CS 115 Final Project - Possible Design structure

One possible way to structure your program is to create two classes:

- `DataFrame` - which will hold individual data frame / table data,
- `DataFrameSystem` - which will populate individual `DataFrame` objects with data and handle interactions with the user.

The latter should be the one with the `main()` method.

Class `DataFrame`

The following is a basic structure you could base your implementation on. **You can add your own attributes and methods (or modify existing ones)** if you feel that your program will benefit from it.

Attributes:

Attribute name	Data Type	Valid range	Comments
`data`	`String [][]`	A 2D array with at most 25000 rows	
`columnHeadings`	`String []`	A 1D array of Strings	Number of elements has to match the number of columns
`dataTypes`	`String []`	A 1D array of Strings	Number of elements has to match the number of columns
`dataFrameName`	`String`	Non-empty string	
`numberOfRows`	`int`	Greater or equal to 0, Less or equal to 25000	
`numberOfColumns`	`int`	Greater or equal to 0	

Methods:

Method name	Access modifier	Return type	Arguments	Notes

`DataFrame`	`public`	`none`	`String frameName`
			`String [] columnHeadings`
			`String [] dataTypes`
				Parametrized constructor

`addRow`	`public`	`void`	`String newRow`	With this approach you will need to parse `newRow` to extract individual fields
`getDataFrameName`	`public`	`String`	`none`	
`getNumberOfRows`	`public`	`int`	`none`	
`getNumberOfColumns`	`public`	`int`	`none`	
`getColumnName`	`public`	`String`	`int columnID`	
`getColumnDataType`	`public`	`String`	`int columnID`	
`importNewCSVFile`	`public`	`void`	`String fileName`	This should populate your "table" with data for individual rows
`findAverage`	`public`	`double`	`String columnName`	It should display an error message for non-numeric columns. You can return zero in such case.
`findMinimum`	`public`	`double`	`String columnName`	It should display an error message for non-numeric columns. You can return zero in such case.
`findMaximum`	`public`	`double`	`String columnName`	It should display an error message for non-numeric columns.

				You can return zero in such case.
createFrequencyTable	public	int []	String columnName	It should display an error message for non-numeric columns. You can return an array of zeros in such case.
createNewDataFrame	public	DataFrame	String columnName String operator String value	You will need to create a new DataFrame type object here

Consider additional (perhaps private) auxiliary / "internal" methods, for example parseRow, etc. Whatever makes it easier. Avoid duplicating code: if you are doing exactly the same ***thing*** twice in two or more different methods, consider creating a separate method for the ***thing***.

Class DataFrametemSystem

The following is a basic structure you could base your implementation on. **You can add your own attributes and methods (or modify existing ones)** if you feel that your program will benefit from it.

Attributes:

Attribute name	Data Type	Valid range	Comments
dataFrames	DataFrame []	A 10 element 1D array of DataFrame type objects.	You can set its size to 10 and keep track of the number of data frames in your system
numberOfDataFrames	int	Greater or equal to 0, Less or equal to 10	Start with zero and increment whenever a new frame is added
activeDataFrameID	int	Greater or equal to 0, Less or equal to numberOfDataFrames	

Methods:

Method name	Access modifier	Return type	Arguments	Notes
main	public	void	String [] args	**Main method**
displayMenu	private	void	none	Displays the user menu and waits for an input
runMenuOption	private	void	int menuOption	Run individual menu option based on user input. Could be realized differently. Individual menu items may require asking user for more information (column ID, etc.)
Below you can find ***potential*** methods that you could call from your runMenuOption method				
setActiveDataFrame	private	void	int frameID	
importNewCSVFile	private	void	String fileName	Loads data into an active data frame
findAverage	private	void	String columnName	Find and display an average for columnName in active data frame
findMinimum	private	void	String columnName	Find and display an minimum for columnName in active data frame
findMaximum	private	void	String columnName	Find and display an

				maximum for `columnName` in active data frame
`createFrequencyTable`	`private`	`void`	`String columnName`	Create and display a frequency table for `columnName` in active data frame

`createNewDataFrame`	`private`	`DataFrame`	`String columnName`
			`String operator`
			`String value`
				Creates new `DataFrame` object based on `columnName` `operator`, and `value`. Add this frame to the `dataFrames` array if there is space. Otherwise, display error.

Consider additional (perhaps private) auxiliari / "internal" methods, for example `parseRow`, etc. Whatever makes it easier. Avoid duplicating code: if you are doing exactly the same ***thing*** twice in two or more different methods, consider creating a separate method for the ***thing***.

Test cases:

Re-think your test cases based on the design above. To begin with, consider:

- Illegal values,
- Boundary values (values that are close to the limit ("above and below"),
- Index out of bounds situations,
- File does not exist error,
- Potential data type mismatch,
- Null object references,
- Etc.