Linear Algebra Calculator Shell

Avraham Katz

Version 1.0

Contents:

Description:

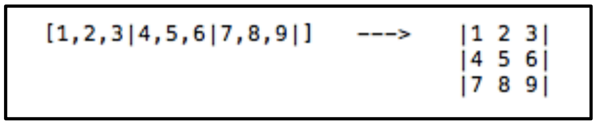
- This is a linear algebra calculator shell that performs a series of 15 linear algebra operations on matrices such as converting a matrix to reduced row echelon form, finding an inverse matrix, and finding a determinant. The program is a shell and runs through the terminal. The calculator cannot perform multiple operations in one line (see Future Features & Updates section). However, if a user inputs an equation, once the calculator returns the solution, the user can then perform an operation on that output, and can keep repeating this process. The user can also store matrices in the calculator, mapping them to a variable, and act on these matrices from the command line using the variable names. When the program is exited, matrices and variables stored in the storage are written to an external file, calculator_storage.txt. When the program is opened again, matrices in the calculator_storage.txt file are read by the program and stored in the calculator storage, available to the user for use.

Features:

- Matrix addition
- Matrix subtraction
- Matrix multiplication
- Raising a matrix to an exponent
- Scalar multiplication
- Reducing matrix to row echelon form
- Reducing matrix to reduced row echelon form
- Finding the inverse of a matrix
- Finding the transpose of a matrix
- Finding the minor of a position in a matrix
- Finding the cofactor matrix of a matrix
- Finding the adjoint matrix of a matrix
- Finding the trace of a matrix
- Finding the determinant of a matrix
- Finding the cofactor of a position in a matrix
- Easy creation of an identity matrix of any size
- Storing matrices, mapping to a variable

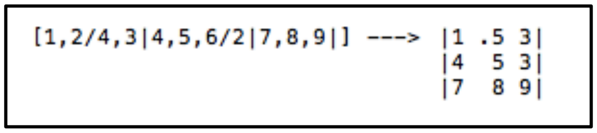
Creating & Storing Matrices:

- There are 2 types of storage in the calculator, internal and external. The internal storage only exists while the calculator is running. The external storage, stored in the file calculator_storage.txt, always exists, even while the calculator is not running.
- String representation of a matrix:
    - In the string representation of a matrix, the string begins with a '[' and ends with a ']'. A line in a matrix is written with ',' between each value (example: 1,2,3,4 == line 1 2 3 4). A '|' marks the end of a line.
    - Example of a string representation of a matrix → matrix:



```
[1,2,3|4,5,6|7,8,9|]    --->    |1 2 3|
                                |4 5 6|
                                |7 8 9|
```

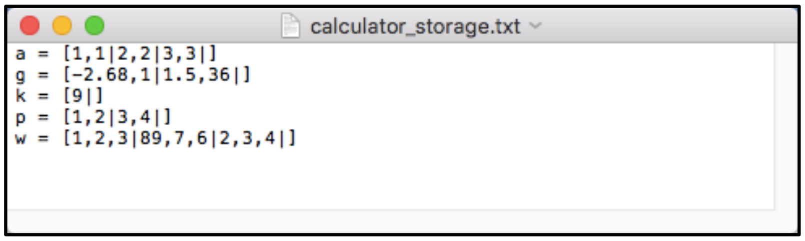
- The string representation of a matrix can also have fractions in it:



```
[1,2/4,3|4,5,6/2|7,8,9|] ---> |1 .5 3|
                              |4  5 3|
                              |7  8 9|
```

- If the user inputs an in incorrect syntax for the string representation of a matrix, or attempts to make a matrix with different sized lines, an ERROR message will appear.

