```
#Lab 13 Solutions
#Q1 (a) True (b) False

#Q2: (a)
#a z=3, y=3/3=1, x=3-1=2 no exception raised
#else clause will be run so y=20 ===> prints 2 20 3
#b z=3, y=3/0 ZeroDivisionError exception is raised
#but Exception clause is listed first so y=10 ===> prints 0 10 3
#b z=3, y=3/0 ZeroDivisionError exception is raised
#and ZeroDivisionError clause is run so y=9 ===> prints 0 9 3

#Q4: This is one of the solutions. Many solutions are possible.
def mean_dict(x,y):
    z={}
    for k in y:
        if k in x:
            value = (x[k]+y[k])/2
            if value != 0:
                z[k] = value
    for k in x:
        if k not in y:
            z[k]=x[k]
    for k in y:
        if k not in x:
            z[k]=y[k]
    return(z)


a = {"a": 2, "b": 1, "c": 5, "w":9}
b = {"a": 3, "b": -1, "c": 3, "d": 3 }

print(mean_dict(a,b))

#Q5
def recursive_power(n,x):
    """returns n**x"""
    if x==0:
        return 1
    else:
        return n*recursive_power(n,x-1)
print(recursive_power(3,4))

#Q6
def get_diagonal(m):
    if len(m) != len(m[0]): #square matrix
        return ("Not a square matrix")

    diagonal=[]
    for i in range(len(m)):
        for j in range(len(m)):
            if i==j:
                diagonal.append(m[i][j])
    return(diagonal)

mat1 = [[1, 2, 3], [5, 4, 6], [9, 7, 8]]
mat2 = [[1, 2, 3], [5, 4, 6], [9, 7, 8], [10, 12, 11]]
print(get_diagonal(mat1)) # would return the list: [1, 4, 8]
print(get_diagonal(mat2)) # displays the message "Not a square matrix "

def mean_dict(d1,d2):
    d3=d1.copy() #start by a copy of d1
    for k in d2:
       if k in d3:
           mean = (d3[k]+d2[k])/2
           if (mean != 0):
               d3[k] = mean   # mean not zero, include in d3
           else:
               d3.pop(k) # mean is zero, remove from d3
       else:
           d3[k] = d2[k] # key not in d1, copy element
    return d3

x = {'a': 2, 'b': 1, 'c': 5}
y = {'a': 3, 'b': -1, 'c': 3, 'd': 3 }
print(mean_dict(x,y))
```