A.

Program Starts

Static Main

Instantiate Objects

Call setPins Method

End of Static Main

setPins Method

Declare constant to increase efficiency

// Call various services to create the alley and pass in the constant

// In between each service, adjust the positioning slightly for smooth transition/increase in efficiency

bottomDiagonal

putInRow

lastPin

returnToStart

End of setPins Method

bottomDiagonal Method

For Loop

Various RobotSE services to adjust positioning and place the thing object

End of For Loop

End of bottomDiagonal Method

putInRow Method

For Loop

Various RobotSE services to adjust positioning and place the thing object

End of For Loop

End of putInRow Method

topDiagonal Method

For Loop

Various RobotSE services to adjust positioning and place the thing object

Call the putInRow Method to fill in the indexes that require an object between the end points

End of For Loop

End of topDiagonal Method

lastPin Method

Various RobotSE services to adjust positioning and place the thing object

End of lastPin Method

returnToStart Method

Various RobotSE services to adjust positioning and place the thing object

End of returnToStart Method

Program Ends

B.

The method I used is very efficient as it prevents code repetition using control structures such as loops along with various services (methods). To prevent the robots querying their current position each time they provide a service I declared a constant value. This prevents the use of extra code to get the current avenue/street but due to this the scalability of this code is affected. Due to this constant the number of rows/columns that can be created is limited. This can be rectified by prompting the user for a row/columns value and making slight adjustments to each service to make this code scalable.