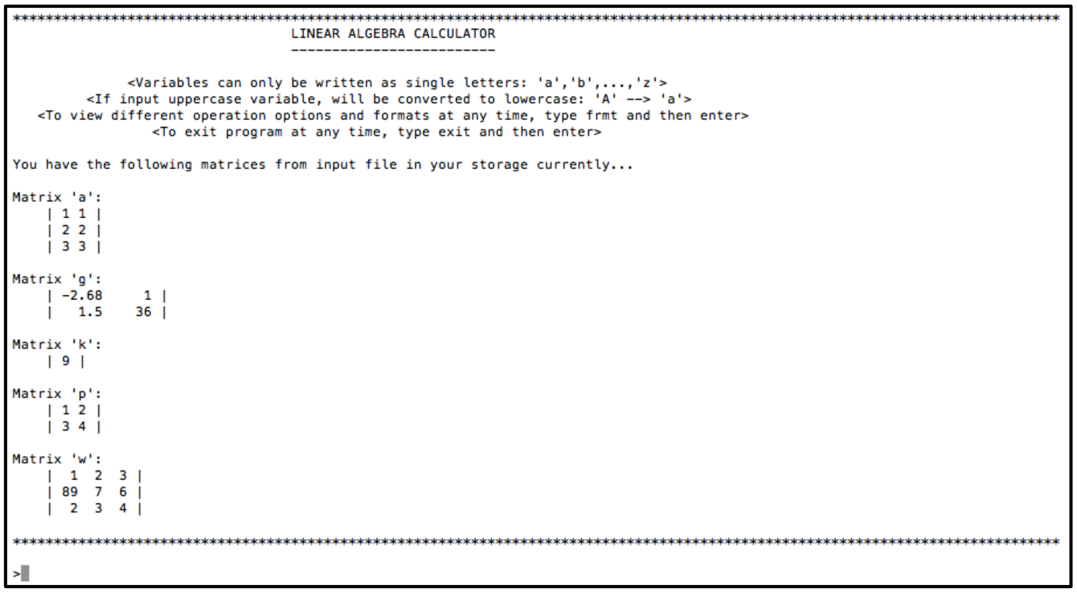
- The calculator has 26 variables to map matrices to (lowercase 'a' to lowercase 'z'). Any other attempt to create a variable, either a non-letter or multiple letter variable, will result in an ERROR message. If a user inputs a capital letter (A, B,…, Z) in an input, the calculator automatically converts it to lowercase (A→a, B→b,…, Z→z).
- There are multiple ways to store matrices in the calculator storage:
    - From the calculator_storage.txt file
        - On separate lines declare 'variable' = 'matrix in string representation':



calculator_storage.txt

```
a = [1,1|2,2|3,3|]
g = [-2.68,1|1.5,36|]
k = [9|]
p = [1,2|3,4|]
w = [1,2,3|89,7,6|2,3,4|]
```

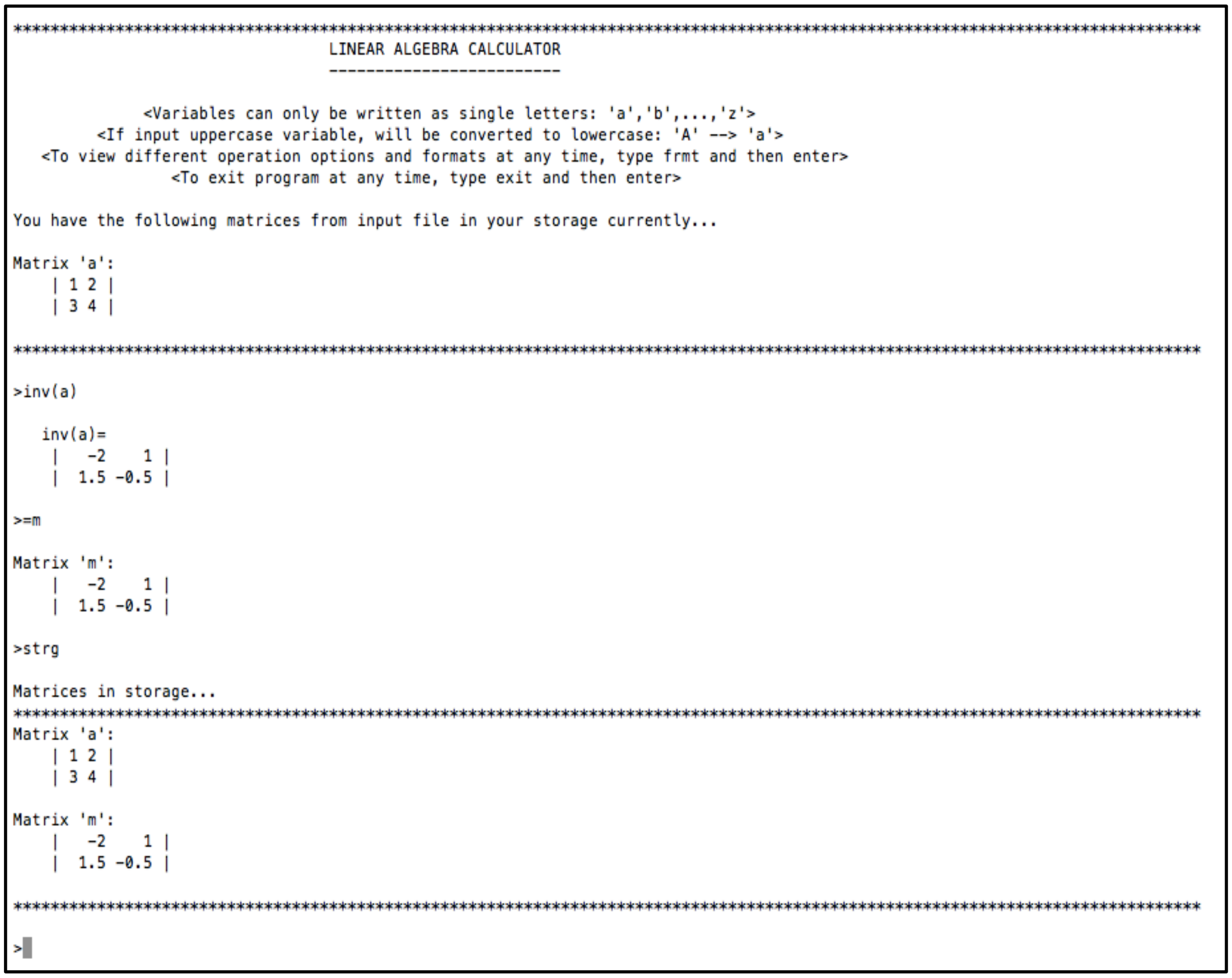
When the calculator opens, will display the following…



```
************************************************************************************************************************************
                                    LINEAR ALGEBRA CALCULATOR
                                    -------------------------

                     <Variables can only be written as single letters: 'a','b',...,'z'>
                <If input uppercase variable, will be converted to lowercase: 'A' --> 'a'>
    <To view different operation options and formats at any time, type frmt and then enter>
                        <To exit program at any time, type exit and then enter>

You have the following matrices from input file in your storage currently...

Matrix 'a':
    | 1 1 |
    | 2 2 |
    | 3 3 |

Matrix 'g':
    | -2.68     1 |
    |   1.5    36 |

Matrix 'k':
    | 9 |

Matrix 'p':
    | 1 2 |
    | 3 4 |

Matrix 'w':
    |  1  2  3 |
    | 89  7  6 |
    |  2  3  4 |

************************************************************************************************************************************
>
```

