Last name ________________________

First name ________________________

LARSON—MATH 310–CLASSROOM WORKSHEET 16
The Mat Class.

1. Set up your CoCalc JUPYTER notebook for today's work.
   (a) Start the Chrome browser.
   (b) Go to `https://cocalc.com`
   (c) Log in.
   (d) You should see an existing Project for our class. Click on that.
   (e) Make sure you are in your Home directory (if you work in your Handouts directory, your work could get overwritten).
   (f) Click "New", then "Jupyter Notebook", then call it **310-c16**.
   (g) Make sure you have PYTHON as the *kernel*.

From: Chapter 2 of Klein's *Coding the Matrix* text

2. Go to your Handouts folder and copy the file "Vec_c15.py" to your Home directory. If you were in class it will already be there (it hasn't changed). That has all our work from the previous class, plus a working version of the `__repr__` method, which prints representations of objects in a Class, and a working `print` method. The Mat class calls it, so we need those definitions.

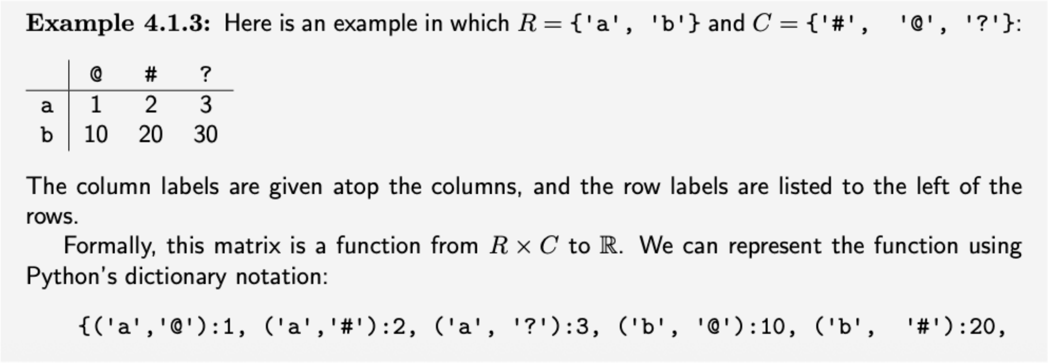
3. Run/evaluate to import everything from that file to memory.

```
from Vec_15 import *
```

From: Chapter 4 of Klein's *Coding the Matrix* text

4. Recall that we can view a matrix as a collection of (row number, column number) pairs, each associated to a real number (or any field element). So in that sense we could code a "matrix" (or think about a matrix) very similarly to how we thought about and represented vectors:

```
class Mat:
    def __init__(self, labels, function):
        self.D = labels
        self.f = function
```



Example 4.1.3: Here is an example in which $R = \{$'a', 'b'$\}$ and $C = \{$'#', '@', '?'$\}$:

	@	#	?
a	1	2	3
b	10	20	30

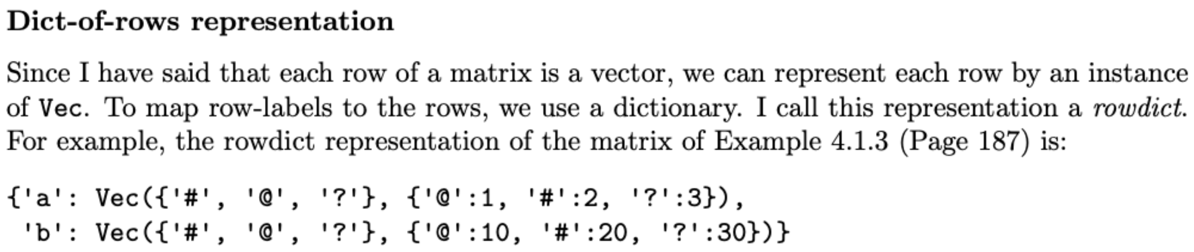
The column labels are given atop the columns, and the row labels are listed to the left of the rows.

Formally, this matrix is a function from $R \times C$ to $\mathbb{R}$. We can represent the function using Python's dictionary notation:

```
{('a','@'):1, ('a','#'):2, ('a', '?'):3, ('b', '@'):10, ('b', '#'):20,
```

5. Copy the Mat stub-definition above from our last class worksheet (pushed to Handouts if you weren't in class), then code, evaluate/run, and test:

```
M=Mat(({'a','b'}, {'@', '#', '?'}), {('a','@'):1, ('a','#'):2, ('a','?')
    :3, ('b','@'):10, ('b','#'):20, ('b','?'):30})
```

