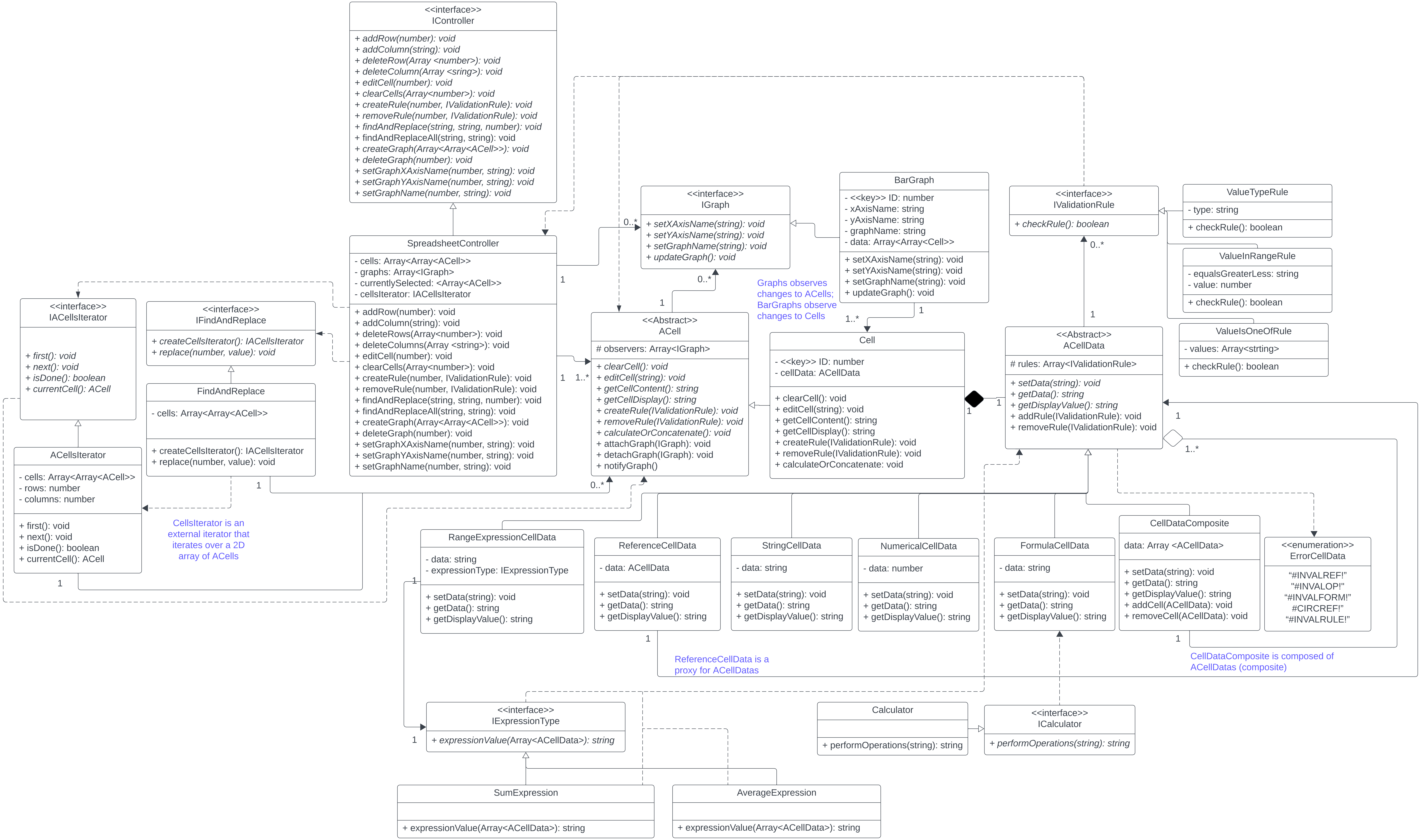


<<interface>>
IController
+ addRow(number): void
+ addColumn(string): void
+ deleteRow(Array <number>): void
+ deleteColumn(Array <sring>): void
+ editCell(number): void
+ clearCells(Array<number>): void
+ createRule(number, IValidationRule): void
+ removeRule(number, IValidationRule): void
+ findAndReplace(string, string, number): void
+ findAndReplaceAll(string, string): void
+ createGraph(Array<Array<ACell>>): void
+ deleteGraph(number): void
+ setGraphXAxisName(number, string): void
+ setGraphYAxisName(number, string): void
+ setGraphName(number, string): void
SpreadsheetController
- cells: Array<Array<ACell>>
- graphs: Array<IGraph>
- currentlySelected: <Array<ACell>>
- cellsIterator: IACellsIterator
+ addRow(number): void
+ addColumn(string): void
+ deleteRows(Array<number>): void
+ deleteColumns(Array <string>): void
+ editCell(number): void
+ clearCells(Array<number>): void
+ createRule(number, IValidationRule): void
+ removeRule(number, IValidationRule): void
+ findAndReplace(string, string, number): void
+ findAndReplaceAll(string, string): void
+ createGraph(Array<Array<ACell>>): void
+ deleteGraph(number): void
+ setGraphXAxisName(number, string): void
+ setGraphYAxisName(number, string): void
+ setGraphName(number, string): void
<<interface>>
IACellsIterator
+ first(): void
+ next(): void
+ isDone(): boolean
+ currentCell(): ACell
ACellsIterator
- cells: Array<Array<ACell>>
- rows: number
- columns: number
+ first(): void
+ next(): void
+ isDone(): boolean
+ currentCell(): ACell
1
CellsIterator is an external iterator that iterates over a 2D array of ACells
<<interface>>
IFindAndReplace
+ createCellsIterator(): IACellsIterator
+ replace(number, value): void
FindAndReplace
- cells: Array<Array<ACell>>
+ createCellsIterator(): IACellsIterator
+ replace(number, value): void
1
0..*
1
0..*
<<interface>>
IGraph
+ setXAxisName(string): void
+ setYAxisName(string): void
+ setGraphName(string): void
+ updateGraph(): void
0..*
1
1
1..*
<<Abstract>>
ACell
# observers: Array<IGraph>
+ clearCell(): void
+ editCell(string): void
+ getCellContent(): string
+ getCellDisplay(): string
+ createRule(IValidationRule): void
+ removeRule(IValidationRule): void
+ calculateOrConcatenate(): void
+ attachGraph(IGraph): void
+ detachGraph(IGraph): void
+ notifyGraph()
0..*
Graphs observes changes to ACells; BarGraphs observe changes to Cells
BarGraph
- <<key>> ID: number
- xAxisName: string
- yAxisName: string
- graphName: string
- data: Array<Array<Cell>>
+ setXAxisName(string): void
+ setYAxisName(string): void
+ setGraphName(string): void
+ updateGraph(): void
1
1..*
Cell
- <<key>> ID: number
- cellData: ACellData
+ clearCell(): void
+ editCell(string): void
+ getCellContent(): string
+ getCellDisplay(): string
+ createRule(IValidationRule): void
+ removeRule(IValidationRule): void
+ calculateOrConcatenate: void
1
1
<<interface>>
IValidationRule
+ checkRule(): boolean
0..*
1
ValueTypeRule
- type: string
+ checkRule(): boolean
ValueInRangeRule
- equalsGreaterLess: string
- value: number
+ checkRule(): boolean
ValueIsOneOfRule
- values: Array<strting>
+ checkRule(): boolean
<<Abstract>>
ACellData
# rules: Array<IValidationRule>
+ setData(string): void
+ getData(): string
+ getDisplayValue(): string
+ addRule(IValidationRule): void
+ removeRule(IValidationRule): void
1
1..*
RangeExpressionCellData
- data: string
- expressionType: IExpressionType
+ setData(string): void
+ getData(): string
+ getDisplayValue(): string
1
ReferenceCellData
- data: ACellData
+ setData(string): void
+ getData(): string
+ getDisplayValue(): string
1
StringCellData
- data: string
+ setData(string): void
+ getData(): string
+ getDisplayValue(): string
NumericalCellData
- data: number
+ setData(string): void
+ getData(): string
+ getDisplayValue(): string
FormulaCellData
- data: string
+ setData(string): void
+ getData(): string
+ getDisplayValue(): string
CellDataComposite
data: Array <ACellData>
+ setData(string): void
+ getData(): string
+ getDisplayValue(): string
+ addCell(ACellData): void
+ removeCell(ACellData): void
1
<<enumeration>>
ErrorCellData
"#INVALREF!"
"#INVALOP!"
"#INVALFORM!"
#CIRCREF!"
"#INVALRULE!"
ReferenceCellData is a proxy for ACellDatas
CellDataComposite is composed of ACellDatas (composite)
<<interface>>
IExpressionType
+ expressionValue(Array<ACellData>): string
1
Calculator
+ performOperations(string): string
<<interface>>
ICalculator
+ performOperations(string): string
SumExpression
+ expressionValue(Array<ACellData>): string
AverageExpression
+ expressionValue(Array<ACellData>): string

