CS 1301 Practice Problems

These are actual test problems from previous classes. To learn the most, solve these problems using only a pencil or pen and a piece of blank paper. Once you think you have them solved, move to a computer and see if you were right. If not, learn from your mistakes!

Create a Dictionary! (4 points)

You need to build a cash register program. Whenever the clerk enters the name of a vegetable, you must return the vegetables price, as shown in the table below. Create a dictionary named veggies2prices that holds this data.

"Corn"	0.25	"Cucumber"	0.99
"Spinach"	0.33	"Onion"	0.13

```
Veggies2prices =  { "Corn": 0.25, "Cucumber": 0.99, "Spinach": 0.33, 
"Onion": 0.13}
```

Write Code! (8 points)

Now that you have the dictionary, create a function **priceLookup(VeggieName)** that will return the vegetables price when it is given the vegetables name. For example, *priceLookup("Onion")* should return 0.13. Assume that the veggies2prices variable is defined outside of the function (i.e. it is a global variable, and must be identified as such.) *If the user calls your function with an invalid vegetable name, return None.*

```
def priceLookup(VeggieName):
     global Veggies2prices
     result =  Veggies2prices.get(VeggieName, None)
     return(result)
```

Write Code! (5 points)

Write a function **updatePrice(VeggieName, newPrice)** that will update the price for a vegetable in the dictionary. For example, *updatePrice("Corn", 0.49)* will update the dictionary such that the price of corn is now 0.49. Again, assume that veggies2prices is a global variable as above.

```
def updatePrice(VeggieName, newPrice):
     global Veggies2prices
     Veggies2prices[VeggieName] = newPrice
```

Times Tables: Write Code! (10 points)

You are hired to develop an educational software package. Your first job: Write a function **printTimestables()** that will print the times tables (up to 9) on the screen. When your function is called, it should print the following:

```
Times:  1       2       3       4       5       6       7       8       9
1       1       2       3       4       5       6       7       8       9
2       2       4       6       8       10      12      14      16      18
3       3       6       9       12      15      18      21      24      27
4       4       8       12      16      20      24      28      32      36
5       5       10      15      20      25      30      35      40      45
6       6       12      18      24      30      36      42      48      54
7       7       14      21      28      35      42      49      56      63
8       8       16      24      32      40      48      56      64      72
9       9       18      27      36      45      54      63      72      81
```

Note that your function must print a header (Times: 1...9) and a first column number that goes from 1..9, while the interior of the grid is the X * Y value.
Hint: Using two loops (one inside of the other) is an easy (but not the only) way to accomplish this.

See answer below, and replace the N with a 9.

Generic Times Tables: Write Code! (10 points)

Your boss was impressed with your 9x9 times table function. Now he wants you to modify the function so that it will work for for any sized times table. Write a printTimestable(N) function that will print a timestable from 1 up to N, for any positive number N.

```
def printTimestable( N):
     #print the first line:
     print “Times”,
     for num in range(1,N+1):
          print “\t”,
          print num,
     #End the line
     print “”

     #print the table itself:
     for x in range(1, N+1):
          print str(x),
          for y in range(1,N+1):
               print “\t”,
               print x*y,
          #End the line
          print “”
```

Write Code: Filter a File (10 points)

Write a function **filterFile(fileName, outputFile)** that will open a specified filename, read each line, and copy any line that does not contain the word “python” to the outputFile. Your function should keep a count of how many lines it omitted (contained the word “python”) and return that number. Be sure to properly close your files!

```
def filterFile(fileName, outputFile):
    count = 0
    f = open(fileName, “r”)
    fout = open(outputFile,”w”)

    line = f.readline()
    while( len(line) > 0):
        if (“python” in line):
            #Omit the line, and count it
            count = count + 1
        else:
            #copy the line!
            fout.write(line)
        #Read the next line!
        line = f.readline()

    #All done, close the files and return the count.
    f.close()
    fout.close()
    return(count)
```