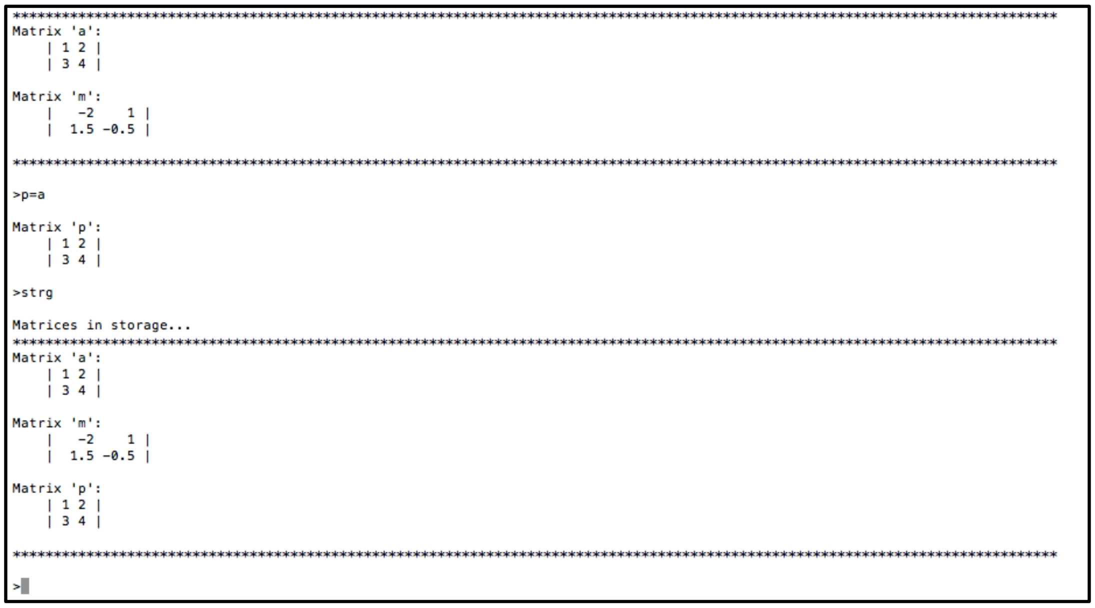
- During normal calculator operations
    - At any time during calculator operations, the user can declare a matrix to be stored under a variable. For example, if after opening the calculator, the user wrote the command "b = [1.1, 2.2| 3.3, 4.4|]", the storage would then hold matrix 'b'.

