This is the file with the instructions to execute the program and the algorithm implemented to solve the exercises.

ASSESSMENT 1:

First of all, I have created a class called “cell” that will inherit the matrices of exercises 1 and 2. This cell has some features which are height, width, position, if it is touched or not (by a button or a circle), and several methods to obtain these features or to change them.

Therefore, I have created another class called “matrix” which creates a matrix that inherits the class “cell” in every cell. This matrix is drawn in the frame and when the user presses the button in the matrix the initial position of the cell is saved. Then, the circle is drawn tracking (with a geometric conversion of coordinates) the mouse when it is dragged.

Finally, when the user releases the mouse, the function “cell_closer” is called to determine which are the cells that are touched by the circle and return 3 elements:

- The radius of the 2 circles that surrounds the cells.
- The variable error represents if the circle dragged is inside the matrix and the radius is not so small.

The touched cells are determined taking into account the center, the radius, and applying cos() and sin() of the 360 degrees. Then, these positions are compared with the positions of the cells and if they match, then the cells change their color.

Moreover, the corner of the furthest and closest touched cells are chosen to create the two circles. However, if the radius of the circle generated by the user is so small or the circle is outside the matrix, the game is reset again.

The main drawback of this activity is that we have to create a single continuous line of cells and avoid the situations of Figures 1, 2, and 3: when we consider a big or medium range or we consider the cells that touch directly the circle or a small range.

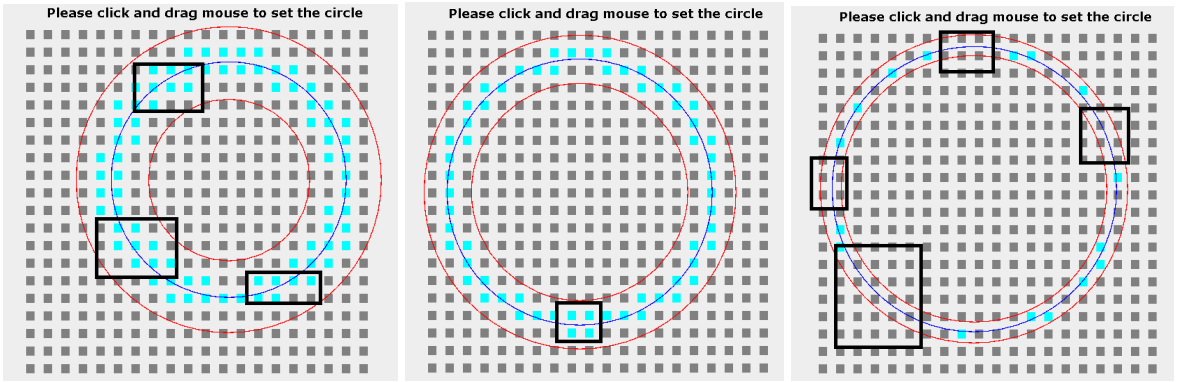

Figure 1,2,3: Examples of bad performance of exercise 1.

So, to avoid this problem, I have considered checking every specific angle which is obtained by Figure 4. Please, take into account that the height is the same as the width and I have added 10% of the height to assure that we will be in a cell in the next iteration.

This step reduces the time of computation and also gives a more precise boundary.

