

Name: _____

Directions: MAKE SURE TO COPY YOUR ANSWERS TO A SEPARATE SHEET FOR SENDING ME AN ELECTRONIC COPY LATER.

1. Consider the Python list `x`, pp.13-14.
 - (a) (10) Write a Python expression for the element in row 1, column 0. Remember, row and column indices start at 0.
 - (b) (10) Write a Python expression for the entire row 1.
 - (c) (10) Is storage for a two-dimensional array (list, actually) row-major, column-major or neither? Assume we are working with tuples if you wish.

2. (10) What function in the `C` library is the inspiration for the `%` operator in line 4 of the middle code snippet on p.52?

3. (10) Consider the `seek()` function in the top code snippet, p.52. One of the operations in a certain physical object inspired this name. What object is that?

4. (25) The `handin` command on CSIF deposits student submissions to a directory named `handin/` in my account. Within that directory are subdirectories, e.g. `132quiz3` and `145quiz8`, and some miscellaneous nondirectory files. It is required that I place a file `users.deny` in each of those subdirectories. I had such a file (empty, actually) in my `handin` directory, and wanted to copy it to each subdirectory. I did so by writing the short script below and running it from within the `handin/` directory. Fill in the blanks. Your code should work for the case of any instructor using `handin`, not just me.

```
import os
flst = blank (a)
for fl in flst:
    if blank (b) :
        blank (c) (fl)
        blank (d) ('cp ../users.deny .')
        blank (e)
```

5. (25) The code below does a *flatten* operation, as mentioned in class. Nested lists are “flattened” to non-nested ones, so that for instance `[[1,2],8,[4,6],[3,5]]` becomes `[1,2,8,4,6,3,5]`. Fill in the blanks. *Your code must work for any number of levels of nesting.*

```
def flatten(x):
    build = []
    for elt in x:
        if blank (a) :
            blank (b)
        else:
            blank (c)
    return build
```

Examples:

```
>>> x
[[1, 2], 8, [4, 6, [3, 5]], 6, 10]
>>> flatten(x)
[1, 2, 8, 4, 6, 3, 5, 6, 10]
>>> z = [1,2,3]
>>> flatten(z)
[1, 2, 3]
>>> flatten([[1,2], 'a', 'b'])
[1, 2, 'a', 'b']
```

You'll need the `type()` function, which works as follows:

```
>>> y = [5,12,13]
>>> type(y)
<type 'list'>
>>> type(y) is list
True
>>> type(y) is tuple
False
>>> type(3)
<type 'int'>
```

Solutions:

1.a `x[1][0]`

1.b `x[1]`

1.c row-major

2. `sprintf()`

3. disk drive

4.

```
import os
flst = os.listdir('.')
for fl in flst:
    if os.path.isdir(fl):
        os.chdir(fl)
        os.system('cp ../users.deny .')
        os.chdir('..')
```

5.

```
def flatten(x):
    build = []
    for elt in x:
        if not type(elt) is list:
            build.append(elt)
        else:
            build.extend(flatten(elt))
    return build
```