TextExcel Design and Test Plan

Note: You should replace all the examples highlighted in yellow below with details specific to your program and turn off the yellow highlight.

|  |  |  |  |
| --- | --- | --- | --- |
| Property | Value | Property | Value |
| Program | TextExcel | Doc Status | Draft 5 |
| Doc Owner | Jase Calloway | Milestone 5 | Checkpoint 5 |
| Included files | Spreadsheet, Program, Cell, Dates, Strings, Doubles, Formulas, InputValueStorer | | |
|  |  | | |

# Checkpoint 1

## Design decisions and justifications:

Note: Include your design decisions and justifications here

* TextExcel CP1 uses 4 classes *Program*, *Spreadsheet*, and *Cell and Gridlock*
* The *sheet* array encapsulated in *Spreadsheet* consists of *Cell* object references
* *Program* includes the main method and runs the command loop
* *Spreadsheet* includes a public method *print*, used to display the entire sheet
* *Cell* is an interface that is used to format the text inside the cell(Not used in this checkpoint)
* The Gridlock interface contains methods like getRow and getCol that will be used for later checkpoints if the user were to ask for a table that is the size she wants

## Supported functionality:

Note: include the functionality you support here … for example

* My *Program* class includes a command loop that handles the PRINT and EXIT commands.
* My *Spreadsheet* is able to create a new sheet with of 7 columns and 10 rows (I couldn't get the letters and numbers to line up with the grid properly so I just left it with the grid only).
* My *Spreadsheet* class includes a print method which prints the grid
* The Gridlock interface successfully inherits its methods to the spreadsheet class

## Test Design:

* ***Class Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name | Test Case | Expected Result | Pass/Fail |
| Gridlock | Inherits methods to other classes | Inherited methods to other classes | PASS |
| Program | Scanner successfully used | Scanner properly asks for user's input | PASS |
| Spreadsheet | Creates new spreadsheet | Created new spreadsheet | PASS |
| Spreadsheet | Print method no errors | No errors in the code | PASS |
| Gridlock | No code in the methods | None | PASS |
| Program | Calls methods from other classes | Calls other methods | PASS |

* ***Supported Command Tests:***

|  |  |  |
| --- | --- | --- |
| Command | Expected Result | Pass/Fail |
| PRINT | Displays the entire spreadsheet and can be called successively multiple times. | PASS |
| print and exit | Case insensitive commands supported. Exits the prompt without causing an error. | PASS |
| Grid part 1 | Grid prints out with 7 rows and 10 columns | PASS |
| Grid part 2 | Grid has A-G and 1-10 | FAIL |
| While loop | While loop does not trigger an infinite loop | PASS |
| Scanner | No error with scanner | PASS |
| Compilation | No errors occur in the code | PASS |

# Checkpoint 2

## Design decisions and justifications:

Note: Include your design decisions and justifications here

* ADDED 3 SEPARATE CELL CLASSES THAT EXTEND THE CELL CLASS (Strings, Dates, Doubles
* ADDED THE SET AND GET COMMAND INTO THE CODE
* GRID NOW HAS TABLE HEADERS ON IT
* ALL THE ERRORS IN CHECKPOINT 1 ARE FIXED

## Supported functionality:

Note: include the functionality you support here … for example

* SETS CELL TO A VALUE ( A1 = “HELLO” or A1 = 5 or A1 = 01/23/1974)
* RETRIEVE A CELL’S VALUE (A1)
* Print command
* Exit command

## Test Design:

* ***Class Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name | Test Case | Expected Result | Pass/Fail |
| Program | Calls Process Method correctly | Calls correctly | PASS |

* ***Command Tests:***

|  |  |  |
| --- | --- | --- |
| Command | Expected Result | Pass/Fail |
| SET METHOD | WORKS FINE | PASS |
| GET METHOD | WORKS FINE | PASS |

* ***Supported Functionality Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Functionality | Test Case | Expected Result | Pass/Fail |
| Displaying empty cells | Use PRINT with no values in any cell in the spreadsheet | All cells display 12 spaces | PASS |
| Value of an empty cells | Enter a cell identifier | **<empty>** is displayed after the command prompt | PASS |
| Value of a cell | ENTER A CELL NAME | CELL VALUE IS DISPLAYED | PASS |
| exit command | Exits the TextExcel Program | Exits well | PASS |
| CAN GO THROUGH WHOLE ARRAY | CAN GO THROUGH WHOLE ARRAY | Cannot go past 9 in row for some reason | Fail |

# Checkpoint 3

## Design decisions and justifications:

Note: Include your design decisions and justifications here

* Added a clear function to the Spreadsheet class
* Added getSaveData to Spreadsheet class
* Added loadFrom to Spreadsheet class

## Supported functionality:

Note: include the functionality you support here … for example

* Spreadsheet now properly truncates excessively long inputs with a “>” at the end
* Spreadsheet properly clears

## Test Design:

* ***Class Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name | Test Case | Expected Result | Pass/Fail |
| Program | Clear function called | Clears spreadsheet | PASS |

* ***Command Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Command | Expected Result | Result | Pass/Fail |
| Program | Save function works | Does not work (idk why it doesn’t) | Fail |
| Program | Load function works | Does not work(idk what goes wrong) | Fail |

