Stryker MAKO

Software Engineer C++ Assignment

# Guidelines

* Remember that you are competing with other candidates for this position. Try your best to impress us!
* Take no more than seven days to return your completed solutions.
* Develop your solutions on Windows, if possible. If you develop on Linux or another platform, provide instructions on how to build and run your solutions.
* A skeleton program is included that provides the functionality for drawing points and circles, as well as handling mouse and keyboard input. The program is packaged in a Visual Studio 2013 project.
* Develop all original code. While it is fine to look at other code for examples, do not copy code or derive your code from other code. Do not rely on any third party libraries.
* Demonstrate your analytical skills with algorithmic solutions that are efficient and original. Do not just pick the easiest to code solution. Explain your algorithms.
* Demonstrate your software engineering skills with:
  + Proper documentation
  + Clear and easy to understand code
  + Consistent naming conventions
  + Good code structure embodied through software design and analysis
  + Elimination or minimization of global variables, hard coded constants, code duplication

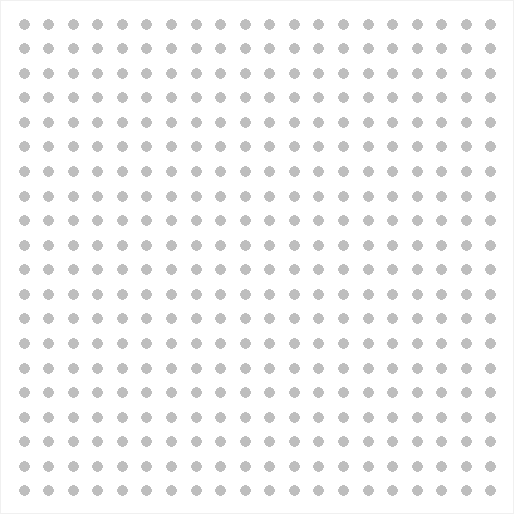
# Problem 1

## Objective

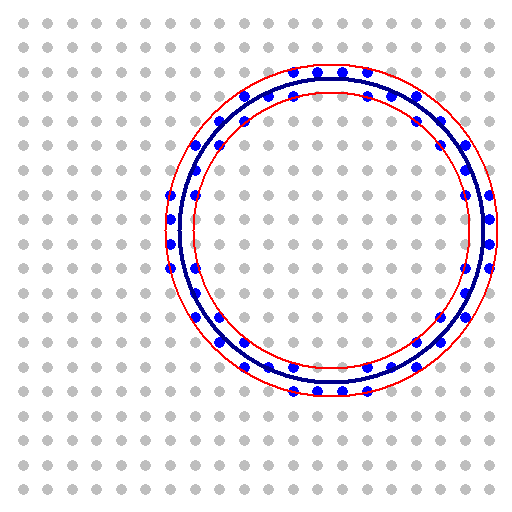
Develop a program capable of rasterizing circles.

## Requirements

* The user interface should show a single window containing a 20x20 grid of points.



* All points should start out colored gray.
* The user should be able to click and drag to define a circle center and its boundary. The mouse left button down event sets the circle center and dragging updates the circle radius.
* On release of the left mouse button, erase the circle and highlight in blue the points that nearest represent the circle.
* Draw three circles: one representing the original, user specified circle (draw this one in blue with a thick pen) and two representing the inner and outer circles for the set of highlighted points (draw these in red with a thin pen).



* Maintain a separation between the logical space of the canvas and the logical space of the 20x20 grid. Don't “snap” your circle center to a grid point. Give some thought to what it means for a point to qualify as the nearest neighbor to the circle.
* If the user clicks and drags again, return the grid to its original state and allow the user to draw a new circle.

# Problem 2

## Objective

Develop a new program similar to the first one that allows the user to click grid points and then generates the best fit circle.

## Requirements

* Start out with a 20x20 grid, as above.
* Allow the user to click to toggle grid points between the colors blue and gray.
* If the user hits “G” on the keyboard, display the circle that best fits the set of points selected.
* If the user hits “C” on the keyboard, return the grid to its original state.

## Extra Credit

Modify your solution for Problem 2 to generate the best fit ellipse.