Summers CAMP Manual

By Bryce Summers. Last Updated on December 12, 2015

Contents

[Sections: 1](#_Toc394753894)

[Overview 2](#_Toc394753895)

[User Interface 3](#_Toc394753896)

[Bryce TEX 4](#_Toc394753897)

[Bryce Numbers 6](#_Toc394753898)

[Opening and saving files 7](#_Toc394753899)

[Problem Solving Operations 8](#_Toc394753900)

[Problem Creation. 9](#_Toc394753901)

[Work Persistency 10](#_Toc394753902)

[Bryce Font 2 12](#_Toc394753903)

[Future Work 13](#_Toc394753904)

[Matrix Creation / Scalar Creation. 14](#_Toc394753905)

[Potential Uses for this Program 15](#_Toc394753906)

[Research Questions 16](#_Toc394753907)

[Current Dormant Features 17](#_Toc394753908)

[Web App 18](#_Toc394753909)

[FIXME : 20](#_Toc394753910)

# Overview

The Summer Computer Aided Math Program (CAMP) is a tool that is designed to help eliminate much of the tedious work that is involved in a traditional Linear Algebra class. In that regard it allows users to perform fundamental actions on data in parallel, such as adding entire vectors together instead of requiring the user to perform a separate addition for every set of corresponding entries within the vectors. It is the author’s intention that this program be used to divert the student’s focus from the mechanical arithmetic to the greater theoretical and practical ideas introduced in this course.

This program provides a seamless logical flow of operations between Matrices, vectors, and scalars.

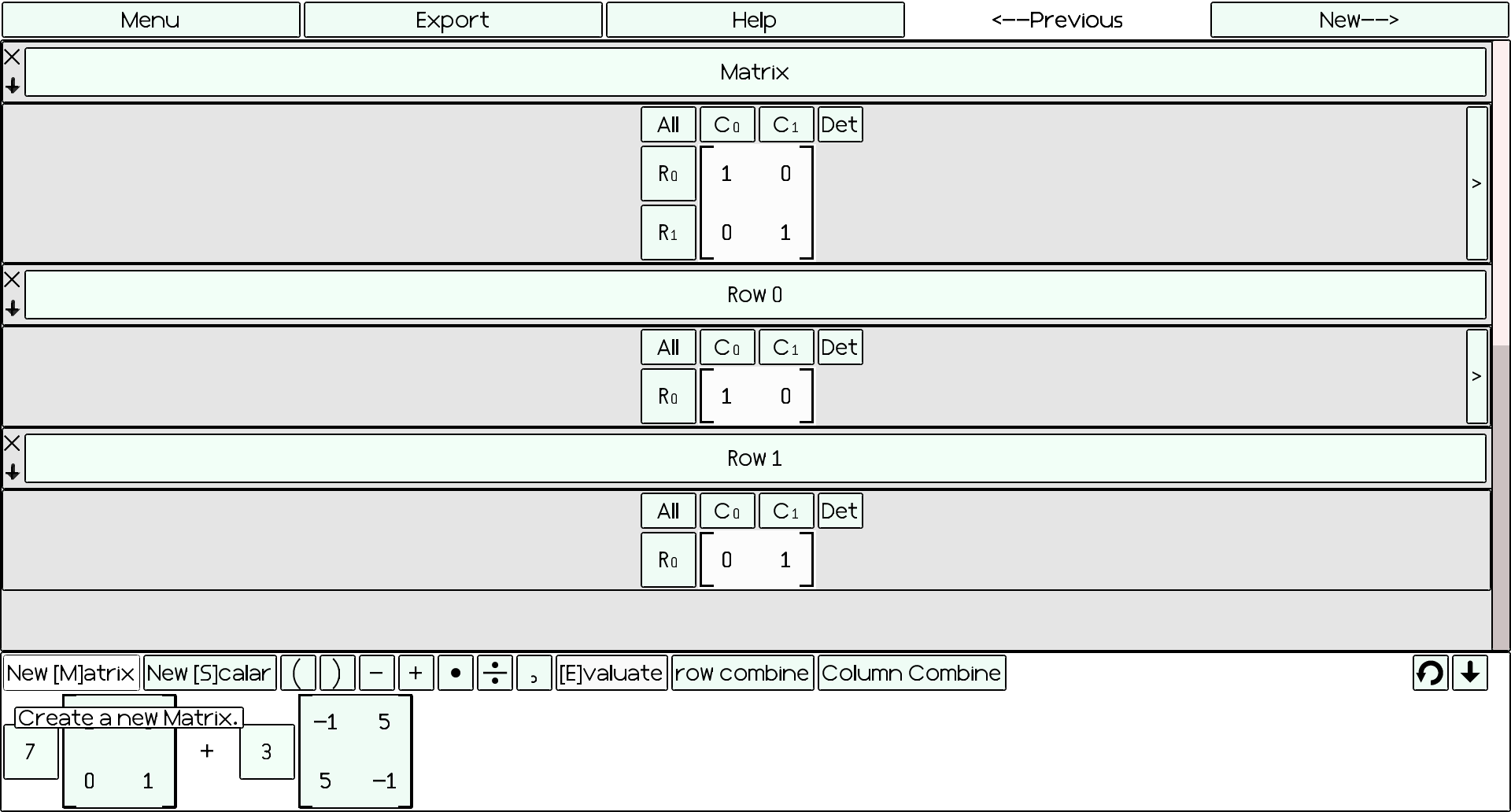
For video tutorials, please visit the following website: <https://www.youtube.com/channel/UCbDKxCh70C4wnAYVjRlw3UA/feed?view_as=public> this youtube channel can also be accessed by clicking on a help button while using the Summers CAMP program.

