mat

Coding the Matrix, Summer 2013

Please fill out the stencil file named "mat.py". While we encourage you to complete the Ungraded Problems, they do not require any entry into your stencil file.

Problem 1: You will write a module `mat` implementing a matrix class `Mat`. The data structure used for instances of `Mat` resembles that used for instances of `Vec`. The only difference is that the domain `D` will now store a pair (i.e., a 2-tuple) of sets instead of a single set. The keys of the dictionary `f` are pairs of elements of the Cartesian product of the two sets in `D`.

Like in `Vec`, we have provided a file `test_mat.py` containing examples of how the `Mat` class is used. You can copy and paste these examples into your REPL, or run `python3 -m doctest test_mat.py` to run all of the examples.

The operations defined for `Mat` include entry setters and getters, an equality test, addition and subtraction and negative, multiplication by a scalar, transpose, vector-matrix and matrix-vector multiplication, and matrix-matrix multiplication. Like `Vec`, the class `Mat` is defined to enable use of operators such as + and *. The syntax for using instances of `Mat` is as follows, where `A` and `B` are matrices, `v` is a vector, `alpha` is a scalar, `r` is a row label, and `c` is a column label:

operation	syntax
Matrix addition and subtraction	`A+B` and `A-B`
Matrix negative	`-A`
Scalar-matrix multiplication	`alpha*A`
Matrix equality test	`A == B`
Matrix transpose	`A.transpose()`
Getting and setting a matrix entry	`A[r,c]` and `A[r,c] = alpha`
Matrix-vector and vector-matrix multiplication	`v*A` and `A*v`
Matrix-matrix multiplication	`A*B`

You are required to write the procedures `getitem`, `setitem`, `mat_add`, `mat_scalar_mul`, `equal`, `transpose`, `vector_matrix_mul`, `matrix_vector_mul`, and `matrix_matrix_mul`.

Download the file `mat.py` to your working directory, and, for each procedure, replace the `pass` statement with a working version. Test your implementation using `doctest` as you did with `vec.py`. Make sure your implementation works with matrices whose row-label sets differ from their column-label sets.

Note: Use the sparse matrix-vector multiplication algorithm described in lecture (the one based on the "ordinary" definition") for matrix-vector multiplication. Use the analogous algorithm for vector-matrix multiplication. Do not use `transpose` in your multiplication algorithms.