- After the calculator returns an answer to an operation input by user, the user on the next line can write "= 'variable name'". This will store the last output in connection with the variable name. In the following example, the user started the program with matrix 'a' in storage. They then did the command "inv(a)", finding the inverse matrix of 'a'. They then wrote "=m", declaring that "inv(a)" should be stored under the variable 'm'. Lastly, after typing in the command "strg" to view the calculators storage, it can be seen that both matrix 'a' and 'm' are in the storage.



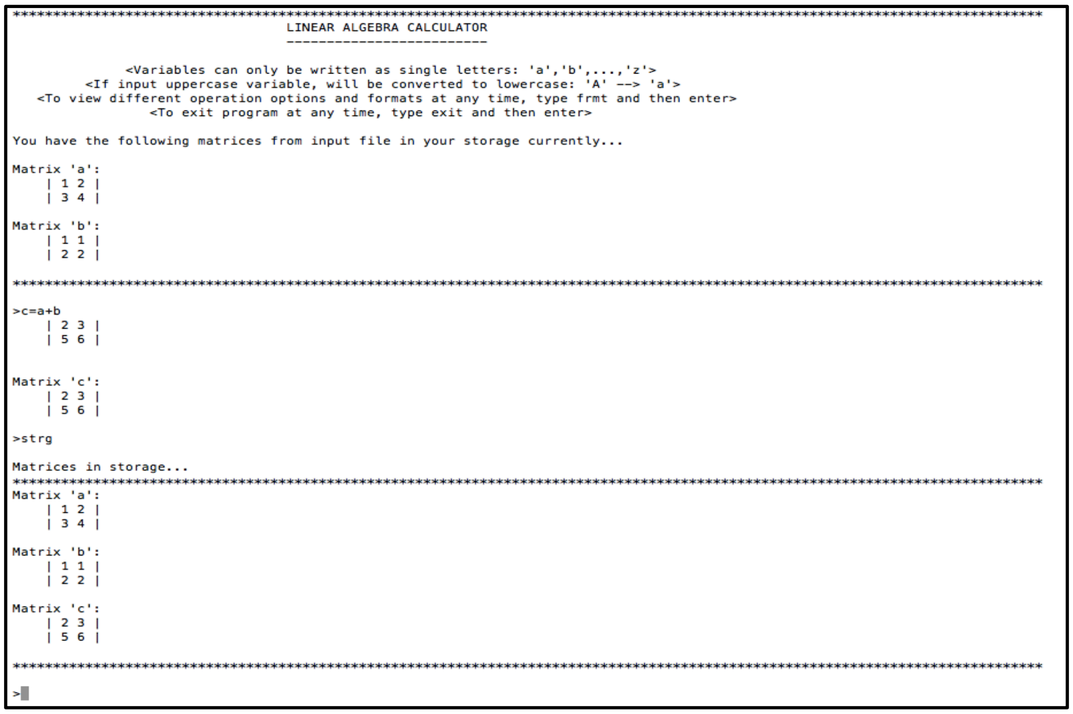
```
*****************************************************************************************************************************
                                  LINEAR ALGEBRA CALCULATOR
                                  -------------------------

              <Variables can only be written as single letters: 'a','b',...,'z'>
         <If input uppercase variable, will be converted to lowercase: 'A' --> 'a'>
   <To view different operation options and formats at any time, type frmt and then enter>
                 <To exit program at any time, type exit and then enter>

You have the following matrices from input file in your storage currently...

Matrix 'a':
    | 1 2 |
    | 3 4 |

*****************************************************************************************************************************

>inv(a)

   inv(a)=
    |   -2    1 |
    |  1.5 -0.5 |

>=m

Matrix 'm':
    |   -2    1 |
    |  1.5 -0.5 |

>strg

Matrices in storage...
*****************************************************************************************************************************
Matrix 'a':
    | 1 2 |
    | 3 4 |

Matrix 'm':
    |   -2    1 |
    |  1.5 -0.5 |

*****************************************************************************************************************************

>
```

- If a variable is already in storage, for this example 'a', user can write " 'variable name' = a ", and calculator will store values of 'a' into new variable too. (NOTE: if a subsequent operation is done on matrix 'a', this will not affect the new variable, the new variable is just created with the same values of the current 'a', but is not dependent on 'a' after its declaration. Continuing example from previous image, in this example user wrote "p=a". Now looking in storage, both 'a' and 'p' have same values:



```
*********************************************************************************************************************
Matrix 'a':
    | 1 2 |
    | 3 4 |

Matrix 'm':
    |   -2    1 |
    |  1.5 -0.5 |

*********************************************************************************************************************

>p=a

Matrix 'p':
    | 1 2 |
    | 3 4 |

>strg

Matrices in storage...
*********************************************************************************************************************
Matrix 'a':
    | 1 2 |
    | 3 4 |

Matrix 'm':
    |   -2    1 |
    |  1.5 -0.5 |

Matrix 'p':
    | 1 2 |
    | 3 4 |

*********************************************************************************************************************

>
```