* ***Supported Functionality Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Functionality | Test Case | Expected Result | Pass/Fail |
| Previous code | All previous code in Checkpoint 1 is fixed | Fixed | Pass |
| Previous code | Much of the previous code in Checkpoint 2 is fixed as stated in the rubric | Fixed | Pass |

# Checkpoint 4

## Design decisions and justifications:

Note: Include your design decisions and justifications here

* Removed the inputParsing class because it wasn’t needed for optimization
* Added formulas because it is supposed to be there
* Fixed code in classes and spreadsheet

## Supported functionality:

Note: include the functionality you support here … for example

* Formulas are now supported (Type in like this A1 = (1 + 1)
* Clear individual cells is now supported
* Fixed the Save and Load methods and they work now

## Test Design:

* ***Class Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name | Test Case | Expected Result | Pass/Fail |
| Formula | Evaluate works | works | Pass |

* ***Command Tests:***

|  |  |  |
| --- | --- | --- |
| Command | Expected Result | Pass/Fail |
| evaluate | A1 = (1 + 1) | Pass |
| evaluate | A1 = (1 \* 5) | pass |
| evaluate | A1 = (9 – 6) | pass |
| evaluate | A1 = (4 / 2) | pass |

* ***Supported Functionality Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Functionality | Test Case | Expected Result | Pass/Fail |
| Save | save fileA | Save works | pass |
| Load | load fileA | Load works | pass |
| evaluate | A1 = (a + b) | Evaluate works | pass |

# Checkpoint 5

## Design decisions and justifications:

Note: Include your design decisions and justifications here

* Tried to change up the spreadsheet to accommodate the sum and avg but failed

## Supported functionality:

Note: include the functionality you support here … for example

* Nothing new. I gave up.
* Multiple Operations for formula cells works

## Test Design:

* ***Class Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name | Test Case | Expected Result | Pass/Fail |
| Sum | Called it in main program | Did not work | Fail |

* ***Command Tests:***

|  |  |  |
| --- | --- | --- |
| Command | Expected Result | Pass/Fail |
| evaluate | A1 = (A1 + A2) | fail |
| evaluate | A1 = (A1 – A2) | fail |
| evaluate | A1 = (A1 \* A2) | fail |
| evaluate | A1 = (A1 / A2) | fail |
| sum | G10 = (sum A1 – A10) | fail |
| avg | G10 = (avg A1 – A10) | fail |

* ***Supported Functionality Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Functionality | Test Case | Expected Result | Pass/Fail |
| Evlauate | A1 = (1 + 1 + 1 + 1) | A1 = 4.0 | Pass |

# Checkpoint 6

## Design decisions and justifications:

Note: Include your design decisions and justifications here

* Deleted the InputValueStorer class
* Evaluating cell values together A1 = ( A2 + B6)
* Added sortA and sortD EVEN Though they don’t work

## Supported functionality:

Note: include the functionality you support here … for example

* Multiple Operations for formulas works now A1 = ( 5 \* 2 + 5 )

## Test Design:

* ***Class Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name | Test Case | Expected Result | Pass/Fail |
| Spreadsheet | sortA | Error | Fail |
| Spreadsheet | sortD | Error | Fail |
| Formulas | Doing math on cells | Works | Pass |
| Formulas | Multiple Operations | Works | Pass |

* ***Command Tests:***

|  |  |  |
| --- | --- | --- |
| Command | Expected Result | Pass/Fail |
| Cell Setting | A1 = ( 5 + 5 ) | Pass |
| Cell Setting | A1 = 5 A2 = ( A1 \* 5 ) | Pass |
| Cell Setting | A1 = 5 A2 = 6 A3 = ( A1 \* A2 ) | Pass |
| sortA | Does not support sorting | Fail |
| sortD | Does not support sorting | Fail |

* ***Supported Functionality Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Functionality | Test Case | Expected Result | Pass/Fail |
| Everything except sorting cells | Tested it all | Worked | Pass |
| Sorting cells | Tested it. Did not work. | Failed | Fail |

# Extra Credit

## Design decisions and justifications:

Note: Include your design decisions and justifications here

* Did the helper method in the program

## Supported functionality:

Note: include the functionality you support here … for example

* Calls to help method work as planned

## Test Design:

* ***Class Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name | Test Case | Expected Result | Pass/Fail |
| Program | Help call | Prints help | Pass |

* ***Command Tests:***

|  |  |  |
| --- | --- | --- |
| Command | Expected Result | Pass/Fail |
| Help | Supported commands and formats are:  Print: Prints out the spreadsheet  Clear: Clears whole spreadsheet // Format for specific cell: Clear A1  Save: Saves your specified spreadsheet // Format: Save fileName  Load: Loads your specified spreadsheet // Format: Load fileName  Cell Setting: Sets a value to a cell // Format: A1 = 5; A1 = "Print"; A1 = 01/01/2016; A1 = ( 5 + 5 ); A1 = ( A5 + B4 );  Cell Value Retrieval: Gets the value of a cell // Format: A1  SortA and SortD do not work. Please do not try them. | Pass |

* ***Supported Functionality Tests:***

|  |  |  |  |
| --- | --- | --- | --- |
| Functionality | Test Case | Expected Result | Pass/Fail |
| help | Call in a different case like “HeLP” | Still calls help method | Pass |