Please send comments and feedback to CAMP@funtheemental.com.

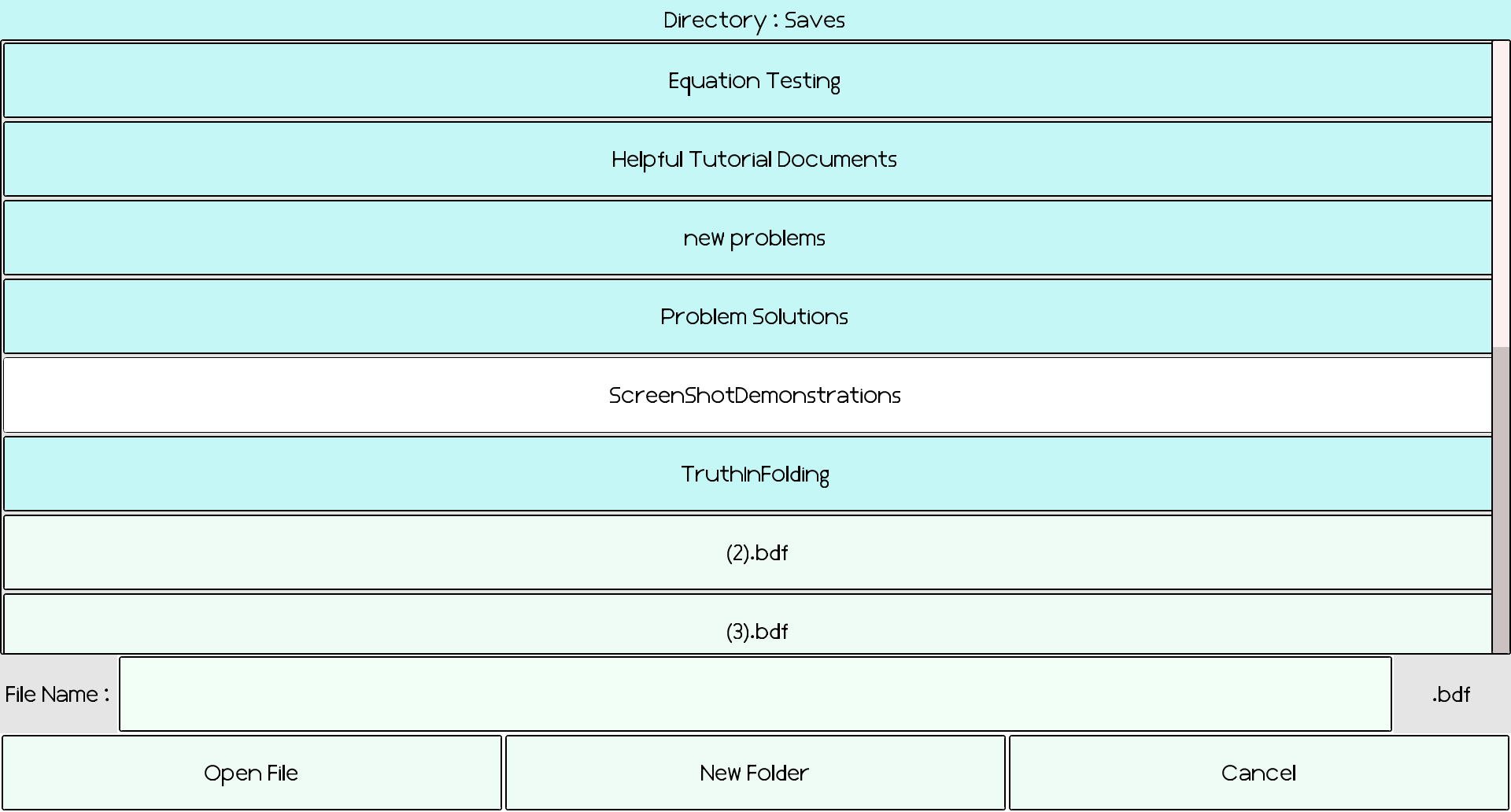
## Potential Uses for this Program

* Helping Students in Linear Algebra Class other technical courses.
* Helping TAs create new problems.
* Creating data sets of how students approach problems.
* General purpose Matrix/Vector/Scalar calculator.

