HOMEWORK #1:

First Job

Due Date: Friday, February the 7th, 11:59pm

For this assignment, you will submit a single C++ file called '`gmatrix.cpp`'.
Remember to put your name and section at the top of your program file.

Problem

Good News!! You have been hired by "Planet Express Softworks", a new division of "Planet Express Inc." and newest venture of Prof. Farnsworth. The business model is simple: you write the software, and the delivery crew delivers it. It is not like Prof. Farnsworth is greatly interested in software. He is just interested in using the profits to fund his research.

You have joined just in time. "Planet Express Softworks" just found their first ~~victim~~ client.

The Cerulian Curators are a group of art enthusiasts from the planet Caelum-9, and they are of late interested in digital art. In particular, they are interested in something they call "Gradient Matrices". A Gradient Matrix is just a rectangular matrix of integers, but filled with values following a particular pattern. Every Gradient Matrix has a '*seed*' and a '*step*'. The top-left corner of the matrix (the 1st diagonal) has the value of the seed. The cells below and to the left of the top-left corner (the 2nd diagonal) have the value of the '*seed* + *step*'. The 3rd diagonal has the value of the 2nd diagonal incremented by '*step*'. The next diagonal has the value of the previous diagonal incremented by '*step*', and so on.
For example, a 6 by 4 gradient matrix with seed 5 and step 2 looks like:

5	7	9	11	13	15
7	9	11	13	15	17
9	11	13	15	17	19
11	13	15	17	19	21

It is hard to say why they find them beautiful, but they want you to create a C++ class to generate them. And, thankfully, they have even **supplied a header file** to guide your implementation!.

Implementation Requirements

You should implement **ALL** the member functions specified in the '`gmatrix.h`' header file, as the clients will test your implementation before payment.

Given that you do not know beforehand how large gradient matrix could be, your class should dynamically allocate a 2D Array. Make sure to de-allocate the 2D Array in the destructor.

Write a program to test your implementation thoroughly before submitting!!