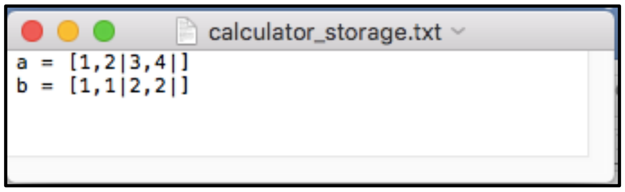
- The last way to declare a variable and store a matrix is to input 'variable name' = 'matrix operation'. For example, if storage has matrix 'a' and 'b', user can write "c=a+b" and 'c' will equal the output of "a + b":



```
*********************************************************************************************************************
                                    LINEAR ALGEBRA CALCULATOR
                                    -------------------------

               <Variables can only be written as single letters: 'a','b',...,'z'>
          <If input uppercase variable, will be converted to lowercase: 'A' --> 'a'>
    <To view different operation options and formats at any time, type frmt and then enter>
                      <To exit program at any time, type exit and then enter>

You have the following matrices from input file in your storage currently...

Matrix 'a':
    | 1 2 |
    | 3 4 |

Matrix 'b':
    | 1 1 |
    | 2 2 |

*********************************************************************************************************************

>c=a+b
    | 2 3 |
    | 5 6 |


Matrix 'c':
    | 2 3 |
    | 5 6 |

>strg

Matrices in storage...
*********************************************************************************************************************
Matrix 'a':
    | 1 2 |
    | 3 4 |

Matrix 'b':
    | 1 1 |
    | 2 2 |

Matrix 'c':
    | 2 3 |
    | 5 6 |

*********************************************************************************************************************

>
```

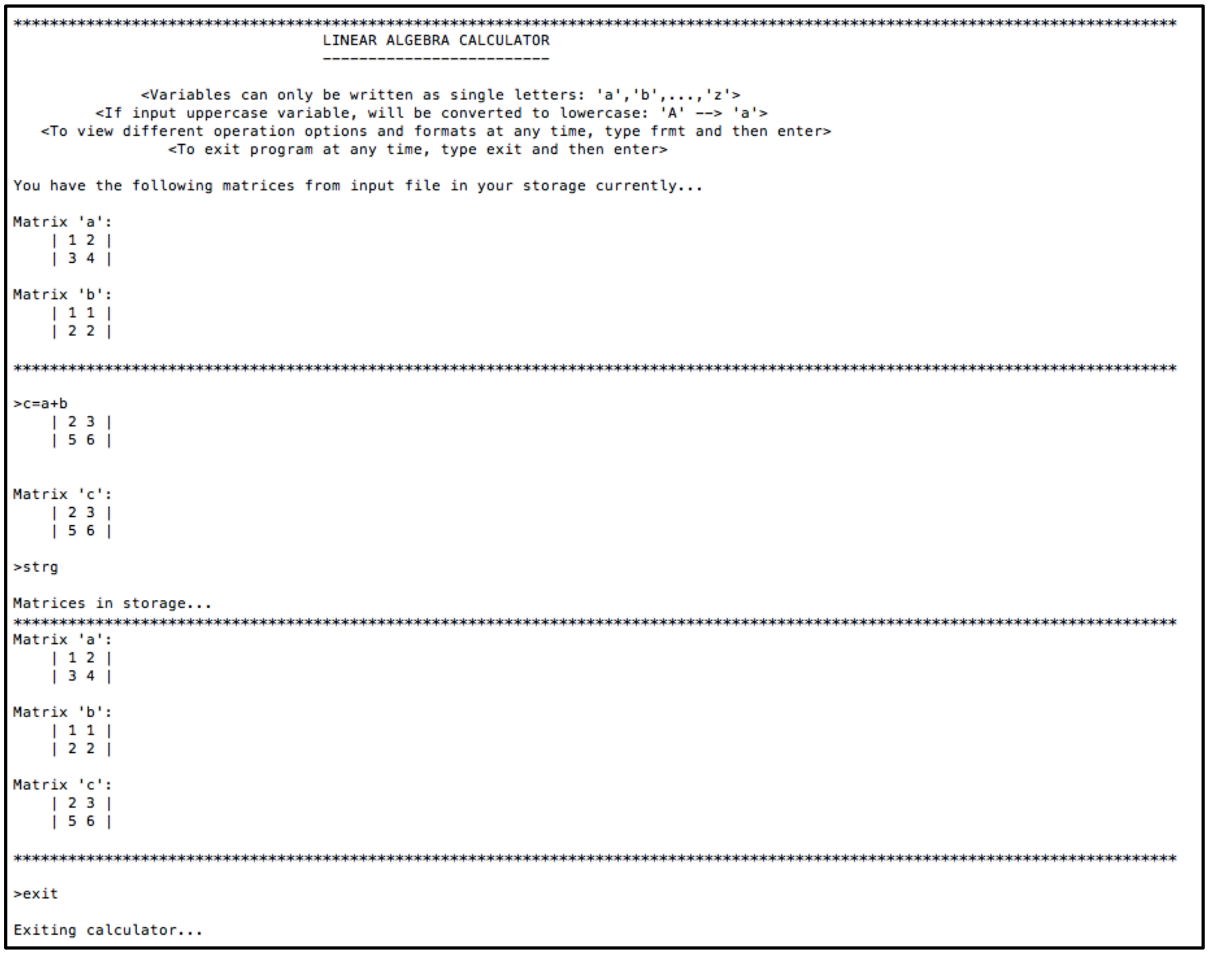
- Even after a matrix has been declared, for example variable 'a' = [9,2||], 'a' can take on new values. For example, if also had matrix 'b' in storage, where 'b' = [5||], user can write a = b, and both matrix 'a' and 'b' would equal [5||]. Or, user could write "a = [1,1,1||]", and 'a' would now equal [1,1,1||].
- It is important to note, any matrices in storage when calculator closes will be converted to their string representation and stored in the external storage, calculator_storage.txt. When the program is run again, the first thing the calculator will do it go to the external storage, read each line of the file, and convert the string representations and variables into Matrix objects, and store them in the internal storage. Example:

Original external storage (only matrix 'a' and 'b' stored):



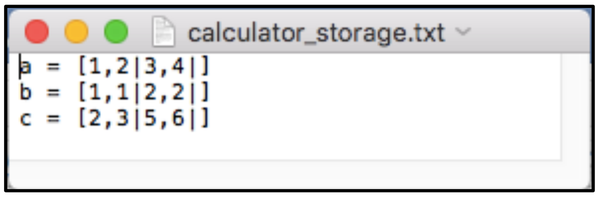
```
a = [1,2|3,4|]
b = [1,1|2,2|]
```

Operation "c=a+b", checking that 'c' in storage, and exiting of program



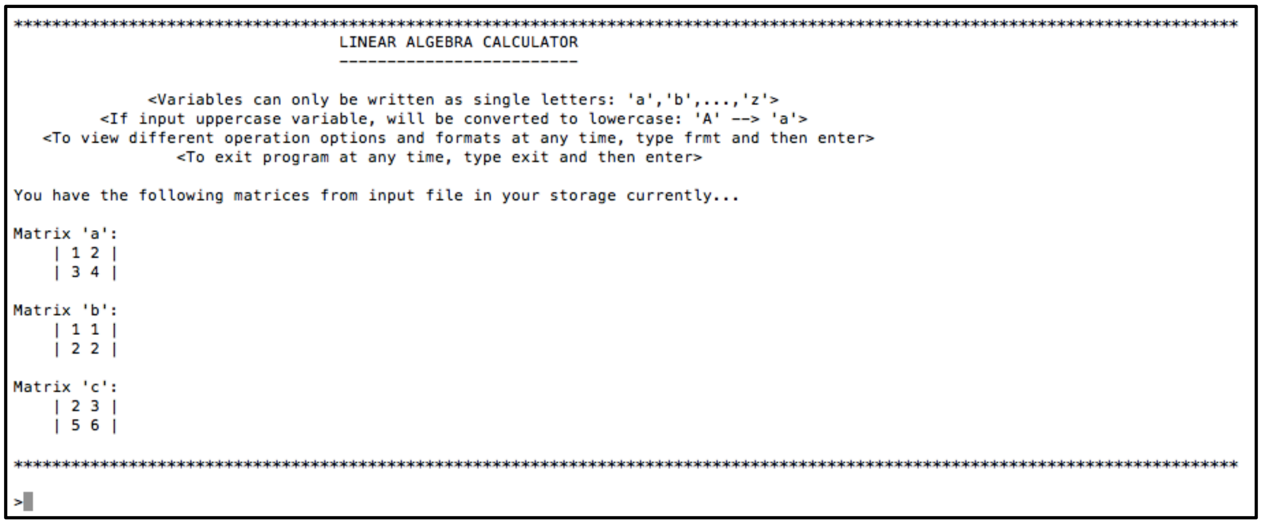
```
*****************************************************************************************************************************
                                 LINEAR ALGEBRA CALCULATOR
                                 -------------------------

              <Variables can only be written as single letters: 'a','b',...,'z'>
         <If input uppercase variable, will be converted to lowercase: 'A' --> 'a'>
   <To view different operation options and formats at any time, type frmt and then enter>
                 <To exit program at any time, type exit and then enter>

You have the following matrices from input file in your storage currently...

Matrix 'a':
    | 1 2 |
    | 3 4 |

Matrix 'b':
    | 1 1 |
    | 2 2 |

*****************************************************************************************************************************

>c=a+b
    | 2 3 |
    | 5 6 |


Matrix 'c':
    | 2 3 |
    | 5 6 |

>strg

Matrices in storage...
*****************************************************************************************************************************
Matrix 'a':
    | 1 2 |
    | 3 4 |

Matrix 'b':
    | 1 1 |
    | 2 2 |

Matrix 'c':
    | 2 3 |
    | 5 6 |

*****************************************************************************************************************************

>exit

Exiting calculator...
```