# User Interface



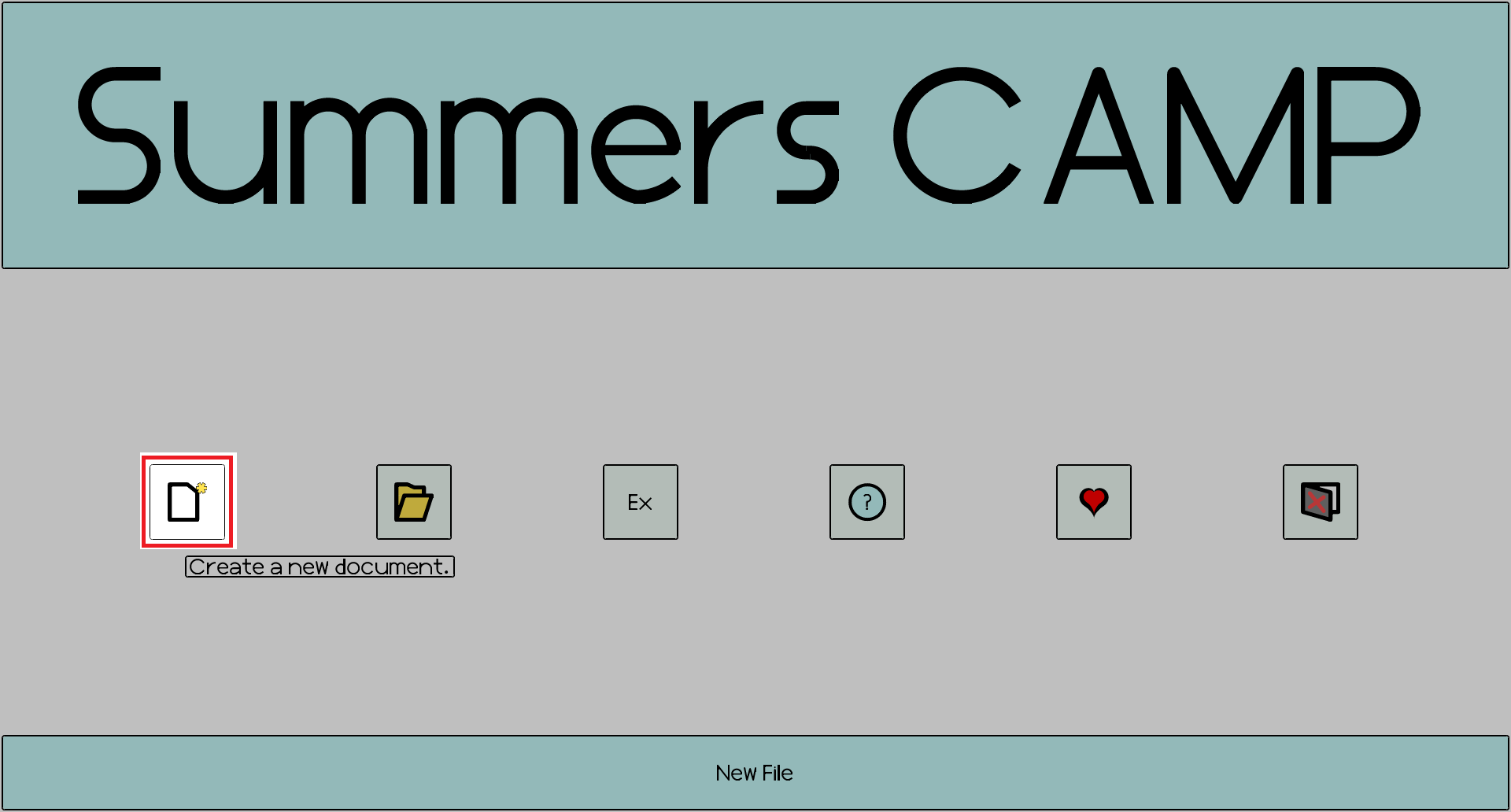
## Opening and saving files



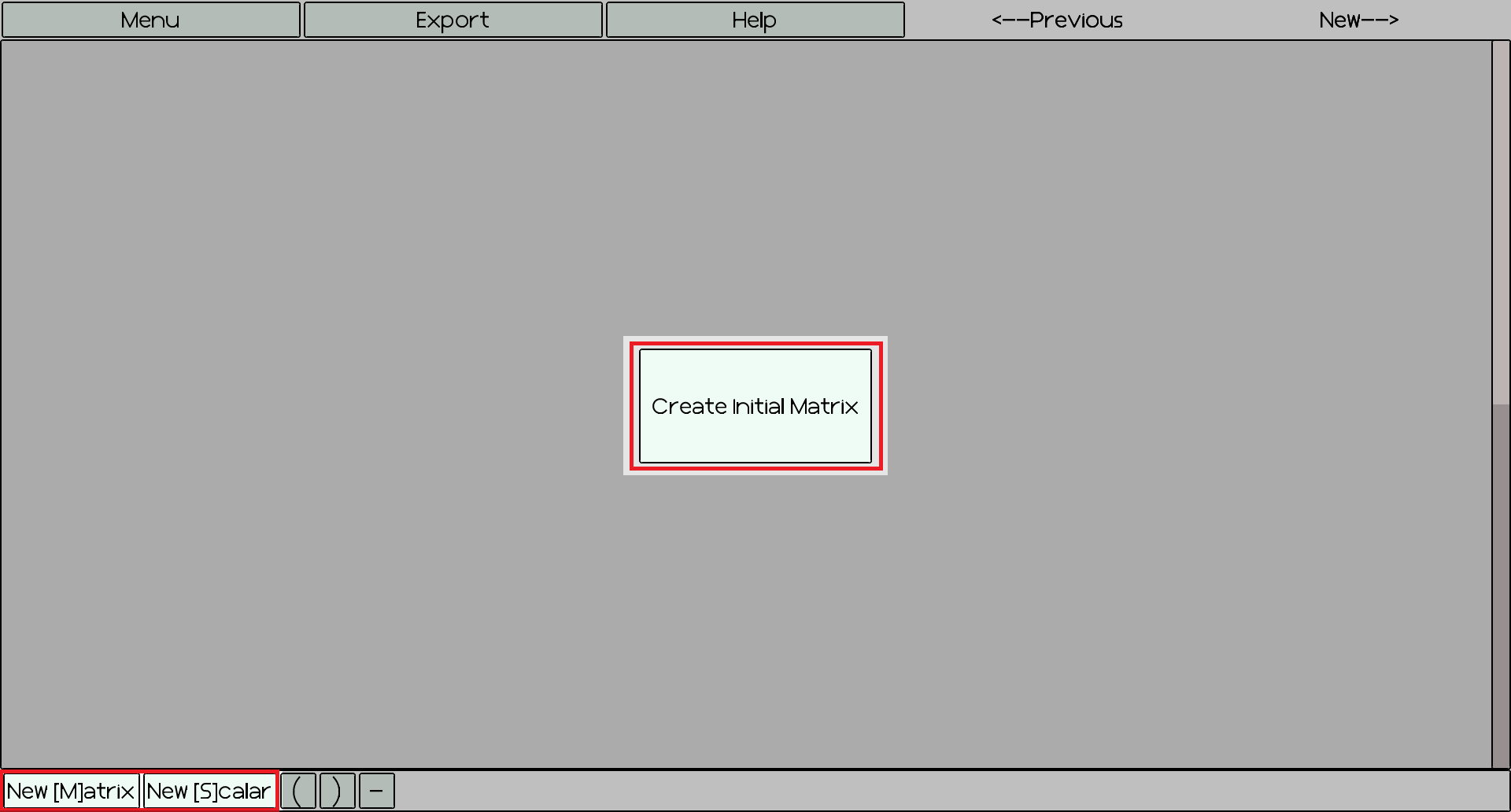
## Problem Creation.

Please follow These steps to create a Problem:

1. In the main menu, click on the “**New Problem”** Button.



1. Click on the **New Matrix** or **New Scalar** buttons to start creating the initial state for your problem. See the Matrix creation or Scalar Creation Sections for more information.