Group 508 - Justin Diament, Aidan Rosenberg, Ayla Dursen, Haley Schmitt

1) TITLE: Web Based Application
PRIORITY: High
ESTIMATE: 0.5 story points

AS A spreadsheet user
I WANT TO be able to access and edit my spreadsheets on the web
SO THAT I can edit and view my spreadsheets from anywhere I can access a web browser
----
ACCEPTANCE CRITERIA:
- The spreadsheet application and all of its functionality is accessible via a web browser
- The spreadsheet application functions across different web browsers and platforms to access the web
- The user can click a help button near the top of the application to view documentation that assists them in using the application

2) TITLE: Spreadsheet made up cells in rows and columns
PRIORITY: High
ESTIMATE: 0.5 story points

AS A spreadsheet user
I WANT TO be able to view a grid of cells arranged into rows and columns
SO THAT I can organize data in a readable and easily expandable way.

ACCEPTANCE CRITERIA:
- The spreadsheet shows a grid of rectangular cells, longer than they are tall.
- Rows in the spreadsheet are labeled on the left side with numbers (1, 2, 3, etc.), while columns are labeled on top with letters (A, B, C, etc).
- The cells are all initially empty/blank
- The user can use scrollbars at the bottom and right of their screen to view cells not currently in the view of their screen

3) TITLE: Cell selection
PRIORITY: high
ESTIMATE: 0.5 story points

AS A spreadsheet user
I WANT TO be able to have the cells that I am editing selected
SO THAT I can see what cells/rows/columns I will be impacting with my actions.

ACCEPTANCE CRITERIA:
- The outline of the cell or cells that have been selected are highlighted in blue
- An individual cells is selected by clicking on it once

- A cell can be edited by double clicking it to enter the “text” on the inside of the cell. After double clicking, the true contents of the cell (e.g., “REF(B2)”) is revealed instead of the cell’s display text (e.g. whatever was in cell B2).
- Multiple cells can be highlighted at the same time by clicking on one cell, holding down the mouse button, and dragging the cursor over more cells.
- The current cell a user has clicked on (or the top left and bottom right cell the user has selected if they have multiple cells selected at the same time) is displayed above the grid of cells on the left side. For instance, if the user has selected cell B6, “B6” is displayed. If the user has selected B6, B7, C6, and C7, “B6:C7” is displayed.
- While a user has clicked on a cell, the text displayed by that cell is also displayed above the grid, to the right of the cell identifier from the prior acceptance criteria.

4) TITLE: Cells can contain numeric and string constants
PRIORITY: High
ESTIMATE: 2 story points

AS A spreadsheet user,
I WANT TO be able to enter and use both numeric and string constants in my cells
SO THAT I will be able to enter labels and data into my spreadsheet to perform evaluations and calculations with my data.

ACCEPTANCE CRITERIA:
- The user is able to click into individual cells in the spreadsheet.
- After clicking into a cell, the user is able to enter both numeric constants (e.g. “1234”) and string constants (e.g. “hello world”) directly into cells in the spreadsheet and that text is displayed in that cell.
- The user is able to edit and change the numeric and string constants directly in their respective cells by clicking back into them in the future.
-The user is able to quickly clear a cell they are clicked into by clicking a button in the Edit dropdown menu, or clear all cells using another button in that dropdown menu.
- In the case that a user has previously entered one type of data, they are able to enter another type of data into that cell without error, unless there is custom data validation present preventing that.
- In the case where the user wants to be able to use text containing containing a number and non-numeric characters (e.g. “Hello3”) in the same cell, the spreadsheet treats the entire value as a single string constant.
- Cell references to a cell containing a numeric or string constant and any uses or modifications to those cell references do not modify the numeric or string constants in the original cell.

5) TITLE: Cell references
PRIORITY: High
ESTIMATE: 1 story point

AS A spreadsheet user,

I WANT TO be able to quickly and efficiently reference values already found in my spreadsheet
So THAT I can more quickly, dynamically, and efficiently evaluate data without needing to manually enter every line.

ACCEPTANCE CRITERIA:
- The user can enter cell references in the form “REF(A1)” into cells.
- When the user clicks off of the cell they were typing in, the spreadsheet correctly displays the value of the cell being referred to. When the user clicks back into the cell containing the reference, they instead see the reference text (e.g. “REF(A1)”), so they can edit the reference if they wish to.
- Calculations involving the cell containing the reference use the value at the cell it is referring to.
- When the cell that is being referenced is changed, the cell that is referencing that cell also reflects that change and displays the new value.
- If the user attempts to reference a cell that does not exist, the cell containing the reference will display “#INVALREF!”
- A cell can be referenced multiple times from different cells or within the same cell
- In the case that there is a circular reference (e.g. cell D1 refers to cell D3, but D3 refers to cell D1), all cells in the circle will display “#CIRCREF!”

6) TITLE: Range expressions
PRIORITY: High
ESTIMATE: 2 story points

AS A spreadsheet user
I WANT TO be able to specify a range of cells that I want to enact an action on at once
SO THAT I can more efficiently evaluate and perform actions/calculations/analytics on many cells at the same time.

ACCEPTANCE CRITERIA:
- The user is able to enter a range expression into a cell (in the form “SUM(B2..D5)”) or “AVERAGE(B2..D5)”).
- The spreadsheet can perform sum and average calculations on the contents of a specified range of cells, specified as described in the above criteria.
- In the case that that the user want to change the expression being performed on the range of cells (e.g. instead of SUM(B2:D5) they want to get the average of those values), they can easily change the expression and the spreadsheet will accurately perform the new calculation on the range of cells.
- In the case that the user enters a range expression with a range of cells that contains a string constant in at least one of the cells in that range, the spreadsheet will display the error message “#INVALOP!” in the cell that has the range expression in it

7) TITLE: Formulas
PRIORITY: High

ESTIMATE: 2 story points

AS A spreadsheet user
I WANT TO be able to use a variety of mathematical formulas in my spreadsheet
SO THAT I can quickly and efficiently gather information and perform calculations on my data.

ACCEPTANCE CRITERIA:
- The user is able to enter a formula into a cell involving numerical constants, references to cells that contain numerical constants, and the addition, subtraction, multiplication, division, and/or exponentiation operators. They can also enter parenthesis.
- When the user clicks off of the cell they are typing in, the value of the formula is shown in the cell they clicked off of. However, when the user clicks back into that cell, it displays the formula text (e.g. “1 + 2 - REF(B2)”) instead, so they can edit the formula if they wish.
- The formula is evaluated accounting for operator precedence rules (e.g. in the case that the user enters “5-2+3*2” it evaluates to -3). Parenthesis are also given first priority.
- Calculations can produce both negative and positive answers
- In the case that the users enter a formula containing a divide by 0 (e.g. “12/0”), the spreadsheet will display an error message: “#INVALFORM!”
- If the formula encounters a text that is not a number, a reference to a cell containing text, or a blank cell, the cell the formula is typed in will display an error message to indicate the illegal parameters: “#INVALOP!” The only exception to this is the “+” operator, which can allow text/string constants and/or numerical constants to be used for concatenation, as described in the user story below. So, no error is displayed for any combination of numerical or string constants for the “+” operator
- Cells that use formulas are able to be referenced by other cells and the contents of the cell with the formula in it maintains their integrity and remains unaffected by the reference
- The user can use more than one formula inside of the same cell (e.g the user can multiply and then also perform addition on the result of that multiplication

8) TITLE: Concatenation operator
PRIORITY: High
ESTIMATE: 0.5 story points

AS A spreadsheet user
I WANT TO be able to perform concatenation on string constants in my cells
SO THAT I can concatenate text I write and references to cells containing text to collect all my written conclusions in one place.

