Traffic Jam

A VBA project by John Harris

# Executive Summary

For my project I created a Traffic Jam game in Excel. The Traffic Jam game is a six by six grid with cars placed horizontally or vertically on it. The cars are constrained to move only left and right or up and down, depending on their orientation. The goal is to move a car, typically red, out of the grid. The difficulty is arranging the cars to clear a path.



I became fascinated with the game for its simplicity and complexity. On the gird there are only 36 tiles, but some games require over 20 moves to free the red car. This project is largely a continuation of my fascination with the game. The game shown above is very simple and can be beat in 8 moves:

1. Move the pink car in the top right to the right 1 square
2. Move the orange car up 1 square
3. Move the cyan car up 1 square
4. Move the brown car left 2 squares
5. Move the green car left 3 squares
6. Move the middle pink car down 2 squares
7. Move the blue car down 3 squares
8. Move the red car out

# Implementation

I had to create several things in order to implement the Traffic Jam game:

* A worksheet outlining the positions of cars in a board layout
* A worksheet with the actual game board on it
* A range outlining where the cars should be placed
* Code to layout the cars on the game board
* Code to handle the “click” event on the worksheet to select cars and move them
* Code to move cars on the game board, prevent collisions, and prevent the cars from going off the game board
* Code to count the moves the player makes
* Code to clear the game board
* Code to allow for multiple levels or board layouts
* Code to download attractive background images from the internet

The game board is created and run almost entirely with ranges. The game board itself is a range called “GameArea”. As each car is created, a range is created to match it. When the cars are “moved,” Excel checks to see if the move is valid by looking for intersecting ranges; it then cuts and pastes the range into the new area. At this point a redraw of the board is done to fix any problems with borders mixed by the copy and paste functions.

# Learning and Difficulties

In this project I encountered several unexpected problems. Ranges and cells proved to be much more difficult to work with than I anticipated. Cells are addressed as *Cells(row, column)*, while ranges are addressed as *Range(“A1”).* I expected Ranges and Cells to behave the same or at least use the same notations as each other.

An effective work-around for this problem was to use cells as arguments when creating ranges. However, when I just placed the cells in, an error would be thrown. To fix this I had to use the address property of the cells. For example, I could select the range A1 with *Range(Cells(1, 1).Address)*.



Another problem I kept running into was the inadequate documentation for VBA functions. The return types of functions wasn’t clear or didn’t work as expected. An example of this is the *intersect* function. This function takes two or more ranges and returns a range of intersecting cells. I expected this function to return a range with no cells if there were no overlaps. Instead, the function returns Nothing or NULL and has to be tested for as such.



Two things that I couldn’t implement due to time and possibly tool constraints were a level generator and a solver. My idea was to write code that would randomly layout the pieces and then recursively move the cars around finding the shortest path to the solution, or throw the layout away and create a new board until one was solvable. The problems were all time. I didn’t have the time required to create something so complex. Also, the processing time may have been prohibitive with Visual Basic tied into Excel. There is a lot of overhead for something so computationally intensive.

# Conclusion

Overall, I’m very happy with the way the project turned out. When the cars started to move and the collision detection was right it almost felt magical. The only things defining the cars were the ranges they were composed of. Adding the background images from the internet made the whole experience very attractive, as can be seen in the screenshot at the top.