External storage after exiting program (now ‘a’, ‘b’, and ‘c’ written to file by program):


calculator_storage.txt

```
a = [1,2|3,4|]
b = [1,1|2,2|]
c = [2,3|5,6|]
```

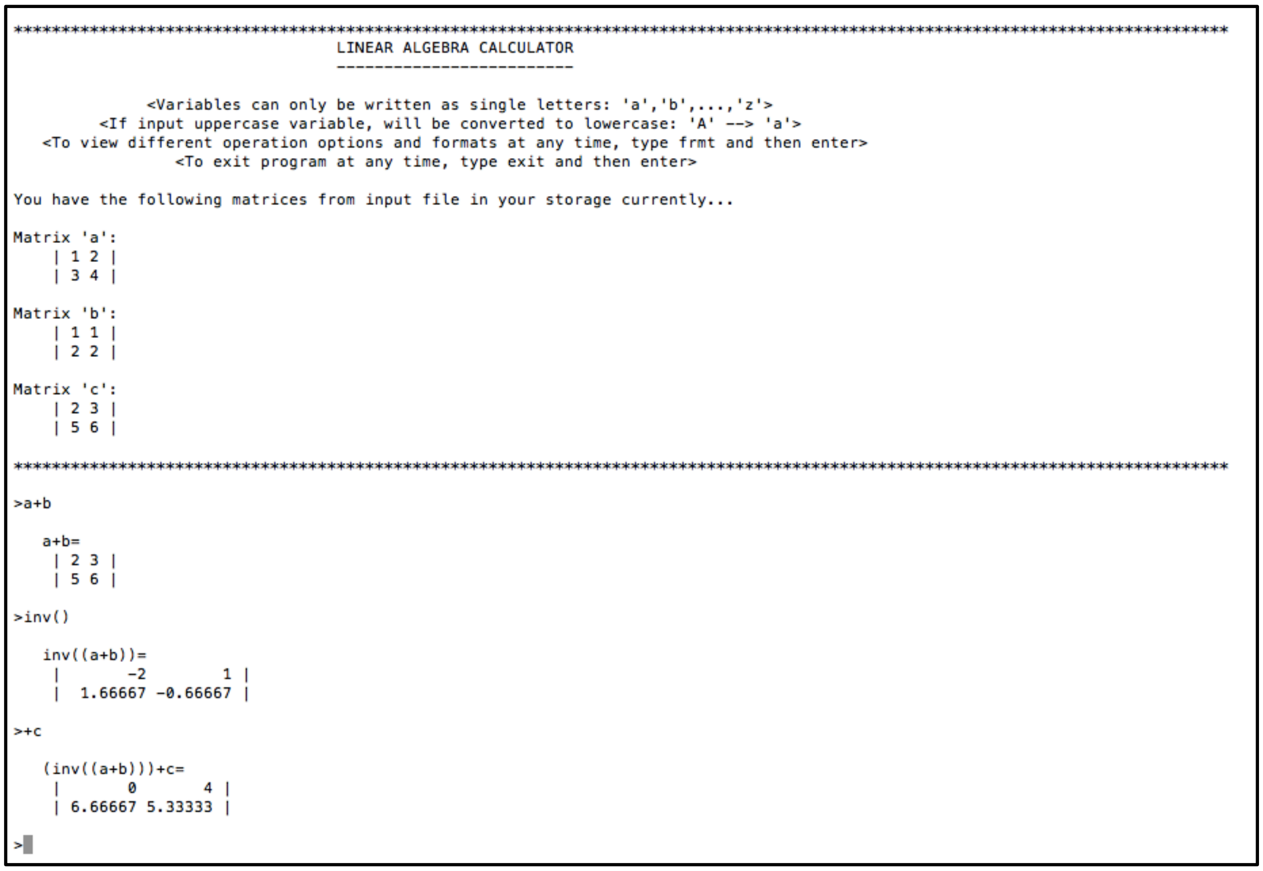
Next time running program (‘a’, ‘b’ and ‘c’ in storage ready for use):



```
************************************************************************************************************************************
                                  LINEAR ALGEBRA CALCULATOR
                                  -------------------------

              <Variables can only be written as single letters: 'a','b',...,'z'>
         <If input uppercase variable, will be converted to lowercase: 'A' --> 'a'>
   <To view different operation options and formats at any time, type frmt and then enter>
                 <To exit program at any time, type exit and then enter>

You have the following matrices from input file in your storage currently...

Matrix 'a':
    | 1 2 |
    | 3 4 |

Matrix 'b':
    | 1 1 |
    | 2 2 |

Matrix 'c':
    | 2 3 |
    | 5 6 |

************************************************************************************************************************************

>
```