ACCEPTANCE CRITERIA:
- The user is able to use the + sign as a concatenation operator for text and reference to cells containing text.
- When the user clicks off of the cell they are typing in, the value of the concatenation is shown in the cell they clicked off of (for instance, “zip + zap” will be displayed as “zipzap”. However,

when the user clicks back into that cell, it displays the formula text (i.e. “zip + zap”) instead, so they can edit the concatenation if they wish.
- In the case that a numeric and string constant are concatenated with the “+” operator, the spreadsheet will allow this (e.g. “hello + 5” will display as “hello5”). Addition will only be performed with the “+” operator (in the way described in the user story above) if both operands are numeric

9) TITLE: Check for error conditions
PRIORITY: High
ESTIMATE: 2 story points

AS A spreadsheet user
I WANT TO be able to have my spreadsheet check for errors/error conditions effectively
SO THAT I can produce accurate evaluations and know why certain actions are not working.

ACCEPTANCE CRITERIA:
- In the case that the user to divide by 0, the spreadsheet notifies the user by displaying “#INVALFORMULA!” in the cell containing the divide by 0 operation
- In the case that there is a circular reference (e.g. cell D1 refers to cell D3, but D3 refers to cell D1), all cells in the circle will display “#CIRCREF!”
- In the case that there are references to an invalid cell or range of cells (such as cells that no longer exist because they were deleted or are beyond the bounds of the spreadsheet), the cells containing the invalid references will display “#INVALREF”.
- In the case that the user references a cell that already has an error, that error’s display message will be displayed in the cell containing the reference. If a cell contains reference to multiple cells with errors, whichever reference is first in the cell will have its error message reflected in the cell’s display text.

10) TITLE: Row/Column insertion
PRIORITY: High
ESTIMATE: 1 story point

AS A spreadsheet user
I WANT TO be able to insert a row or column
SO THAT I can add new blank rows/columns in a chosen location without having to manually move my values.

ACCEPTANCE CRITERIA:
-The user can select a cell by clicking it and have the inserted cells be inserted based on the location of the selected cell (to the right or left of the selected cell for columns and above and below the chosen cell for rows)
- The user can go to the menu bar at the top of the spreadsheet and go to the edit tab then select one of the insertion options (those options being “Insert Row Below”, “Insert Row Above”, “Insert Column Left”, and “Insert Column Right”)

- The cell references for cells that have been shifted due to column/row creation will refer to the cell that was initially selected / has been shifted (for instance, if the user included a reference to cell B2 in a cell and column B had to be shifted one to the right to accommodate a new column B, the reference will be automatically updated to C2 so it refers to the intended cell and contents).

11) TITLE: Row/Column Deletion

PRIORITY: High
ESTIMATE: 1 story point

AS A spreadsheet user
I WANT TO be able to delete a row or column
SO THAT I can remove a row/column and all of the data inside it quickly.

ACCEPTANCE CRITERIA:
- The user can select a cell or cells and have the location of the deleted rows or columns be based on the location of the selected cells
- The user can go to the menu bar at the top of the spreadsheet and go to the edit tab then select one of the deletion options (those options being “Delete Row(s)” and “Delete Column(s)”)
- If a cell that another cell reference refers to is deleted, the cell with the reference will display “#INVALREF!”
- The cell references for cells that have been shifted due to deletion of another row/column will refer to the cell that was initially selected / has been shifted (for instance, if the user included a reference to cell B2 in a cell and column B had to be shifted one to the left to accommodate the deletion of column A, the reference will be automatically updated to A2 so it refers to the intended cell and contents).

12) TITLE: Bar Graph creation

PRIORITY: Medium
ESTIMATE: 5 story points

AS A spreadsheet user
I WANT TO be able to create bar graphs using my data
SO THAT I can more clearly visualize my data and evaluations and make impactful statements with my data.

ACCEPTANCE CRITERIA:
- The user can highlight any number of consecutive cells in two neighboring columns
- The user can press the “create graph” button in the Data dropdown menu to create a bar graph of the highlighted data in the spreadsheet, where the top row of the highlighted columns becomes the axis names, the rest of the left column becomes the names of the bars, and the rest of the right column becomes the height of the bars. The top row being axis names is optional and is determined by if that row contains entirely numerical data (in which case it is not axis names) or not. If there are no provided axis names, the axes will have no names.

- The bar graph will appear in a popup window on top of the spreadsheet itself.
- The bar graph is able to automatically adjust its range/height depending on the size of the numbers in the right highlighted column.
- Clicking on a graph popup window will open a graph editing tool on the right size of the spreadsheet. This tool can be closed by clicking an X in the top right corner of the tool.
- The user can change the colors of each bar in the bar graph using color pickers in this graph editor tool. Each column name has its own respective color picker in the tool.
- At the bottom of the graph editor tool, the user can click a button to delete the graph.
- The user can customize the colors of the bars in the graph using buttons in the .
- In the case where the right column contains text (not a number) in any cell except the topmost one that becomes the axis header, the bar graph popup window will display “Invalid data provided- must be all numerical” to the user.

13) TITLE: Find and replace
PRIORITY: Medium
ESTIMATE: 3 story points

AS A spreadsheet user
I WANT TO be able to utilize a find and replace function
SO THAT I can more effectively and accurately make corrections, additions, or deletions to my data.

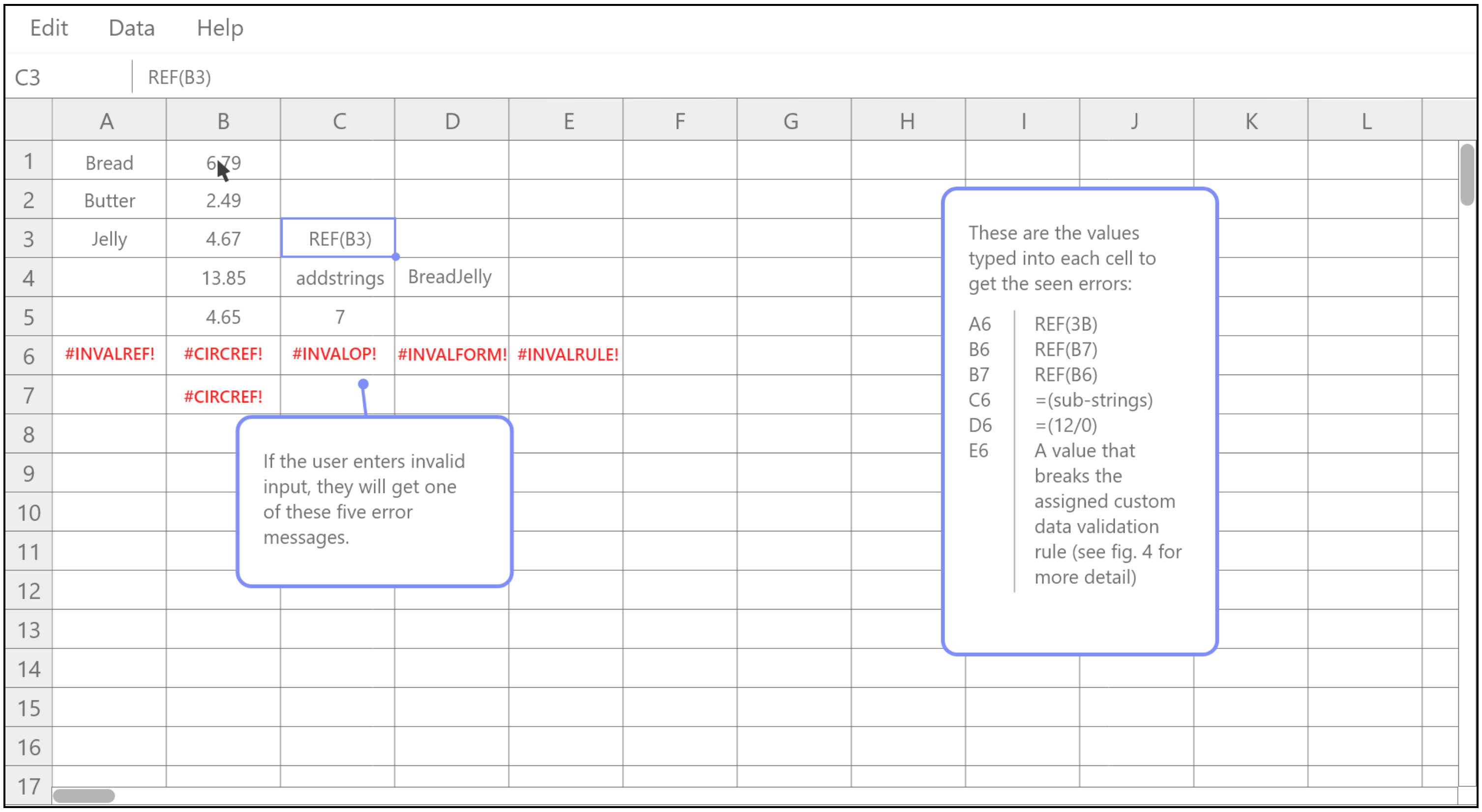
ACCEPTANCE CRITERIA:
- The user can open find and replace via a button in the Data dropdown menu in the top of the spreadsheet. The find and replace tool will open on the right side of the screen. It can be closed by clicking an “X” in the corner of the find and replace tool.
- Using a box in the tool, the user can search for the specific word. They can specify a second word in the box below in the tool.
- The first word in the spreadsheet (reading across the first row, then down to the second row, etc) that is found in a cell matching the searched word will be highlighted in yellow.
-The user can then click a button in the tool to either replace this word with the “replace” word, skip this word, or “replace all,” which will replace all matching words in the spreadsheet automatically.
- If the user skips or replaces the single word, then the next word is highlighted. This is repeated until all words that match have been highlighted or the user presses “replace all” at some point, eventually looping back to the start if necessary.
- The search is for exact, case-sensitive matches
- If the replace box is empty, pressing “replace” or “replace all” will delete the respective words and replace them with nothing.
- The find and replace search and replace boxes can also be filled with numerical constants and it will function the same (e.g. the user could search for “7” or replace with “15”).
- In the case that the data the user wants to replace is a string constant and the user wants to input a numerical constant or vice versa, the spreadsheet should be able to complete this (e.g. the user could replace “seven” with “7”).

