Assignment 3

Task 1 - Iterative Multiplication!

Implement a function that can perform multiplication by repeatedly adding the multiplier to 0.

```
def mul_iter(x, y):
    result = 0
    # do something in a for loop here
    return result
```

Task 2 - Recursive Multiplication!

Now, implement the same operation, only recursively. Think about what this means: you need to make the function call itself with a simpler version.

```
def mul_rec(x, y):
    if x == 0:
        # this is your base case - what needs returning?
    else:
        return y * # what goes here?
```

Task 3 - Strings!

Before we proceed to task 4, we just need to solidify some string operations.

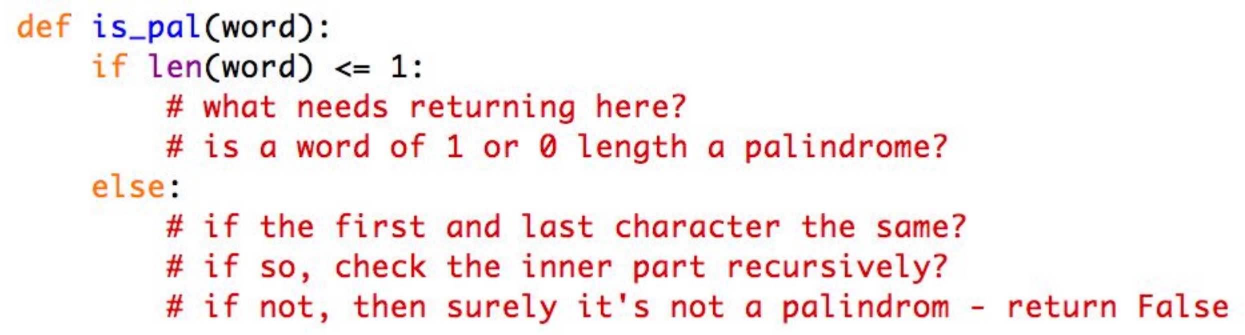
1. How do you print the first character of a string?
2. How do you print the last character of a string?
3. How do you take a substring, removing the first and last characters?

Feel free to look these up or ask for help. You need the ability to do these things to implement the recursive case in task 4.

Task 4 - Recursive Palindromes!

A palindrome is a word which reads the same backward or forward, such as madam or kayak.

Implement a function to check if a string is a palindrome.



```python
def is_pal(word):
    if len(word) <= 1:
        # what needs returning here?
        # is a word of 1 or 0 length a palindrome?
    else:
        # if the first and last character the same?
        # if so, check the inner part recursively?
        # if not, then surely it's not a palindrom - return False
```

Task 5 - Another look Palindromes!

There is actually another way to check if something is a palindrome! Simply reverse the string, and check if it's equal. Have a go at writing this too.

Having trouble reversing the string? Ask for help, or look it up.

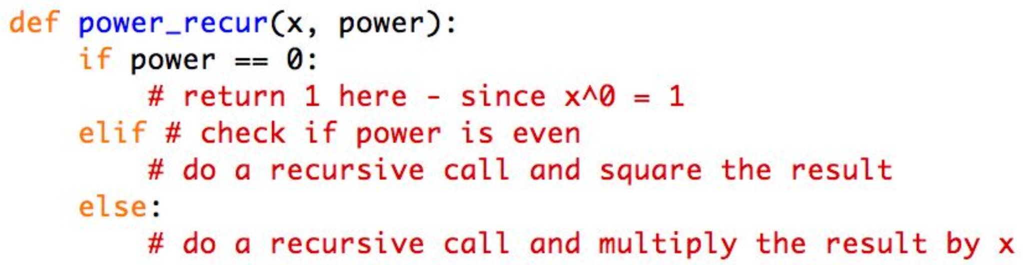
Task 6 - Factorials!

Write a function that computes factorials recursively. The structure will be similar to that of the recursive multiplication. Use the definition from the slides to help you too.

Additional: Try to implement this iteratively too. See how the recursive implementation is much more natural and expressive.

Task 7 - Powers!

Finally, write a function to recursively compute powers of a number. Think about splitting it in half and computing powers of that. Note that there are two very different recursive cases to deal with.



```
def power_recur(x, power):
    if power == 0:
        # return 1 here - since x^0 = 1
    elif # check if power is even
        # do a recursive call and square the result
    else:
        # do a recursive call and multiply the result by x
```