CSCA48, Assignment #1-1
Design of Matrix Class Hierarchy
due 11:55pm June 4, 2017

Introduction

In this assignment you will design and then implement a `Matrix` ADT. In this first phase, you must design the ADT itself, including a class hierarchy and all methods and external documentation. In the second phase, you will then implement the functionality, but in the interest of recreating a realistic scenario, you won't know about the implementation constraints while you're designing the abstract data type.

In order to further recreate a realistic scenario, you won't be receiving a nice orderly well defined set of instructions that are carefully crafted to point you in the right direction (for this phase at least). Instead, you'll get the notes from the client meeting where the requirements were discussed, and part of your job is to decide what is and is not meaningful/important.

The "Instructions"

Below is the transcript from the meeting with the client.

> I need to be able to create matrices. Not like the movie, but like the mathematical things. Usually they'll be holding numbers, but down the road we may want to hold letters and stuff. When they're holding numbers, normal matrix mathematics should apply. I need to be able to multiply, add and subtract. When the matrix doesn't have numbers, I guess that multiply and subtract doesn't make sense, but I'd like add to do the obvious thing, like that programming language you showed me where when you add words together they combine into a longer word.
>
> For any matrix I should be able to get or set a value of a column or a row. For square matrices, I'd like to be able to get or set the diagonal in the same

manner. I work with 1 dimensional matrices a lot, so those should be easy to deal with (I don't want to have to always ask for element (1, 7), I should just be able to ask for element 7.

I should be easily able to swap around any rows/columns.

I need to be able to take the determinant of a matrix, but only if it's 2x2. I should be able to transpose any matrix.

Once I've decided on the dimensions of a matrix, they never change. I never need to add a row or a column, or delete one.

I tend to work with either relatively small matrices (2-3 rows/columns) or really big ones (100s of rows/columns).

I tend to work with identity matrices a lot, so I want an easy way to just create them (I'd better not have to type loads of 7s if I want to create a 100x100 identity matrix with all values of 7 on the diagonal... that's what an identity matrix is, right?).

Some of my work also deals with symmetric matrices, but sometimes I change one value and forget to change its mirror, so I'd like some way of making that process easier.

Your Task

Your first task is to read the transcript very carefully and try to understand everything that the client wants. And from that understanding to design an ADT for the `Matrix` class.

We highly recommend that you start by drawing the UML diagram. You'll find it much easier to plan and arrange your classes and methods this way. You don't need full UML, just the level which we have been using in lecture. Think carefully about inheritance and abstraction, including exceptions and over-writing methods.

Once you've decided on the basic structure, you will write the ADT, complete with external documentation (i.e., DocStrings). You do not need to write the bodies of any of the methods, you can leave them blank, or write `pass`.

Finally, you will design and implement a full test-suite in the form of a series of UnitTests. These should test all methods as fully as possible without knowing any details of the eventual implementation (this is called `black box testing`).

After the submission deadline, the second phase of the assignment will be released where you will actually implement a Matrix data type. In this phase you will be using a standard ADT provided by the course instructors (so don't get too attached to your design decisions; but the good news is that any bad decisions won't haunt you into the next phase).

What to Submit

You will submit three files to MarkUs, `a1_design.py`, which will contain all of the code for your `Matrix` related classes, complete with all external documentation (but no method bodies yet), `a1_testing.py`, which will contain your UnitTests, and `a1_uml.xxx` where `xxx` can be one of `jpg`, `pdf` or `png`, which will contain your UML diagram[1]. The diagram can be hand drawn and scanned/photographed, but should be clearly legible, and all files have a maximum size of 1MB.

Hints & Notes

The first phase of this assignment is really all about being able to read and parse instructions that aren't laid out for you as clearly as what you're used to receiving in `cscA08`. There is a lot of information here, and you will need to read this handout carefully, most likely several times in order to make sure that you've got everything.

Remember that this is an **individual** assignment. **DO NOT** discuss your approach or design with your fellow classmates. You may ask for clarification on Piazza, but please do not ask or post anything that gives away any part of your design.

[1] Notice that since we don't have a specific file type, we won't be able to specify a file name, so MarkUs will not warn you if you forget/misname this file