14) TITLE: Custom data validation
PRIORITY: Medium
ESTIMATE: 5 story points

AS A spreadsheet user
I WANT TO be able to specify custom validation rules for cells
SO THAT I can be alerted when cells in my spreadsheet contain things they should not and I can ensure the accuracy and integrity of my data.

ACCEPTANCE CRITERIA:

- The user can highlight any number of consecutive cells in neighboring columns and rows
- The user can open the custom data validation tool using a button in the Data dropdown menu at the top of the spreadsheet. The custom data validation tool will open on the right side of the screen. It can be closed by clicking an “X” in the corner of the custom data validation tool.
- Using a dropdown menu in this tool, the user can select if the highlighted cells are required to be a number, a word (anything that is not just a number), or anything is legal.
- If the user chooses that the cells must be a number, they can use another dropdown to select from “equal to”, “greater than”, and “less than”, and then use a box below that to enter a number (e.g. “greater than 7” or “equal to 15”). Leaving the box blank will make this rule not apply and the highlighted cells can be any value that is legal according to the rules described above and below.
- The user can use boxes at the bottom of the tool to specify the only legal values for the highlighted cells. For instance, the user could enter “5”, “hello”, and “goodbye” in these boxes, which would require the highlighted cells to be equal to one of those 3 possible values. Leaving all of these boxes blank will make this rule not apply and the highlighted cells can be any value that is legal according to the rules described above.
-If any cell with a validation rule or rules applied to it via this method contains data that breaks a rule or rules, or later ends up with data that breaks a rule or rules (either by the user entering it or by the value of a cell being referenced changing), the cell will display “#INVALRULE” as its display text. The actual value contained within will be unaffected and is still accessible by double clicking into the cell to edit it as usual.
- Cells that have been given rules while they are highlighted maintain their rules into the future after being clicked off of, unless the rules are manually removed
- Highlighting a cell or cells that have rules applied to them and opening the validation rules tool will show the details of the rule(s) being applied to the cell or cells, provided all the highlighted cells have the same rules being applied to them. The rule(s) can then be modified using the tool.
- If the highlighted cells don’t all have the same rule(s) being applied to them, the tool will display “Error- mismatched rules” and will not allow any rules to be added or removed until the highlighted cell(s) are changed to either a single cell or all cells with matching rules.
- At the bottom of the data validation rule tool, the user can click a button to delete the current rule.



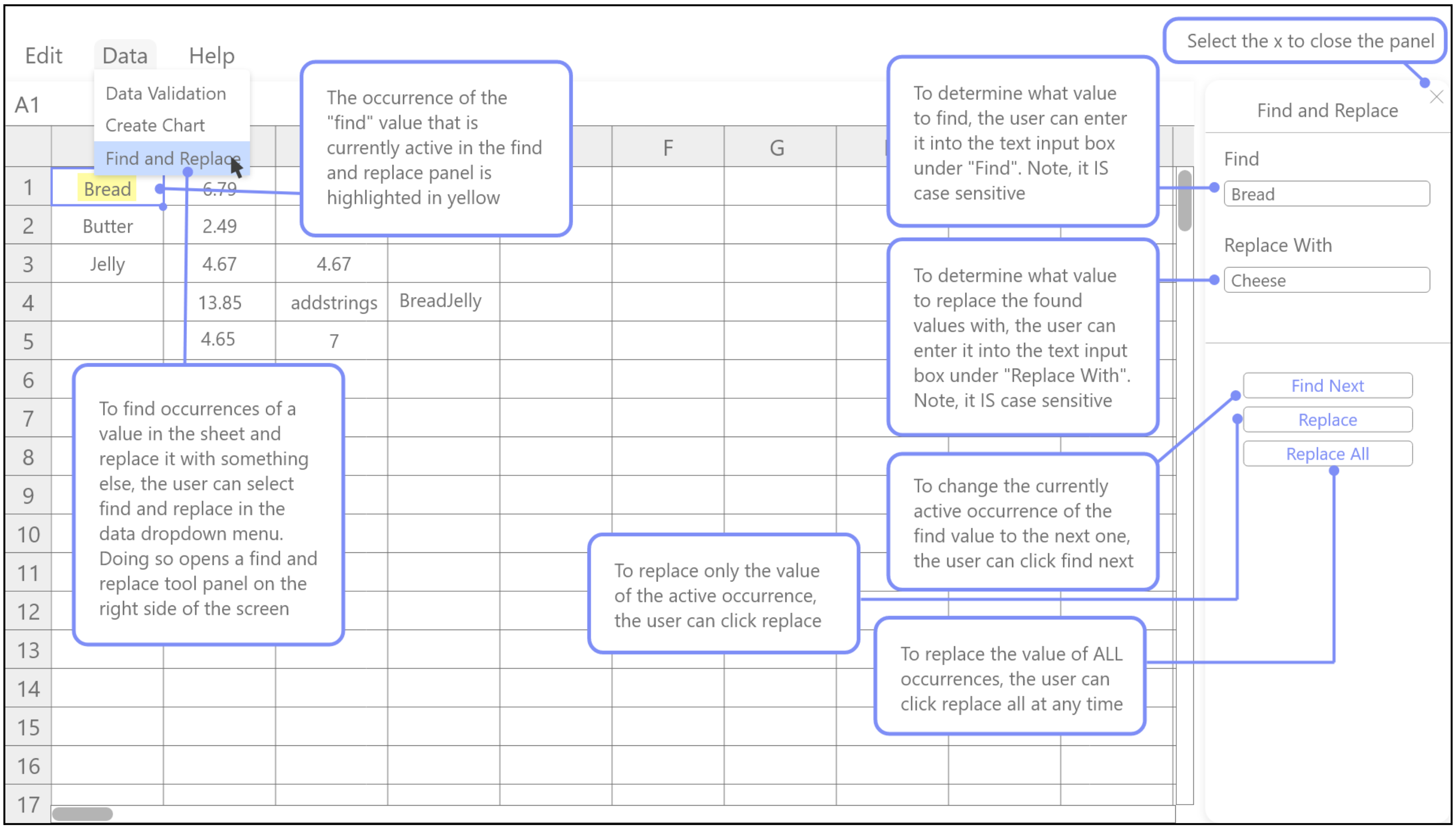
C3 REF(B3)

	A	B	C	D	E
1	Bread	6.79			
2	Butter	2.49			
3	Jelly	4.67	REF(B3)		
4		13.85	addstrings	BreadJelly	
5		4.65	7		
6	#INVALREF!	#CIRCREF!	#INVALOP!	#INVALFORM!	#INVALRULE!
7		#CIRCREF!			