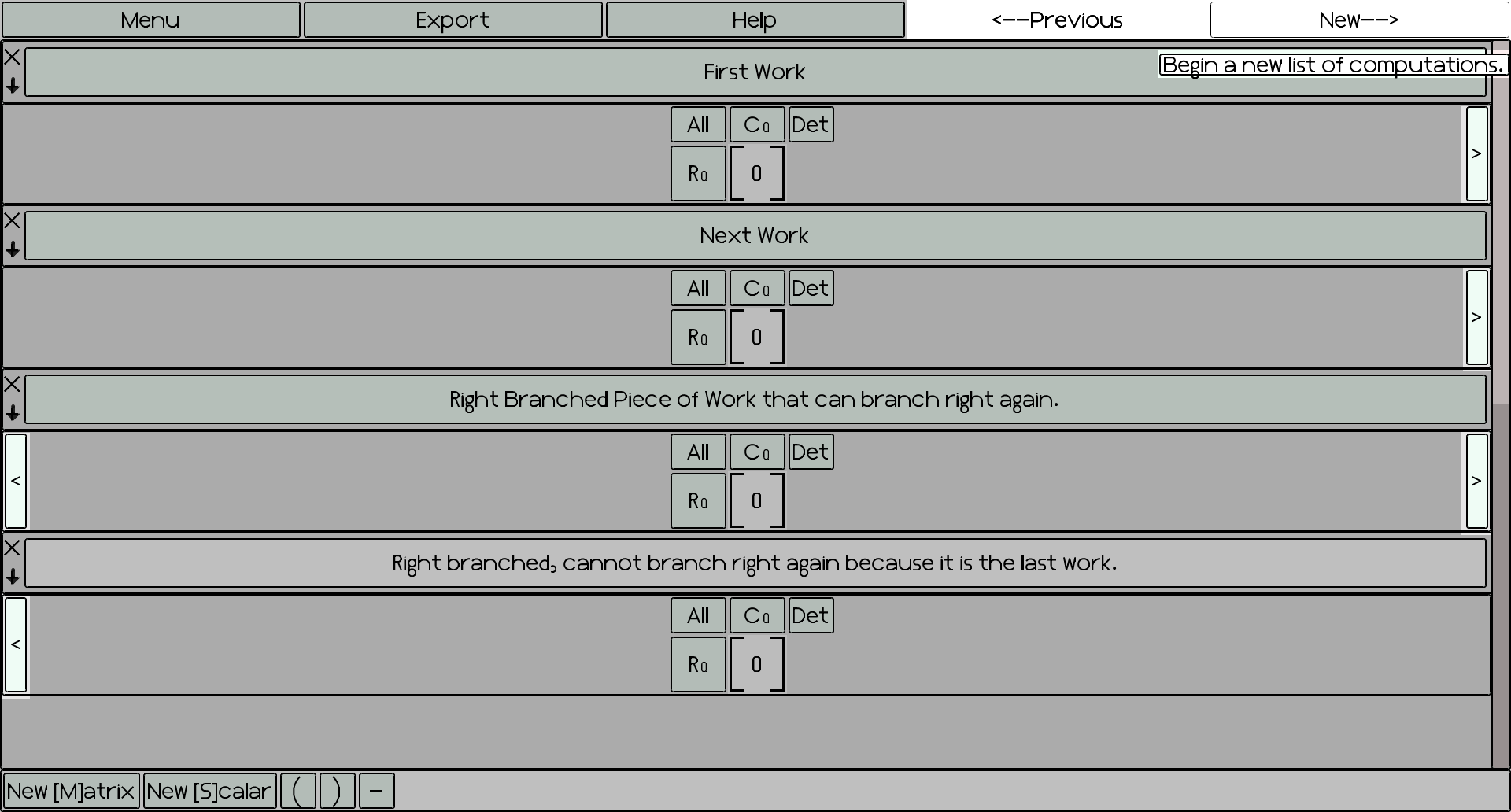
## Work Persistency

It is impossible to erase your mistakes with this program.

When opening a document, your steps are saved in a persistent tree structure. This allows you to:

* Try different solution attempts for problems by **navigating** the ‘Try’ buttons.
* Work on an **entire assignment** of problems within one document.
* All of your mathematical results are stored and can be used for further calculations. This encourages experimentation in the user’s problem solving.

In the User interface, the buttons that allow you to navigate the tree structure are as follows:

What each navigation button does:

|  |  |
| --- | --- |
| Button Name | Function |
| New Try | All of the steps that have been completed after the frame that contains this button will be erased and the user will have the opportunity to begin work on the problem as if the following work never existed. If a newer chain of reasoning exists, then it will reinstated. The Old try button can always be used to go back to the older chain. |
| Old Try | The previous line of reasoning that was erased following this frame is brought back. The user may now resume viewing or working on this chain. The new try button can always be used to go back to the newer chain. |
| New Problem | Moves the program to another root node of the tree, which was specified by the document’s creators. This functionality effectively moves the user to a newer problem. |
| Old Problem | Moves the program to another root node of the tree, which was specified by the document’s creators. This functionality effectively moves the user to an older problem. |