Operation Chaining:

- While the calculator does not allow operations like 'ref(inv(a)) + b' to be performed, a "multi operation" as it contains multiply operations in one equation (first the program would have to find the inverse of 'a', then find the row echelon form of the inverse of 'a' and then add this to b), the program does support "operation chaining."
- Operation chaining is the process of continuing to perform operations on the previous output of the program. In the following example, the user begins with matrices 'a', 'b', and 'c'. The first operation the user performs is "a+b" which gives a matrix with values [2,3|5,6|]. The next operation performed by the user is "inv()". As there is no matrix variable inside the parenthesis of the operation, this means that the user is performing the "inv" (inverse) operation on the last output of the calculator, the last output being "a+b" or [2,3|5,6|]. The calculator finds that inv(a+b) = [-2,1|1.66667,-0.66667|]. Now the user inputs "+c". As there is no matrix variable before the plus sign, this tells the calculator to perform the operation on the last calculator output, being "inv(a+b)" or [-2,1|1.66667,-0.66667|]. As seen, the user can continue to perform operations on the previous output of the calculator.



```
*************************************************************************************************************************************
                                  LINEAR ALGEBRA CALCULATOR
                                  -------------------------

              <Variables can only be written as single letters: 'a','b',...,'z'>
         <If input uppercase variable, will be converted to lowercase: 'A' --> 'a'>
   <To view different operation options and formats at any time, type frmt and then enter>
                 <To exit program at any time, type exit and then enter>

You have the following matrices from input file in your storage currently...

Matrix 'a':
    | 1 2 |
    | 3 4 |

Matrix 'b':
    | 1 1 |
    | 2 2 |

Matrix 'c':
    | 2 3 |
    | 5 6 |

*************************************************************************************************************************************

>a+b

   a+b=
    | 2 3 |
    | 5 6 |

>inv()

   inv((a+b))=
    |       -2        1 |
    |  1.66667 -0.66667 |

>+c

   (inv((a+b)))+c=
    |       0       4 |
    | 6.66667 5.33333 |

>
```

- The different syntaxes for how different operations "operation chain" can be found through the format menu in the calculator itself, the format menu accessible by inputting "frmt" and enter.

Running Calculator:

- If the user DOES NOT move the calculator_storage.txt file from the LinearAlgebraCalculator folder, the program MUST be run from the LinearAlgebraCalculator folder so that it has proper access to the external storage file calculator_storage.txt. Otherwise an ERROR message “cannot find calculator_storage.txt” will be displayed and the program will exit immediately.
- If the user DOES move the calculator_storage.txt file, the program MUST be run from wherever the file is stored. Otherwise an ERROR message “cannot find calculator_storage.txt” will be displayed and the program will exit immediately.
- Calculator access point is in the Driver class, and program should be run from there.

Known Bugs:

- There are currently no known bugs.
- Currently in the middle of Beta Testing.

Future Features & Updates:

- Ability for user to input a “multi operation”, such as ’ref(inv(a)) + b’ where ‘a’ is the variable for one matrix and ‘b’ for another, and “ref” being the command for row echelon form.

Javadoc

- Javadoc can be found at LinearAlgebraCalculator/target/site/apidocs