If the user enters invalid input, they will get one of these five error messages.

These are the values typed into each cell to get the seen errors:

A6	REF(3B)
B6	REF(B7)
B7	REF(B6)
C6	=(sub-strings)
D6	=(12/0)
E6	A value that breaks the assigned custom data validation rule (see fig. 4 for more detail)



A1

	A	B	C	D
1	Bread	6.79		
2	Butter	2.49		
3	Jelly	4.67	4.67	
4		13.85	addstrings	BreadJelly
5		4.65	7	

Find: Bread

Replace With: Cheese

Select the x to close the panel

The occurrence of the "find" value that is currently active in the find and replace panel is highlighted in yellow

To determine what value to find, the user can enter it into the text input box under "Find". Note, it IS case sensitive

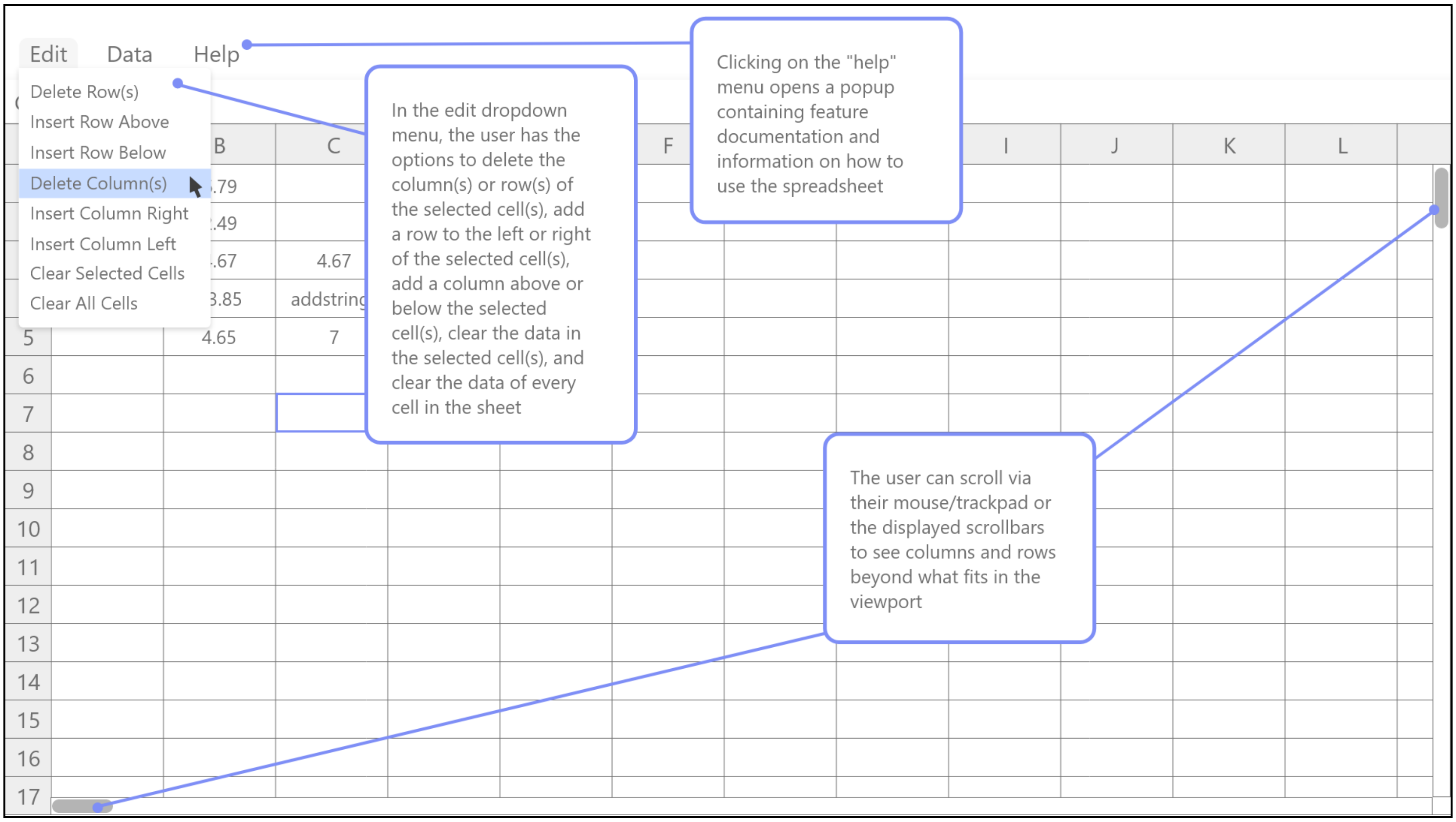
To determine what value to replace the found values with, the user can enter it into the text input box under "Replace With". Note, it IS case sensitive

To find occurrences of a value in the sheet and replace it with something else, the user can select find and replace in the data dropdown menu. Doing so opens a find and replace tool panel on the right side of the screen

To change the currently active occurrence of the find value to the next one, the user can click find next

To replace only the value of the active occurrence, the user can click replace

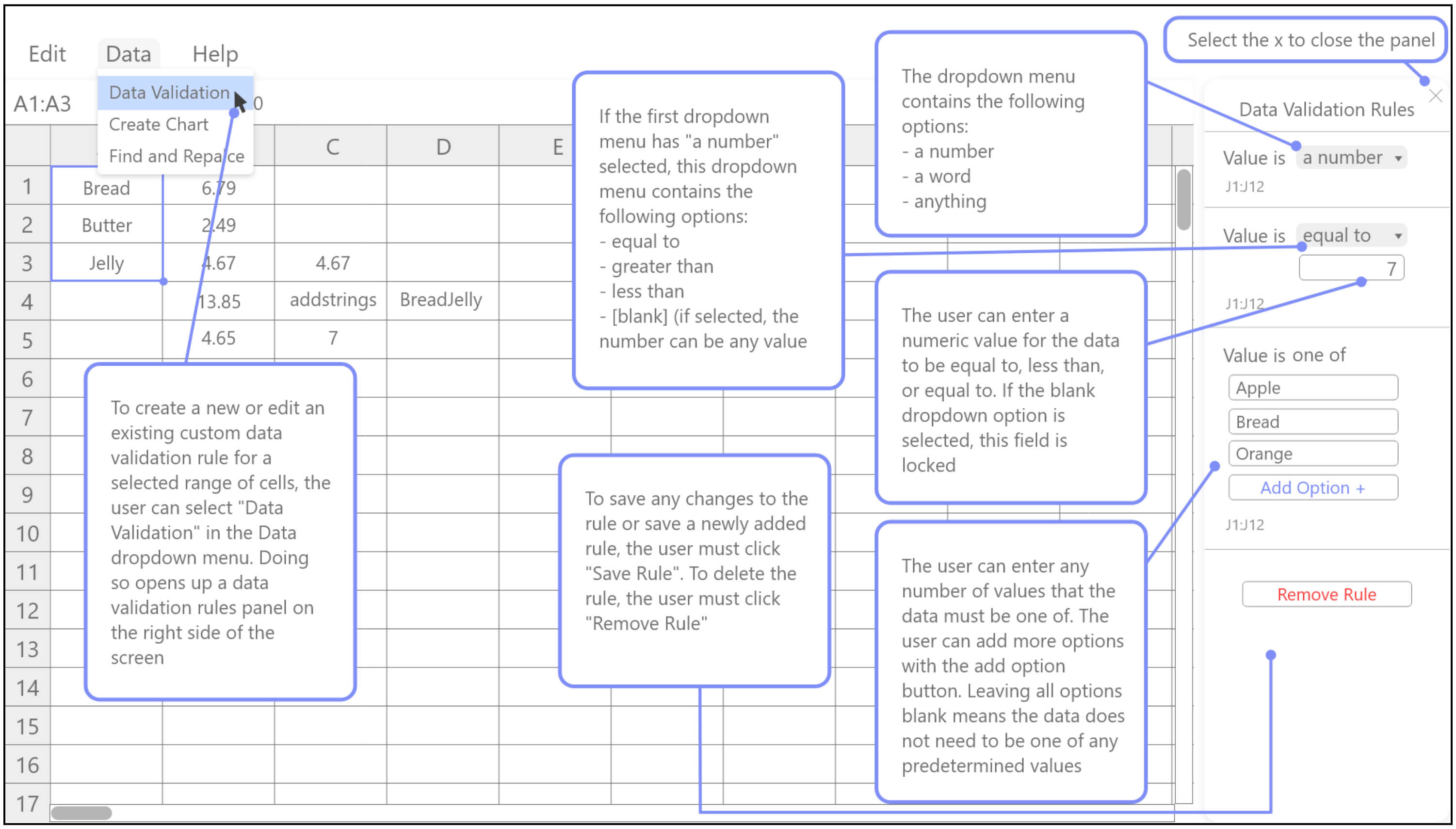
To replace the value of ALL occurrences, the user can click replace all at any time



