Lists, Aliasing, and Mutability

1. Write the non-destructive function median(a) that takes a list of floats and returns the median value, which is the value of the middle element, or the average of the two middle elements. If the list is empty, return None.

```
assert(median([]) == None))
assert(median([1.0, 2.0, 3.0]) == 2.0))
assert(median([3.1, 8.0, 3.2, 5.0]) == 5.6))
```

2. Write the destructive function rotateList(a, n) which takes a list a and an positive integer n, and destructively modifies the list so that each element is shifted to the right by n indices (including wraparound). The function should then return this new list.

```
assert(rotateList([], 1) == []))
assert(rotateList([1, 2, 3], 0) == [1, 2, 3]))
assert(rotateList([1, 2, 3], 2) == [2, 3, 1]))
assert(rotateList(["hi", "hey", "hello", "howdy"], 5) ==
["howdy", "hi", "hey", "hello"]))
```

3. Write the function sumTable(n) that takes in a number n and return a nxn table that holds the sum of i and j at the ith row and jth column.

```
L = [[0,1,2],
     [1,2,3],
     [2,3,4],
     [3,4,5]]
assert(sumTable(4) == L)
```

4. Write the function matrixMultiply(m1, m2) that takes two matrices (2D Lists) and returns a list that multiplies the values at the same index in each matrix together. The matrices will have the same number of rows and columns.

```
M1 = [[1,2,3],
      [4,5,6],
      [7,8,9]]
M2 = [[2,4,6],
      [1,3,5],
      [9,3,2]]
result = [[2,8,18],
          [4,15,30],
          [63,24,18]]
assert(matrixMultiply(M1, M2) == result)
```