Wayne State University
Department of Computer Science
Proficiency exam – Programming Skills
Winter 2020

Your goal is to develop classes that implement a simple version of a spreadsheet (e.g. Microsoft Excel) and a test program that creates and utilizes the objects of these classes. A cell in your simplified spreadsheet can store a number (integer or floating point) or a string. Specifically, you need to implement:

1. A virtual class Cell, which provides an interface for interacting with the cells of a spreadsheet through the following virtual functions:
   - **void setValue(string value)**: sets the contents of a cell to the value provided as a string argument, performing conversion (e.g. string to float), if necessary, and throwing an exception, if such conversion is not possible **[5 points]**
   - **float getFloatValue()**: returns the contents of a cell as a floating point number. The function should throw an exception, if type conversion is impossible (e.g. string of characters to float is impossible, but string of digits to float is possible) **[5 points]**

2. Classes StrCell, IntCell and FloatCell that inherit from Cell and correspond to the spreadsheet cells with strings, integers and floating point numbers **[10x3 = 30 points]**

3. Implement a class SpreadSheet that allows to store and access data in a spreadsheet of a given size. You can use any data structure of your choice to implement the container of cells in a spreadsheet. The rows in a spreadsheet are addressed with the integer numbers starting from one (i.e. 1, 2, 3, etc.) and columns are addressed with the characters starting from "A" (i.e. "A", "B", "C", …, "AA", "AB", etc.). This way the first cell in the spreadsheet will have the row index of 1 and the column index of "A". Your spreadsheet should provide the following functions:

   - constructor that creates a spreadsheet of empty cells with the given number of rows and columns **[5 points]**
   - **bool isEmpty(int row, string column):** checks whether a cell with a given row and column indices is empty or not **[5 points]**
   - **float getFloatValue(int row, string column):** returns the contents of a cell with the given row and column indices as a floating point number (performing type conversion, if necessary). The function should throw an exception, if type conversion is impossible (e.g. string of characters to float is impossible, but string of digits to float is possible) **[5 points]**
   - **bool setCellValue(int row, string col, string value, val_type valueType):** sets the value and type of a spreadsheet cell with the given row and column index to the given value provided as a string and value type provided as val_type, which is an enum of INT, FLOAT or STR. A function should throw exceptions if the row or column indices are out of bounds or if the value conversion to the target cell type is impossible (e.g. string of characters is provided for a floating point number or an integer) **[5 points]**
   - **getRange(int topLeftRow, string topLeftCol, int botRightRow, string botRightCol):** returns a one-dimensional array of the cell values as floating point numbers within a rectangle specified by the row and column indices of the top left corner and the row and column indices of the bottom right corner. Empty cells should not be included into this array **[15 points]**

- **insertColumn(string column):** adds a new column of empty cells with the specified index, shifting existing columns after this index to the right **[15 points]**

4. Implement a test program that does the following operations:

- Creates a 10 by 10 cell spreadsheet **[1 point]**
- Sets cell (11, A) to “5” and type INT (this should result in an error message “(11, A) is out of bounds”, cell (1,A) to “1” and type INT, cell (2,A) to “3” and type INT, cell (1,B) to “4.0” and type FLOAT, cell (2,B) to “a” and type INT (this should result in an error message “can’t covert ‘a’ to integer”), cell (2,B) to “6” and type INT **[6 points]**.
- Inserts a new column with index “B” into the spreadsheet **[1 point]**
- Obtains an array of values for the cells in the range from (1,A) to (2,C) and prints the contents of this array (this should print “1.0 3.0 4.0 6.0”) **[2 points]**

Instructions

- To get full credit, your code must do what is required for each task, compile and run without crashing producing the required output.
- Provide a README file along with your code, in which you will have written detailed instructions on how to compile, build and execute your code. **Leave at least 15 minutes for preparing this file!**
- Save all files (executable, source code and README) onto the provided flash drive. It is your responsibility to make sure all the files are saved and are readable.
- Do **not** write your name on the disk or put any comments in the source code that reveal your identity. Your disk will be identified based on a pre-assigned number.
- Raise your hand, if you have any questions about the problems in this exam
- No instruction on using the software development environment that you are using will be given.
- Do **not** use the Internet.
- Good luck!