1. Write a function backwards(n), that when passed an integer n, prints out the digits of n in reverse order.

2. Redo question 1 recursively.

3. Write a function reverse(n), that when passed an integer n, returns the integer with the digits of n in reverse order. For example reverse(123) will return the single three digit integer 321.

4. Redo question 3 recursively. (This one is not so easy …)

5. Consider an nXn array “a”. We might label its columns 0 through n-1 (from the left) and -1 through –n from the right. Write a function sum-diagonals(a,n) where a is the two dimensional array, and n is either a positive number from 0-(n-1) or a negative number -1 through –n. The function will return the sum of the diagonal of the array a, which starts on the top row of a column n. If n is positive, the diagonal goes to the right, if n is negative, the diagonal goes to the left.

6. Redo problem 5 above where the sum is done with list comprehensions.

7. Write a function print_all_in_list(x) that takes a list, possibly with sublists (and sublists of those) where all the “final” values are integers. print_all_in_list(x) prints all of the integers in all the sublists of x, in order.

For example:

```
z=[[1,2,3,[4,5,6]],7,[8,9]]
```

print_all_in_list(z)

prints

```
  1  2  3  4  5  6  7  8  9
```

8. Write the following 3 functions:

```python
def get_next(n): Generate and return a list with eight digits representing the base 8 representation of the base 10 number n. Return that list as the value of this function.
```
def duplicates(q): if list q contains duplicate digits: return True else: return False

def diagonal_threat(q): if there is a diagonal “threat” (2 queens occupying the same diagonal) return True else: return False

9. Use the above functions to solve the 8 queens problem.

10. Consider the 5X5 square below:

Write a recursive function to solve this problem? How