Clicking on the "help" menu opens a popup containing feature documentation and information on how to use the spreadsheet

In the edit dropdown menu, the user has the options to delete the column(s) or row(s) of the selected cell(s), add a row to the left or right of the selected cell(s), add a column above or below the selected cell(s), clear the data in the selected cell(s), and clear the data of every cell in the sheet

The user can scroll via their mouse/trackpad or the displayed scrollbars to see columns and rows beyond what fits in the viewport



Select the x to close the panel

The dropdown menu contains the following options:
- a number
- a word
- anything

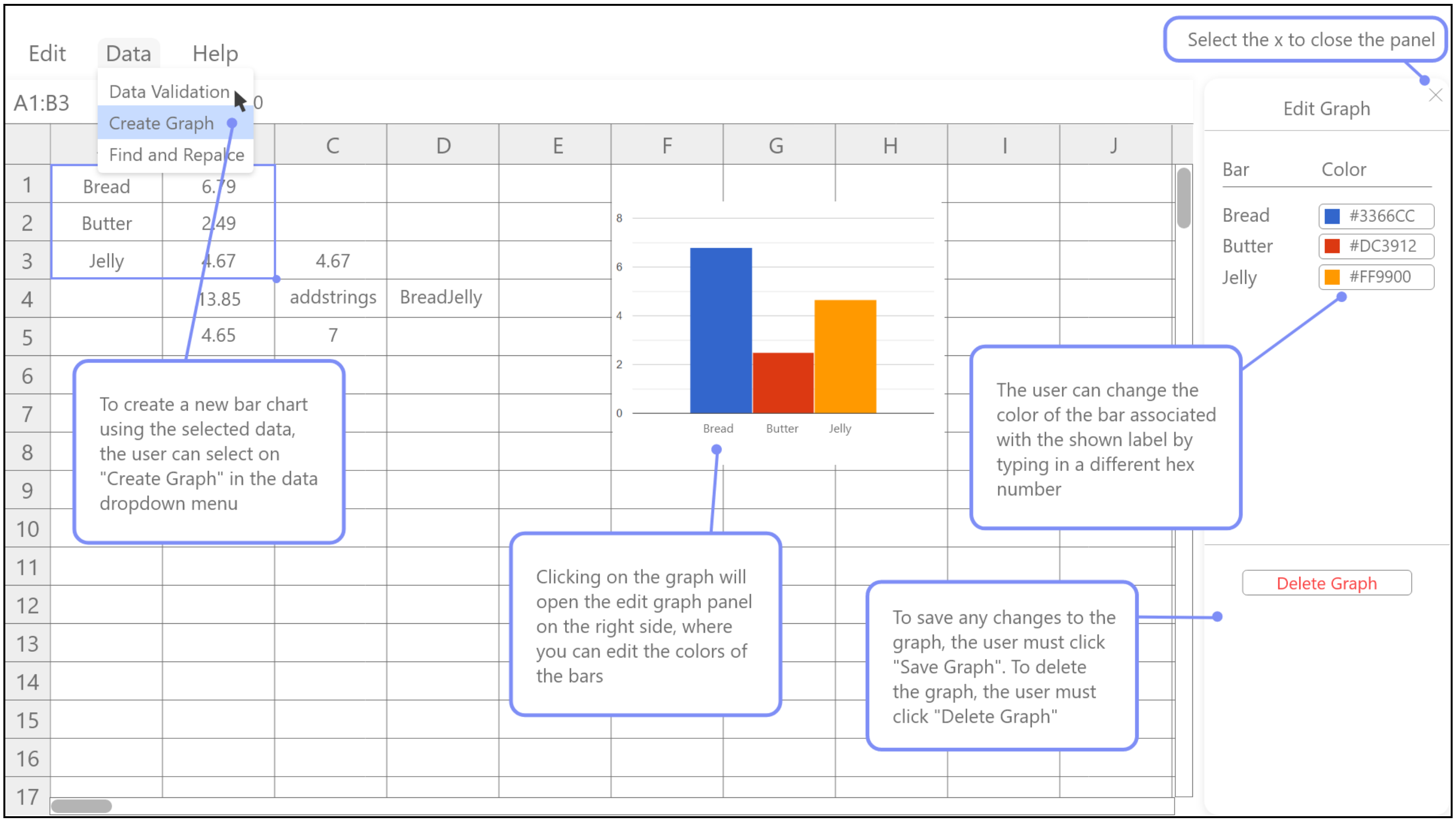
If the first dropdown menu has "a number" selected, this dropdown menu contains the following options:
- equal to
- greater than
- less than
- [blank] (if selected, the number can be any value

The user can enter a numeric value for the data to be equal to, less than, or equal to. If the blank dropdown option is selected, this field is locked

To create a new or edit an existing custom data validation rule for a selected range of cells, the user can select "Data Validation" in the Data dropdown menu. Doing so opens up a data validation rules panel on the right side of the screen

To save any changes to the rule or save a newly added rule, the user must click "Save Rule". To delete the rule, the user must click "Remove Rule"

The user can enter any number of values that the data must be one of. The user can add more options with the add option button. Leaving all options blank means the data does not need to be one of any predetermined values



	A	B	C	D
1	Bread	6.79		
2	Butter	2.49		
3	Jelly	4.67	4.67	
4		13.85	addstrings	BreadJelly
5		4.65	7	

