Undo/Redo System Implementation
Cpt S 321 Homework Assignment
Washington State University

Submission Instructions: (see syllabus)

Assignment Instructions:

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

In this assignment, you will implement an undo and redo system for your spreadsheet application. You will also add the ability to choose the background color of a cell. Your undo system will support undo and redo actions for changing the cell's text or background color.

Part 1 – Add a background color property to cells with accompanying UI changes (3 points)

Make sure you implement this in the same way you've dealt with other cell properties. You'll have a background color in the logic layer and changing that property will invoke a property-changed event that the UI will respond to.

In the logic engine:

- Add a **uint** property to the Cell class named "BGColor". Since it's in the logic engine, we must have a UI-independent data type, and **uint** certainly meets that requirement.
- Make sure that when it gets changed you invoke the property changed event.

In the WinForms app:

- Extend your event-handling code to respond to changes to cell background colors
- Update cell backgrounds in the DataGridView accordingly
- Use the DataGridViewCell.Style.BackColor property
- That property is of type System.Drawing.Color which has a FromArgb method
- Add a button or menu option to change the background color of the selected cells
  - Change the color for all selected cells when the user selects this option and chooses a color
- Use a ColorDialog to prompt the user for a color
  - Remember that you should be setting the background color in the data/logic cell, then the UI should update in response to this change.

Part 2 – Implement the undo/redo system (7 points)

- Implement an undo/redo system as we discussed in class
- Support undoing cell text changes and cell background color changes
- Every undo that is executed should automatically push an item onto the redo stack
- Neither the undo or redo stacks should be publically exposed. That is, don't do:
    - public Stack<UndoRedoCollection> Undos
- Declare it privately then offer the ability to add and execute undo commands through public functions
    - Have a public **AddUndo** function in the spreadsheet class (or workbook class if you decide to create one of those)
- There must be menu options or buttons that allow the user to undo or redo at any time.
    - If the undo stack is empty then disable the undo menu item.
    - If the redo stack is empty then disable the redo menu item.
- Recall that each item on the undo or redo stack should be a collection of simple command objects with an accompanying title that tells the user what is going to be undone/redone. Display that information in the menu items as shown in the screenshot below.
- The design must support the addition of new undo/redo functionality without disrupting or requiring changes in any existing functionality. For example, if you implement everything correctly in this assignment then in future assignments you could add a Border property to the cell and add undo/redo functionality for changing cell borders WITHOUT having to modify anything in the undo/redo classes you implemented for this homework. You would just need to add a new class or set of classes for the new functionality.