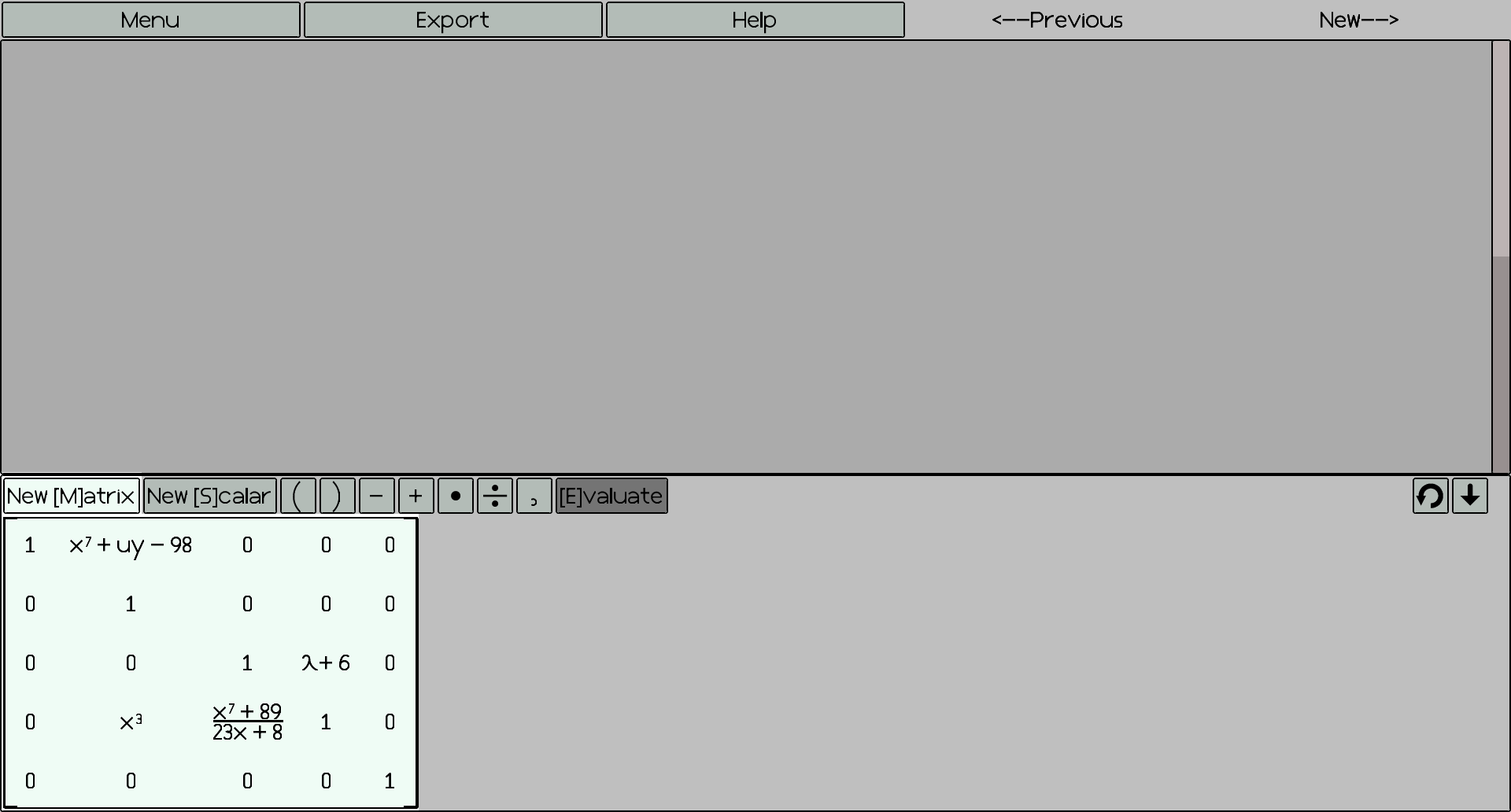
\*\*\*Show an example of a user using an asset from the first frame for a future frames, such as a cofactoring example.

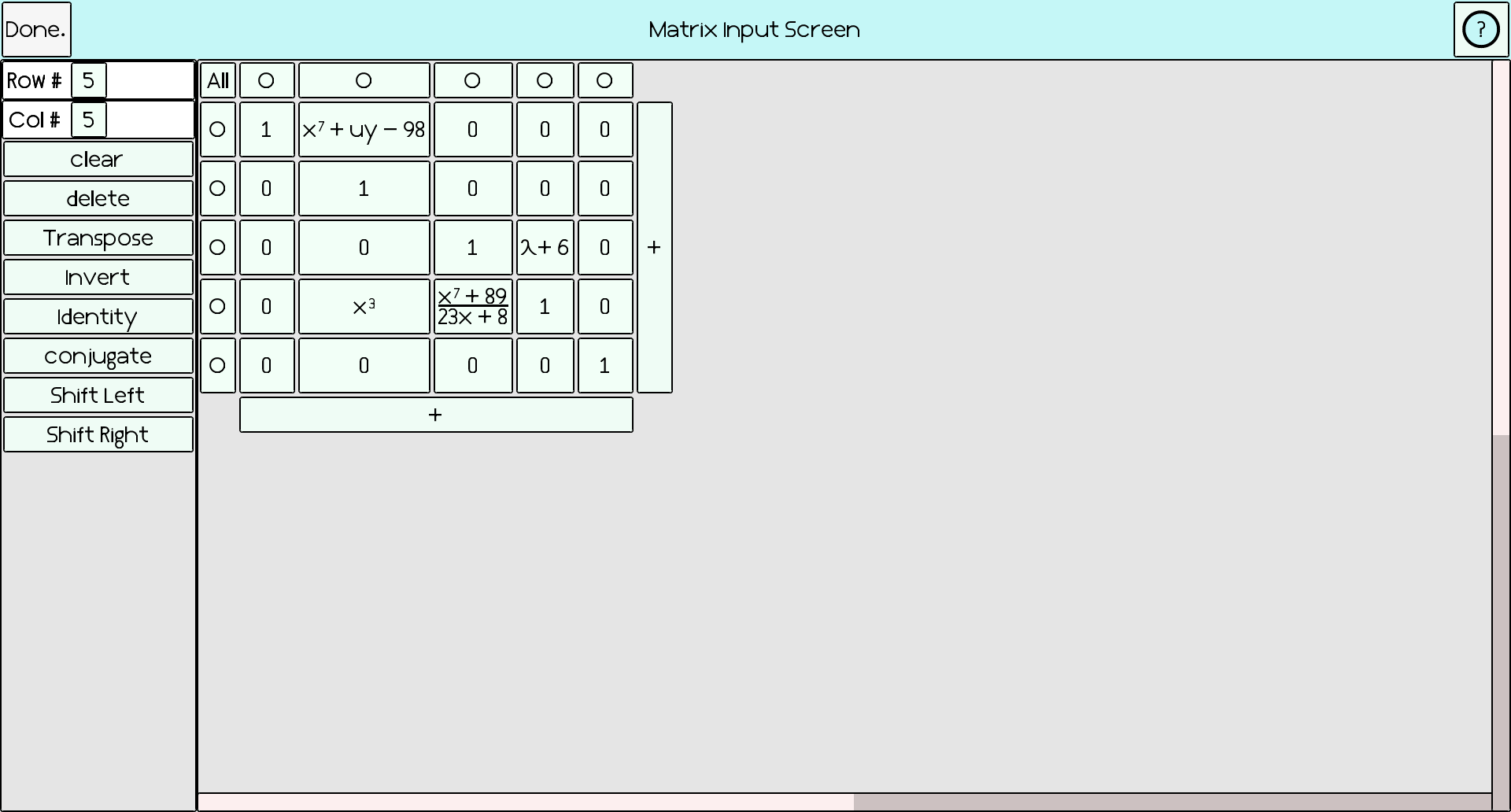
## Matrix Creation / Scalar Creation.

