CSci 169 Programming HW 2

25 points
Python 1
Due Wed. Apr. 17 at the beginning of class
Submission instructions at the end

1. (5 points) Write a function in python

```
def fun(x, y):
```

that if called with

```
fun(1, 2)
```

returns 3
and if called with

```
fun("hi", "there")
```

returns "hithere"

Your function should work for any input as long as both inputs are strings or both inputs are numbers.

2. (10 points) Write a function in python

```
def report(xs):
```

that if called with

```
report(["Jill", "Johnson", 87, "Billy", "Ray", "Cyrus", 78, "Rita",
"Yeats", 94, "Bobbie", "Sue", "Palmer", 72])
```

returns the string

```
"Jill, Billy Ray, Rita, Bobbie Sue, averaged 82.75."
```

Your function should work for any input list formatted similarly to the example, with first, possibly middle, and last names, followed by a score.

3. (10 points) Translate the C++ program here: http://math.scu.edu/~linnell/cs169s19/hw/hw2/main.cpp to Python (Quicksort):

Submission instructions: You will print out your code for each problem, stapling together multiple sheets (there will be deductions for unstapled homework!). Turn in the hardcopy at the beginning of class. You will ALSO submit all of your .py files as attachments, to

cs169scu@gmail.com (NOT Dr. Linnell's email!!)

The subject line of the email should be "CS169 HW2 YourLastName YourIDNumber "

Note: The following problem will be due with Programming HW 3 Wed 10/12, but I HIGHLY suggest you start it early (But wait to submit it with HW3).

1. (15 points) In this problem, you will write a Python program to solve the 8-Queens problem. In chess, a queen can move any number of squares horizontally, vertically, or diagonally. The 8-queens problem is the problem of trying to place eight queens on an empty chessboard in such

a way that no queen can attack any other queen. This problem is intriguing because there is no efficient algorithm known for solving the general problem. Rather, the straightforward algorithm of trying all possible placements is most often used in practice, with the only optimization being that each queen must be placed in a separate row and column:

a. Starting with the first row, try to place a queen in the current column.
b. If you can safely place a queen in that column, move on to the next column
c. If you are unable to safely place a queen in that column, go back to the previous column, and move that queen down to the next row where it can safely be placed. Move on to the next column.

Write a program in python to solve the 8-queens problem. Your program should produce as output an 8X8 diagram of the chessboard, with a 1 indicating the presence of a queen in a square and a 0 indicating the absence of a queen.

Hints: You can represent the board as either an 8X8 list or as a one-dimensional list with the ith item representing the row number of the queen in column i. You can solve this problem recursively or iteratively, but the recursive solution is usually much easier.