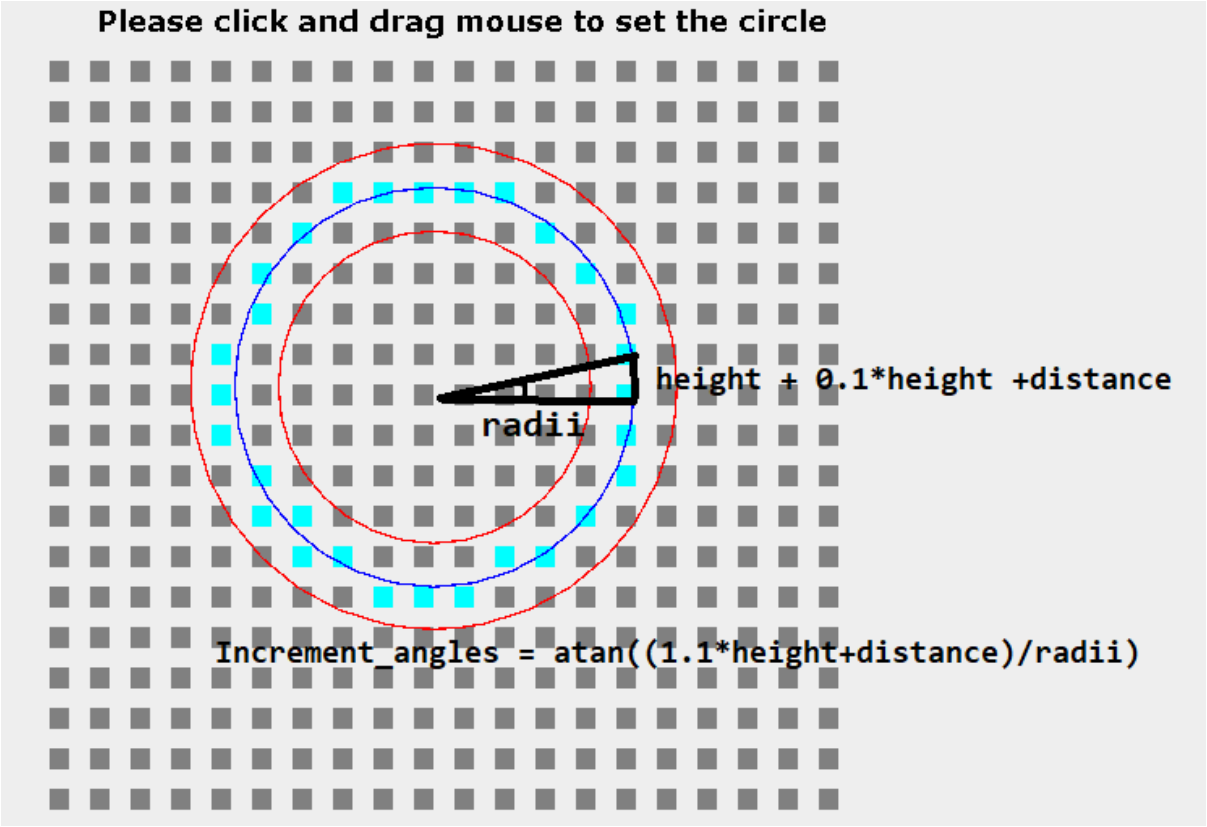

Figure 4: Strategy implemented to improve the performance of exercise 1.

ASSESSMENT 2:

In the second assessment, I have used the same matrix as in the previous exercise but adding a button to generate the circle that fits better the generated points.

First of all, the user has to select the desired cells. Then, the program obtains the position of the mouse to check (with the function cell_closer) if it is inside a cell and changes its color and state. If the cell was clicked previously, then the position of the cell is removed.

Furthermore, when the button “Generate” is clicked, the function “fill_Center” is called to find the center of the circle that fits better the selected points. The strategy implemented is the following:

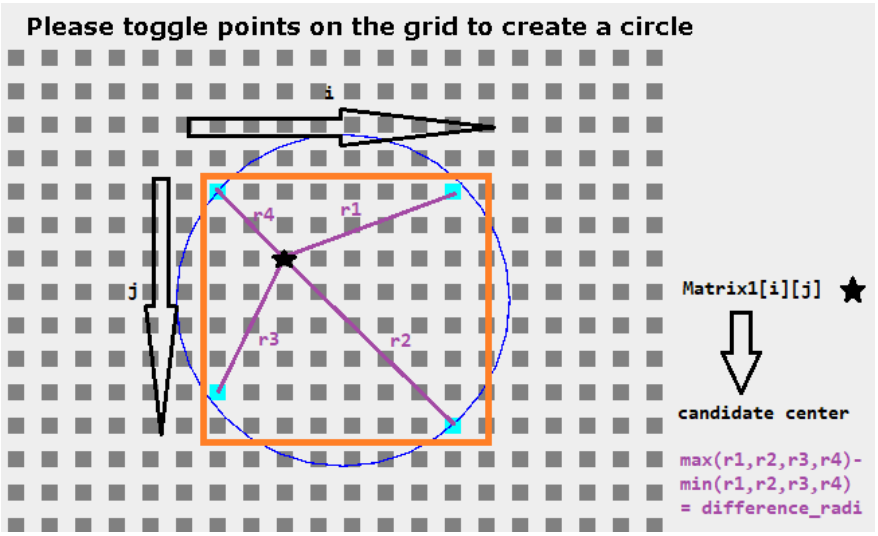

Figure 5: Strategy implemented to iterate between the points generated to find the circle that fits them.

I have created a mesh between the selected points and I have iterated for all the positions of this grid. Moreover, I have calculated the radius between the position of the mesh with the points generated and have chosen the center (with the function “calc_radii”) whose radius is similar to the different points generated.

Please take into account that I have not iterated between the limits of the grid continuously, but I have been iterating every 5 units (since the side of a cell is 10) to save time and computational cost.

If the difference between the different radius to each point generated is smaller than a threshold (I have considered a threshold of 40 units), then update the positions and the radius to consider it as the best center and radius at the moment.

However, when the user selects the points that form a contour different from a circle (see Figure 6) , then the program does the mean of the points and generates the maximum radius to ensure that the points are inside the circle.

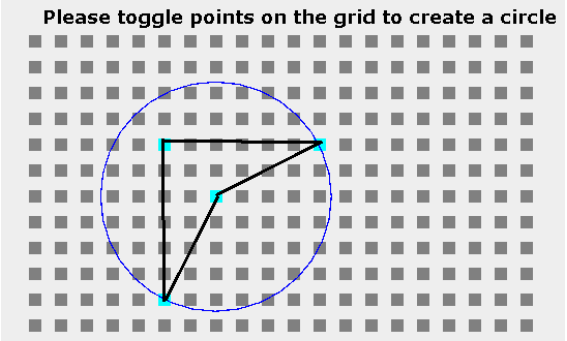

Figure 6: Points that form a contour different to a circle.