Turn this in at the end of class for participation points!

Full Name: ______________________________ Section: ________

On this worksheet, you'll work through writing recursive functions step-by-step. For each function, you'll begin by coming up with your base and recursive cases, then come up with some test cases. Finally, you'll fill in the blanks to write the function.

1 factorial

Recall (from math class) that $5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$. Observe that $5! = 5 \cdot 4!$ and $4! = 4 \cdot 3!$. 1! is simply 1.

Describe your base case (in English, not code)	
Describe your recursive case (in English, not code)	
Write at least one function call that will return from your base case without making any recursive calls	
Write at least one function call that will require recursive calls (hint: the simpler the better!)	

```
def factorial(n):
    if ________________________ :
        return ________________________
    else:
        return ________________________ * factorial(________________________)
```

2 print_range

So far, we have seen examples in python that use loops to print numbers from `a` to `b`. We can also do this using recursion! For this exercise, you'll work on a function called `print_range` that prints numbers from `a` (inclusive) to `b` (inclusive). If `a` is greater than `b`, you don't need to print anything. As an example, `print_range(100, 101)` should print:

```
100
101
```

Describe your base case (in English, not code)	
Describe your recursive case (in English, not code)	
Write at least one function call that will return from your base case without making any recursive calls	
Write at least one function call that will require recursive calls (hint: the simpler the better!)	

```
def print_range(a, b):
    if ________________________:
        return # this will return None by default - no need to change
    else:
        print(________________________)
        print_range(________________________,________________________)
```

3 mask_other_chars

In class, we wrote a function called mask_other_chars that took a string s and a character c as input and returned the same string with characters besides c replaced with -. For instance mask_other_chars("hello", "l") would return "--ll-". Try to write a recursive function to solve this problem.

Describe your base case (in English, not code)	
Describe your recursive case (in English, not code)	
Write at least one function call that will return from your base case without making any recursive calls	
Write at least one function call that will require recursive calls (hint: the simpler the better!)	

```
def mask_other_chars(s, c):
    if ________________________:
        return ________________________
    elif ________________________:
        # note: + is for string concatenation
        return __________________ + mask_other_chars(________________________)
    else:
        return __________________ + mask_other_chars(________________________)
```

4 is_palindrome

Palindromes are strings that are same forward and backward. For example "ada", "step on no pets", etc. Your task is to write a recursive function that will check if a string is palindrome and return True if it is and False if it isn't (you can consider a string with a single character and the empty string to be palindrome).

Describe your base case (in English, not code)	
Describe your recursive case (in English, not code)	
Write at least one function call that will return from your base case without making any recursive calls	
Write at least one function call that will require recursive calls (hint: the simpler the better!)	

```
def is_palindrome(text):
    if ________________________:
        return ________________________
    elif ________________________:
        return is_palindrome(________________________)
    else:
        return ________________________
```