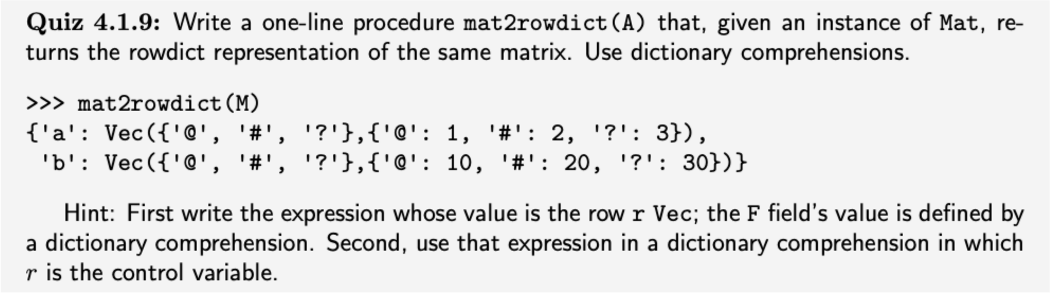


Dict-of-rows representation

Since I have said that each row of a matrix is a vector, we can represent each row by an instance of Vec. To map row-labels to the rows, we use a dictionary. I call this representation a *rowdict*. For example, the rowdict representation of the matrix of Example 4.1.3 (Page 187) is:

```
{'a': Vec({'#', '@', '?'}, {'@':1, '#':2, '?':3}),
 'b': Vec({'#', '@', '?'}, {'@':10, '#':20, '?':30})}
```

6. Code, evaluate/run, and test:



Quiz 4.1.9: Write a one-line procedure mat2rowdict(A) that, given an instance of Mat, returns the rowdict representation of the same matrix. Use dictionary comprehensions.

```
>>> mat2rowdict(M)
{'a': Vec({'@', '#', '?'},{'@': 1, '#': 2, '?': 3}),
 'b': Vec({'@', '#', '?'},{'@': 10, '#': 20, '?': 30})}
```

Hint: First write the expression whose value is the row r Vec; the F field's value is defined by a dictionary comprehension. Second, use that expression in a dictionary comprehension in which r is the control variable.

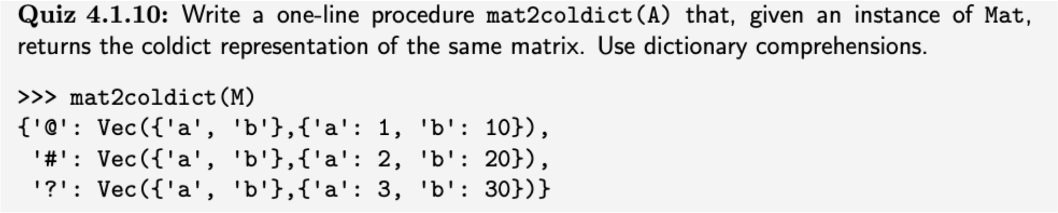
Here's a fixed up version of what we tried last class:

```
def mat2rowdict(A): #A is a Mat matrix
    D = A.D #this is the domain of the given matrix
    print("D",D)
    R = D[0] #gives the "rows"
    print("R",R)
    C = D[1] #gives the "columns"
    print("C",C)
    rowdict = {}
    for r in R: #we'll associate a vector
        pairs={(r,c) for c in C}
        print(pairs)
        row_vector = Vec(C,{c:A.f[(r,c)] for c in C})
        print(row_vector)
        rowdict[r] = row_vector
    return rowdict
```

Dict-of-columns representation of M

```
{'#': Vec({'a','b'}, {'a':2, 'b':20}),
 '@': Vec({'a','b'}, {'a':1, 'b':10}),
 '?': Vec({'a','b'}, {'a':3, 'b':30})}
```

7. Now try:



Quiz 4.1.10: Write a one-line procedure mat2coldict(A) that, given an instance of Mat, returns the coldict representation of the same matrix. Use dictionary comprehensions.

```
>>> mat2coldict(M)
{'@': Vec({'a', 'b'},{'a': 1, 'b': 10}),
 '#': Vec({'a', 'b'},{'a': 2, 'b': 20}),
 '?': Vec({'a', 'b'},{'a': 3, 'b': 30})}
```

8. What is the *transpose* of a matrix?

9. (**Quiz 4.4.2**) Write the procedure `transpose(M)` that, given an instance of `Mat` representing a matrix, returns the representation of the *transpose* of that matrix.

10. What is matrix-vector multiplication?

11. How can we code matrix-vector multiplication?

Getting your classwork recorded

When you are done, before you leave class...

(a) Click the “Print” menu choice (under “File”) and make a pdf of this worksheet (html is OK too).

(b) Send me an email (`clarson@vcu.edu`) with an informative header like “Math 310 - c16 worksheet attached” (so that it will be properly recorded).

(c) Remember to attach today’s classroom worksheet!