#CodeYork

Assignment 3

June XX, 2017

Task 1 - Recursive Factorials

We can use recursion to write a function that will compute the factorial of a number.

Here is some code to get you started:

```
def factorial(n):
      if n == X:
            return Y
      else:
            return Z * factorial(W)
```

Finish off this code by replacing the X, Y, Z, W with the correct expressions, and then test your function out to make sure it works.

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

Task 2 - String Manipulation

Recall from the first session that we can splice and access elements of lists. We can do the same with strings. If:

```
Word = "hello"
```

then, word[0] will give us “h”, word[-1] will give us “o”. **How do we get “ell”?**

Task 3 - Recursive Palindromes

A palindrome is a word which reads the same backward or forward, such as madam or kayak. **Implement a recursive function to check if a string is a palindrome.**

You may start with the following code:

```
def is_palindrome(word):
      if X:
            return True
      elif word[0] == Y:
            return is_palindrome(Z)
      else:
            return W
```

Just like in task 1, you'll need to replace the X, Y, Z, W with the correct expressions.

Hint: See how the *elif* block is the recursive case. We need to call it with a simpler case. What's the simpler case here (task 2 is a hint)?

Task 4 - Palindromes Again

Recall that we can reverse a string by using [::-1]. **Write an alternative version of the function** that checks if a string is a palindrome by reversing it and checking if it's equal.

Hint: The code inside the function should only be one line! Much shorter than in task 3.

Task 5 - Printing a List

Amazingly, we can use recursion to print a list! Consider the following code:

```
def print_list(ls):
    if len(ls) > 0:
        print(ls[0])
        print_list(ls[1:])
```

This works by printing the first element, and then printing the rest, recursively. The base case is implicit here, since it really is just "do nothing". See how this function does not "return", it only has the "side effect" of writing to the screen.

Before continuing, please make sure you really understand what *ls[1:]* does. It gives you all of the list, apart from the very first element.

Now, modify this code so that it prints the list in reverse order.

Hint: Think about where the print statement now needs to go.