# Handling Circular References
# Cpt S 322 Homework Assignment #10
# by Evan Olds

## Submission Instructions:

Submit source code (zipped) to Angel BEFORE the due date/time. If the Angel submission is not working, then submit to TA via email BEFORE the due date/time. "Angel wasn't working" is never an excuse.

Optional: Include a readme.txt file in the zip with any relevant information that you want the grader to be aware of.

## Assignment Instructions:

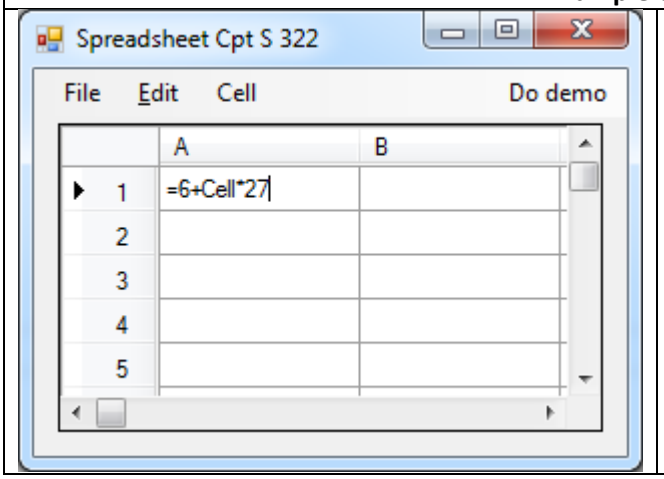
**Read each step's instructions *carefully* before you write any code.**

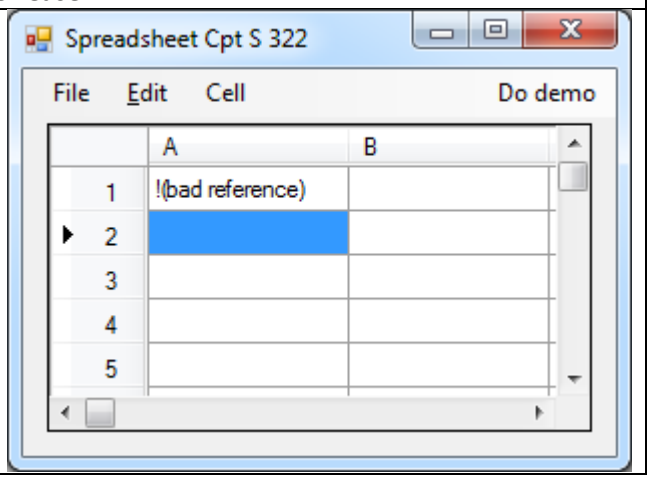
In this assignment you will give your spreadsheet application the ability to deal with problematic references in formulas. This may sound simple at first, but it is actually somewhat involved if you consider all the cases. In addition, it may require some significant adjustments to your existing spreadsheet code BEFORE you add the new capabilities (refactoring).

Below are the cases that you need to handle with respect to references in formulas. In industry work you'll need to come up with such lists on your own, but here you are given a list of possibilities to ensure that you don't miss anything. Handle all cases by setting some sort of simple descriptive message as the cell value (see video demo for examples).

1. The cell formula could reference something that doesn't exist on the spreadsheet. This could mean it's a cell name that's just beyond the range of what our spreadsheet supports, such as "Z12345". It could also just be a bad cell name, such as "Ba". Set the cell value to an error message as opposed to treating the non-existent cell's value as 0. Note that this is different than referencing a cell that DOES exist, but has no numerical value or no value at all, in which case you still treat its value as 0. This is the simplest case and is **worth 1 point1**.

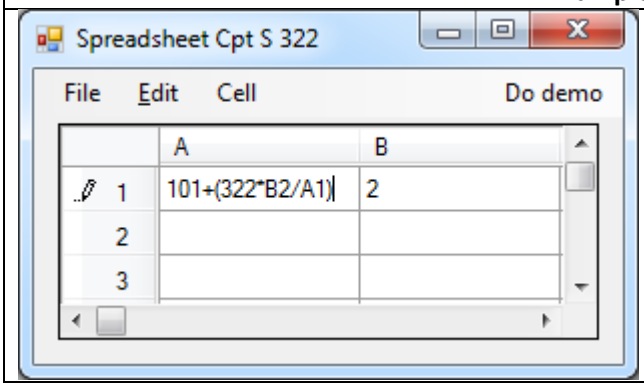
**Example of Case 1**

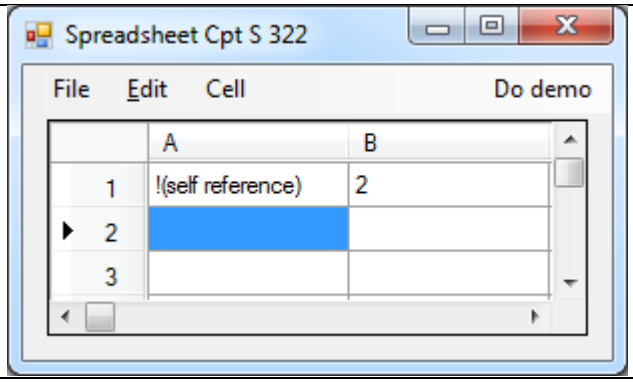



2. The cell formula could reference itself. Think of cell A1 and note that a self-referencing formula could be as simple as "=A1" or something more complex such as "=B2/(A1*A2)*7". In ANY case where it contains itself in the formula the cell must set its value to an error string. This is comparable in difficulty to case 1 and is also **worth 1 point1**.

**Example of Case 2**




3. The cell formula could contain a circular reference. A simple example is that cell A1 has the formula "=B1" and cell B1 has the formula "=A1". But note that the harder case is that the circular reference comes from more than one "step" in the evaluation chain. Consider a sheet with the following formulas:

| | **A** | **B** |
|---|---|---|
| **1** | =B1*2 | =B2*3 |
| **2** | =A1*5 | =A2*4 |

In this example no cell contains a reference to another cell that references directly back to it. But there is a circular reference chain nonetheless.

<u>Handling the circular references:</u>

You are only required to have one of the cells in the reference chain display the error message in the case of a circular reference. Or you could have all the problematic cells display an error message (or anywhere in between). That is up to you. But what you must ensure is that your app doesn't enter an infinite loop, cause a stack overflow, or (CRITICAL) fail to update properly when the formula is altered to eliminate the circular reference. This is by far the most difficult aspect of the assignment and is **worth 7 points**.

As usual, **1 point** of the 10 is reserved for proper coding style and commenting. In the scheme of things this should be easy, so take the time to write clean, well-documented code to get this point!

Final notes:

- Watch the video included in the assignment .zip file to see some simple examples of how my implementation of the application responds to circular references.
- Remember that if a cell has a bad formula and displays an error message, resolving the error must trigger a proper update so that everything refreshes properly.