Copy Rights:

This is the program **FINDNULL** created by **David Zhang** written for TI-83 or compatible calculators which finds the null space, or kernel, of a matrix.

The reason why I created this program is because I couldn't find a program like this over the internet for my TI-83. Having no choice, I was forced to write one.

I don't want others who are also looking for a program like this going through the same trouble as I have, therefore, you can distribute, modify or do whatever you want with this program (the level of programming is pretty basic because I don't know how to program in assembly language for TI-83 yet, so if you can, GREAT!)

All I'm asking for is, if you distribute or modify this program, please include this README file with the program; or you can give a portion of the credit to me as the original creator or the 'forefather' of the program.

If you have any comments or bugs to report email me at dazhang@ucsd.edu

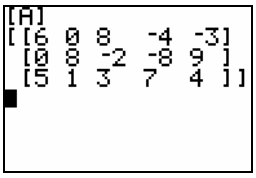
Accuracy of the Program:

The level of accuracy of this program is as accurate as your calculator's build-in rref ( ) (reduced row echelon form) function of you calculator.

How to Use the Program:

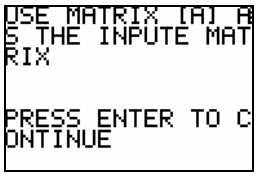
Enter the matrix you want to find the null space (kernel) into matrix A (in TI-83, that would be [A], consult your TI-83 manual if you have no idea what I'm talking about)

Example:

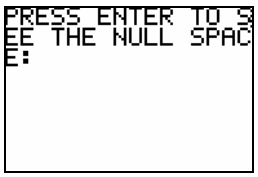


Then run the NULLFIND program:

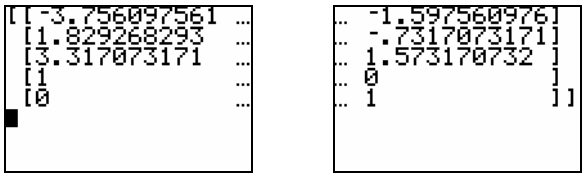
The program would tell you to enter the matrix into matrix [A] (if you're reading this, I know this is unnecessary for you, but this is done for those idiots who don't read the REAME file first)



After you press “Enter”, the program would ask you to press “Enter” again (I know, I know, but this is done to let the user know that the null space (kernel) is coming up).



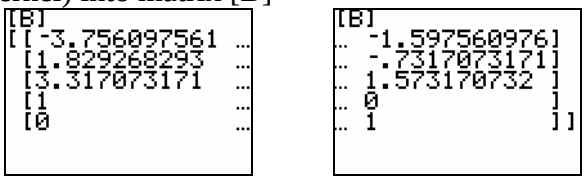
Finally, after all those finger-tiring button-pushings, voila! You have you null space (kernel)*



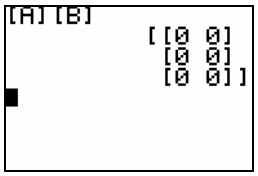
*For you TI-83 users, you know how to scroll left and right, right?

“How do you know this is right,” you say? Well, if this is indeed the null space (kernel), then if you multiply matrix [A] with each of the column vector of the null space (kernel), it should give you all zeros.

For those of you who didn’t write down the null space (kernel), fortunately, this program store the null space (kernel) into matrix [B]



Let’s see if this is true:



Sure enough, it’s right. This concludes the tutorial. Hope you enjoyed it despite my some-what annoying comments