Bar	Color
Bread	🟦 #3366CC
Butter	🟥 #DC3912
Jelly	🟧 #FF9900

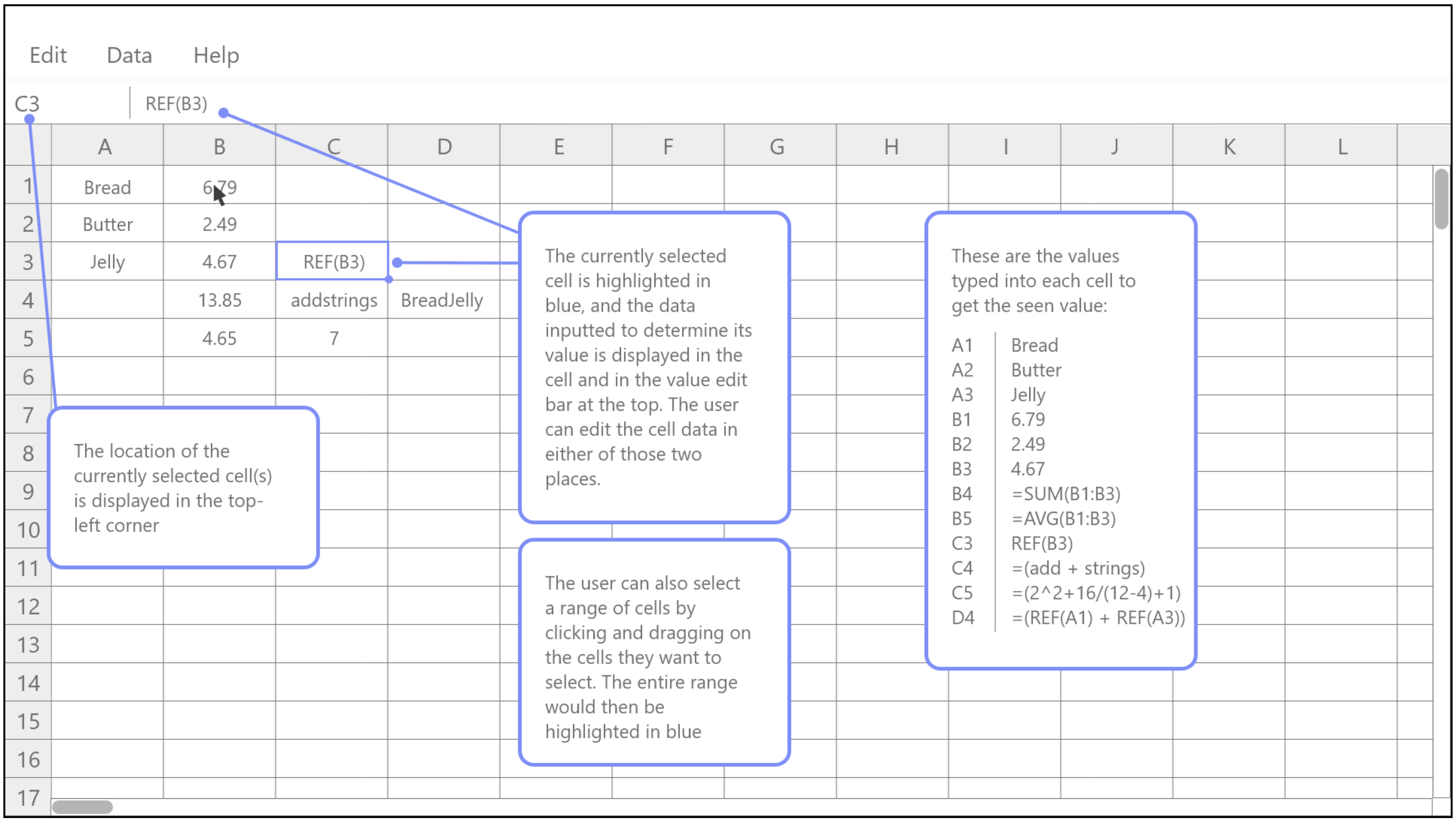
Select the x to close the panel

To create a new bar chart using the selected data, the user can select on "Create Graph" in the data dropdown menu

Clicking on the graph will open the edit graph panel on the right side, where you can edit the colors of the bars

The user can change the color of the bar associated with the shown label by typing in a different hex number

To save any changes to the graph, the user must click "Save Graph". To delete the graph, the user must click "Delete Graph"



	A	B	C	D
1	Bread	6.79		
2	Butter	2.49		
3	Jelly	4.67	REF(B3)	
4		13.85	addstrings	BreadJelly
5		4.65	7	

The location of the currently selected cell(s) is displayed in the top-left corner

The currently selected cell is highlighted in blue, and the data inputted to determine its value is displayed in the cell and in the value edit bar at the top. The user can edit the cell data in either of those two places.

The user can also select a range of cells by clicking and dragging on the cells they want to select. The entire range would then be highlighted in blue

These are the values typed into each cell to get the seen value:

A1	Bread
A2	Butter
A3	Jelly
B1	6.79
B2	2.49
B3	4.67
B4	=SUM(B1:B3)
B5	=AVG(B1:B3)
C3	REF(B3)
C4	=(add + strings)
C5	=(2^2+16/(12-4)+1)
D4	=(REF(A1) + REF(A3))

NOTE: The most important interfaces and classes are included below. As outlined in Piazza post #143, we selected the most important interfaces and classes to add to this document, while the rest of the classes and interfaces are included in the Git Repo: https://github.com/neu-cs4530-fall2023/team508-project

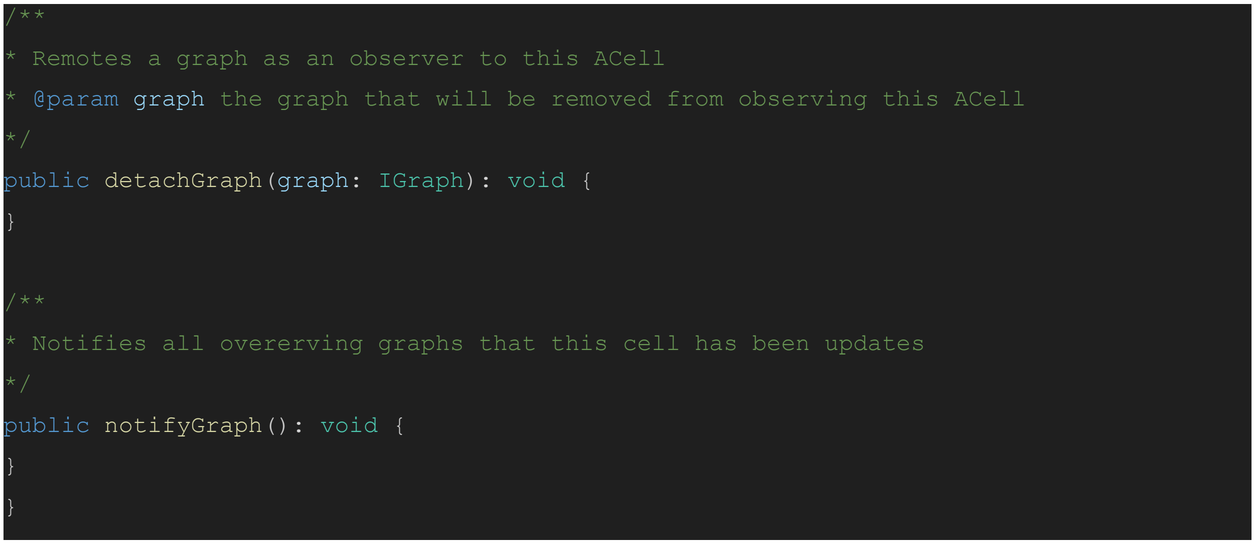
```
import { ACell } from "./cell-abstract-class";
import { IValidationRule } from "./validation-rule-interface";

/**
* Represents the main controller for the spreadsheet application
*/
export interface IController {

/**
* Adds a new row to the spreadsheet
* @param rowId the id representing where to insert the new row
*/
addRow(rowId: number): void;

/**
* Adds a new column to the spreadsheet
* @param colId the id representing where to insert the new column
*/
addColumn(colId: string): void;

/**
* Removes a row from the spreadsheet
* @param rowId the id representing what row to delete
*/
deleteRow(rowIdes: Array<number>): void;

/**
* Removes a column from the spreadsheet
* @param colId the id representing what column to delete
*/
deleteColumn(colIdes: Array<string>): void;

/**
* Changes the value of a cell
* @param cellId the id of the cell to change
* @param newValue the new value of the cell
*/
editCell(cellId: number, newValue: any): void;
```

```
/**
* Removes the value for a selection of cells
* @param cellIds the ids of the cells to clear
*/
clearCells(cellIds: Array<number>): void;

/**
* Adds a validation rule to a cell
* @param cellId the id for the cell to set
* @param rule the new rule that the value must adhere to
*/
createRule(cellId: number, rule: IValidationRule): void;

/**
* Removes a validation rule from a cell
* @param cellId the id for the cell to set
* @param rule the rule that should no longer apply
*/
removeRule(cellId: number, rule: IValidationRule): void;
/**
* Finds where a value is present and replaces it with a new value at the selected id
* @param find the value to find
* @param replace the value to change to
* @param id the id of the position of the found value
*/
findAndReplace(find: string, replace: string, id: number): void;
/**
* Finds where a value is present and replaces all instances of it with a new value at
the selected id
* @param find the value to find
* @param replace the value to change to
*/
findAndReplaceAll(find: string, replace: string): void;

/**
* Creates a graph based on a selection of cells
* @param cells the cells to base the graph on
*/
createGraph(cells: Array<ACell>): void;

/**
```