Scalars can be thought of as numbers, scalars, vectors, or matrices depending on the context.

Vectors can be thought of as vectors or Matrices depending on the context.

To manipulate such objects within the program, we must become acquainted with the **Matrix Input Screen.** This screen will pop up every time the user needs to input or change a Matrix. To navigate to it you can do any of the following :





# Systems

In this section, this manual endeavors to describe the different Capabilities of the Summers CAMP.

## Bryce TEX

Donald Knuth invented TEX, which is a computer language for typesetting documents, back in the 1980s, when he was frustrated at the lack of typesetting support for his opus : *The Art of Computer Programming*. The TEX program has been expanded to modern incarnations such as LaTeX, ConTeXt. These fully functional modern packages usually require quite a bit of memory space on your computer, so the Summers CAMP comes with built in support for a very narrow subset of the typesetting commands commonly available with the intent of improving the aesthetic look of the mathematics the user will peruse on their computer screens.

To use this built in functionality, you may type text into in program text boxes and the text will be formatted according to the following visual transformations after you have left the box:

|  |  |  |
| --- | --- | --- |
| Name | Syntax | Result |
| Superscript | x^{7} |  |
| Subscript | x\_{2} |  |
| Fractions | \frac{x}{y} |  |
| Dot | \cdot |  |
| Greek Letter | \l  \a  \b  \g  \d  \e  \t | α  β  γ  δ  ε  θ |
| Big Parens | \( \{ \[ | {[( |
|  |  |  |
|  |  |  |

## Bryce Numbers

The Summers CAMP program is equipped with a built in Computer Algebra System (CAS) called Bryce Numbers. Behind the scenes this CAS is performing processing on the matrices and scalars using internal Bryce numbers, whenever the user performs an operation.

Table 1 communicates the various numbers the user can use within the system. Please note that each number type is a subset of all of the numbers that come after it.

|  |  |  |
| --- | --- | --- |
| Number Type | Usage | Examples |
| Natural Number | Used to represent positive constant expressions. | 1, 2, 3, 4, … |
| Integer | Used to represent all constant numbers with no fractional component. All Natural numbers are Integers. | -10, -32678, 0, 1,  6, 4534 |
| Rational | Used to represent all numbers , where A and B are integers and B != 0. All integers are Rationals. | ,, , |
| Complex | Used to represent all numbers of the form A + Bi, where A and B are rationals and *i* is the imaginary constant equal to the square root of -1. All rationals are Complexes | 1, -*i,* 6 + *2i,* |
| Monomial | Used to represent a set of algebraic symbols and integer exponents along with a complex scalar value. All complexes are monomials with an empty set of algebraic terms. |  |
| Multinomial | A set of monomials with non-equal algebraic symbol sets that is interpreted as the addition of all of the terms. All monomials are multinomials. |  |
| Rational Multinomial | Represents all numbers , where A and B are Multinomials and B != 0.  All multinomials are rational multinomials with a denominator of 1. |  |
| Expression  (Not yet Implemented) | Exponentiation of algebraic and complex terms by any expression will be possible. All Rational multinomials | , |
| Equation | Represents any number of the form A = B, Equations can either be true or false, but represent a relationship between two numbers. Algebra between Equations is defined as follows.  (A=B)○C 🡪 A○C = B○C,  (A=B)○(C=D) 🡪 A○C=B○D  All expressions are equations, where B is not defined. | , |
| Vector | A vector is a one dimensional collection of numbers. Operations on vectors of the same size are distributed between the cooresponding indexed numbers. | , |
| Matrix | A Matrix is a collection of vectors that forms a 2 dimensional array of numbers |  |

Table 1

Table 2 communicates the operations that can be performed with numbers. Please note that the operations are written in roughly reverse order of operations order. Exponentiation by any number is not yet supported.

