# Offline Exercises

## Task 1

Given a string, return a string where for every char in the original, there are two chars.

***doubleChar("The") → "TThhee"  
doubleChar("AAbb") → "AAAAbbbb"  
doubleChar("Hi-There") → "HHii--TThheerree"***

## Task 2

A sandwich is two pieces of bread with something in between. Return the string that is between the first and last appearance of "bread" in the given string, or return the empty string "" if there are not two pieces of bread.

***getSandwich("breadjambread") → "jam"  
getSandwich("xxbreadjambreadyy") → "jam"  
getSandwich("xxbreadyy") → ""***

## Task 3

Given three ints, a b c, one of them is small, one is medium and one is large. Return true if the three values are evenly spaced, so the difference between small and medium is the same as the difference between medium and large.

***evenlySpaced(2, 4, 6) → true  
evenlySpaced(4, 6, 2) → true  
evenlySpaced(4, 6, 3) → false***

## Task 4

Given a string and an int n, return a string made of the first and last n chars from the string. The string length will be at least n.

***nTwice("Hello", 2) → "Helo"***

***nTwice("Chocolate", 3) → "Choate"***

***nTwice("Chocolate", 1) → "Ce"***

## Task 5

Given a string, return true if it ends in "ly".

***endsLy("oddly") → true***

***endsLy("y") → false***

***endsLy("oddy") → false***

## Task 6

Given a string, return recursively a "cleaned" string where adjacent chars that are the same have been reduced to a single char. So "yyzzza" yields "yza".  
***stringClean("yyzzza") → "yza"  
stringClean("abbbcdd") → "abcd"  
stringClean("Hello") → "Helo"***

## Task 7

The fibonacci sequence is a famous bit of mathematics, and it happens to have a recursive definition. The first two values in the sequence are 0 and 1 (essentially 2 base cases). Each subsequent value is the sum of the previous two values, so the whole sequence is: 0, 1, 1, 2, 3, 5, 8, 13, 21 and so on. Define a recursive fibonacci(n) method that returns the nth fibonacci number, with n=0 representing the start of the sequence.

***fibonacci(0) → 0  
fibonacci(1) → 1  
fibonacci(2) → 1***

## Task 8

We have a number of bunnies and each bunny has two big floppy ears. We want to compute the total number of ears across all the bunnies recursively (without loops or multiplication).

***bunnyEars(0) → 0  
bunnyEars(1) → 2  
bunnyEars(2) → 4***

## Task 9

Output a diamond in characters to the console window with loops that construct it

# https://image.freepik.com/free-icon/diamond-silhouette_318-40490.jpg