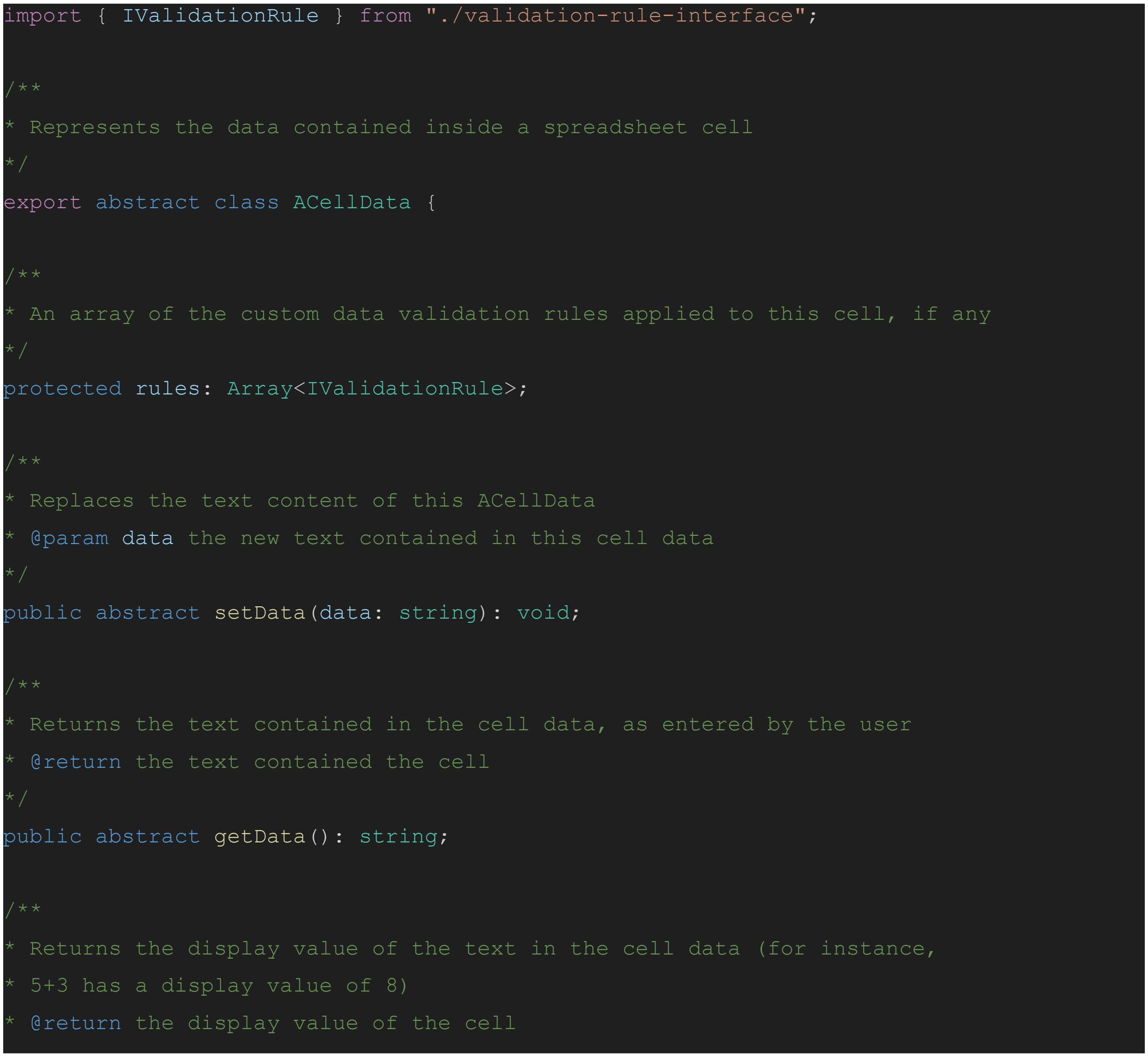
```
* Deletes a graph from the spreadsheet
* @param graphId the id of the graph to remove
*/
deleteGraph(graphId: number): void;

/**
* Sets a graph x axis name
* @param id the id of the graph to be renamed
* @param name the new name
*/
setGraphXAxisName(id: number, name: string): void;

/**
* Sets a graph y axis name
* @param id the id of the graph to have its x axis renamed
* @param name the new name
*/
setGraphYAxisName(id: number, name: string): void;

/**
* Sets a graph name
* @param id the id of the graph to have its y axis renamed
* @param name the new name
*/
setGraphName(id: number, name: string): void;
}
```

```
import { IGraph } from "./graph-interface";
import { IValidationRule } from "./validation-rule-interface";

/**
* Represents a spreadsheet cell
*/
export abstract class ACell {
/**
* Clear the content of the cell
*/
public abstract clearCell(): void;

/**
* Replaces the content of a cell
* @param newValue the new content ot the cell
```

```
*/
public abstract editCell(newValue: string): void;

/**
* Returns the true content of the cell in string form
* @returns what has been typed into the cell
*/
public abstract getCellContent(): string;

/**
* Returns what the cell displays
* @returns the display value of the cell
*/
public abstract getCellDisplay(): string;

/**
* Adds a validation rule to this cell
* @param rule the rule to add to this cell
* @returns the display value of the cell
*/
public abstract createRule(rule: IValidationRule);

/**
* Removes a validation rule from this cell
* @param rule the rule to remove from this cell
* @returns the display value of the cell
*/
public abstract removeRule(rule: IValidationRule);

/**
* Determines whether the math in the cell is a calculation or
* concatenation and performs it, updating the display value accordingly
*/
public abstract calculateOrConcatenate(): void;

/**
* Adds a graph as an observer to this ACell
* @param graph the graph that observes this cell
*/
public attachGraph(graph: IGraph): void {
}
```



```
/**
* Remotes a graph as an observer to this ACell
* @param graph the graph that will be removed from observing this ACell
*/
public detachGraph(graph: IGraph): void {
}

/**
* Notifies all overerving graphs that this cell has been updates
*/
public notifyGraph(): void {
}
}
```



```
import { IValidationRule } from "./validation-rule-interface";

/**
* Represents the data contained inside a spreadsheet cell
*/
export abstract class ACellData {

/**
* An array of the custom data validation rules applied to this cell, if any
*/
protected rules: Array<IValidationRule>;

/**
* Replaces the text content of this ACellData
* @param data the new text contained in this cell data
*/
public abstract setData(data: string): void;

/**
* Returns the text contained in the cell data, as entered by the user
* @return the text contained the cell
*/
public abstract getData(): string;

/**
* Returns the display value of the text in the cell data (for instance,
* 5+3 has a display value of 8)
* @return the display value of the cell
*/
public abstract getDisplayValue(): string;


/**
* Adds a rule to the ACellData's array of data validation rules
* @param rule the rule to be added
*/
public addRule(rule: IValidationRule): string {
return "";
}


/**
* Removes a rule to the ACellData's array of data validation rules
* @param rule the rule to be removed
*/
public removeRule(rule: IValidationRule): string {
return "";
}
}
```

```
/**
* Represents a graph that displays data in chosen categories
*/
export interface IGraph {
/**
* Sets the name of the graph's X axis
* @param name the new name of the X axis
*/
setXAxisName(name: string): void;

/**
* Sets the name of the graph's Y axis
* @param name the new name of the Y axis
*/
setYAxisName(name: string): void;

/**
* Sets the name of the graph
* @param name the new name of the overall graph
*/
setGraphName(name: string): void;

/**
* Updates the graph based on the current data in the cells it is observing
*/
updateGraph(): void;
}
```

```
import { IACellsIterator } from "./cell-iterator-interface";

/**
* Represents an object that performs a find and replace operation on a representation
of a group of spreadsheet cells
*/
export interface IFindAndReplace {
/**
* Returns an iterator over a representation of a group of spreadsheet cells
* @return the iterator object
*/
createCellsIterator(): IACellsIterator;

/**
* Replace a cell's value with a new value
* @param id the id of the cell to have its value replaced
* @param value the new value of the cell
*/
replace(id: number, value: string): void;
}
```

```
import { ACell } from "./cell-abstract-class";

/**
* Represents an iterator that iterates over a 2D array of ACells
*/
export interface IACellsIterator {
/**
* Resets the iterator to the first cell in the array of ACells
*/
first(): void;

/**
* Advances the iterator to the next cell in the array of ACells
*/
next(): void;

/**
* Returns true if the current ACell is the last cell in the array, false otherwise
* @return true if the current ACell is the last cell in the array, false otherwise
*/
isDone(): boolean;

/**
* Returns the current ACell the iterator is at
* @return the current ACell the iterator is at
*/
currentCell(): ACell;
}
/**
* Represents a data validation rule for a cell that determines if the data in
* that cell is valid or not
*/
export interface IValidationRule {

/**
* Is the cell data this rule is applied to valid or invalid according to the rule?
* @param cellData the data in a cell to be tested
* @return true if the data is valid, false if it is not
*/
checkRule(cellData: string): boolean;
}
```