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Input | Output | Example |
| Creation | *literal* | Creates a Bryce Number from a string literal by reducing | (X + 1.1 + Y)/5 = |
| Addition (+) | A, B | A + B | 1 + 2 = 3  (X + 1) + (-Y) = X – Y + 1 |
| Subtraction (-) | A, B | A – B | 1 – 2 = -1, () – () = () |
| Multiplication(\*) | A, B | A \* B | 2\*3 = 6, |
| Division |  |  | 5 / 6 = |
| Integral Exponentiation(^) | A, I | A^I (I needs to be an integer.) | 2^3 = 8, X^3 = X3,  (X + 1)^2 = |
| Complex Conjugation | A | All of the imaginary components of A will be negated. | Conj(5 + 7i) = 5 – 7i |

Table 2

## Simplification

The Summers CAMP program simplifies every number inputted by the user into a canonical Bryce Number form. For example, a decimal entry such as .1 will be changed into the fraction . This process aids in the internal processing of numbers and is also meant to serve as a convenience to the user. No simplification method is perfect for all users, but the methods chosen for this program are meant to be useful to linear algebra students.

## Problem Solving Operations

*This section has not yet been written.*

## Row reducing.

https://www.youtube.com/watch?v=IEunQOvLRZ0

## Column reducing.

## Cofactor expansions.

https://www.youtube.com/watch?v=hO1l4HZ20cw

## Matrix / Scalar / Vector Arithmetic.

## Bryce Font 2

The Summers CAMP program displays text to the screen using glyphs (letters) that are rendered internally in the program. This functionality is designed to standardize the font viewing experience across all computers and eliminates the programs’ dependency on user installed fonts. This also allows the fonts to be displayed correctly with proper antialiasing (Non-jagged edges) at desired sizes. In hindsite, it may be easier and possible to obtain the desired technical properties regarding text display using more standardized methods and fonts. In particular java and most other programming languages do indeed support antialiasing.

# Web App

Thus far the development of a Web version of this program has stalled due to the current state of fear regarding java applets. It is terribly cumbersome to get the applets signed and to get past all of the saving and cryptic problems. It may be best to implement this program in a web friendly language in the future if that is desired.

In the future, the Summers CAMP program may be ported to Javascript.

# Conclusive Information

This section endeavors to provide the reader with a glimpse of possible futures and uses for the Summers CAMP program.

## Current Dormant Features

There are several features within the Summers CAMP code base that currently lie dormant. They may be removed in the future, but they are currently still in the code although not shown to the user.

Look for the @Dormant annotations in the code.

1. Determinant mode : scalars and matrix bar modifications are used to provide the user with a scaling factor for use in Gaussian elimination. I have removed it, because it is confusing to others and is only useful for one approach to determinant calculation algorithms. The user can easily manually keep track of the data through extra frames, although it will be less streamlined than with the feature.
2. Visual operation frames. These frames allow the user to visually see the previous steps that have been performed. They require a lot of edge casing to display properly, increase the size of the files, and I also bet that they confuse the user. The user would rather see their results. The text boxes should be used instead for the user to communicate with themselves regarding the operations that they have done.

## Research Questions

1. Can custom made programs help in the learning process. There are two different
2. How can programs be used to gain deeper data regarding the process of Learning. Can we observe the creative process behind homework instead of merely viewing the end result?
3. How can we use computers to test and evaluate students at the level of abstraction they are being taught at? In the case of linear algebra, we want them to think at the level of row operations instead of arithmetic.
4. Are such programs useful for the creation of new problems for students? While students take a problem description and find an answer, teachers want to start with nice answers and reverse engineer them into problems. Because of this teachers will also benefit from using the Summers CAMP program.

## Future Work

In order for this program to be used to diagonalize Matrices and find the eigenvalues of even simple 2 by 2 matrices, more powerful expressions will need to be implemented, because exponentiations and squareroot signs and functionality are currently not supported. An article that seems to hold the secret to such an extension can be found here :

<http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=CDE49A72547E32A376D34064D072F5CD?doi=10.1.1.137.7024&rep=rep1&type=pdf>

If exponentiation by any expression were to be implemented, it would also give the number system the capability to handle algebra on trigonometric functions. (The reduction is via Euler’s formula.)

# Troubleshooting

If double clicking on the jar brings up a blank program, try going to this web page to fix your computer’s jar association details: <http://johann.loefflmann.net/en/software/jarfix/index.html>.

If parts of the screen seems to be filled with grey rectangles that look like they do not belong, try moving your mouse over the regions where buttons should be drawn.

Make sure the jar file that you are attempting to run is not inside of a zip folder. If this is the case, then you will have to *extract* it from the relevant zip folder before it will run properly.

You must have java installed on your computer.