined by **gridW*colN** and y position by **gridH*rowN**. The grid perimeter is drawn by: **rect(gridW*colN, gridH*rowN, gridW, gridH);**

Third, to determine what to display in the grid. A grid either displays the number **neiM** of neighboring mines, a **mine** image, a **flag** image, a red X (**flagX**) image (only if the game is over), or **nothing** at all.
Grid only displays **neiM** or **mine** if the player chooses to **open** the grid. So, we need to be able to tell if the player chose to open the grid or not. I created a **boolean** variable **open** to do so. Grid only displays a flag image if the player chose to put a **flag** on the grid by clicking right of the mouse. We can create a **boolean flagOn** variable to keep track of that. Grid only displays a red X if the **game is over**, the player has put a **flag** on the grid but it **does not contain a mine**. I decided to create another **boolean** variable **flagError** to know if that case is occurring. Grid should display nothing inside if none of these boolean variables (open, flagOn, flagError) is true. The reason to create all of these boolean variables is all about organization of the program because this way, everything about a grid’s display is programmed inside the method (of Grid) **void display()**_the method uses these boolean variables to decide on what to display while these booleans are being decided by the player (while playing) programmed inside the method **void mouseClick()** method or by the function **void gameOverPage()** when the game is over or **void gameComplete()** when the game is complete.

Therefore, instead of creating a 2D array **grid** of integers as I mentioned in my design, I created a **2D array of Grid,** which makes the game much easier to code.

The second biggest change is the implementation of the recursive function **void ifOpen0()** that opens every neighboring grid if the player opens a grid that is not a mine and has 0 neighboring mines.

Another function that was not mentioned in my design that my program has is the **void resetGame(int n)** fonction that will reset variables of the game according to the input n, which is the player's choice of numbers of mine.