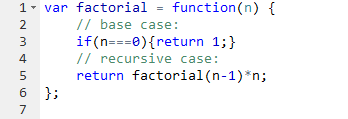
Recursion is when an algorithm solves a smaller instance of a problem, unless the problem is so small that the algorithm can solve it directly. Let’s take a look at how to compute the factorial function.

**Iterative factorial**

for (i=n; i>0; i--){ result = result \* i}

**Recursive factorial**

Much more interesting way of dealing with the problem, and with for loops in general. Pay attention. We are going to multiply our (n-1)! \*n for each iteration.



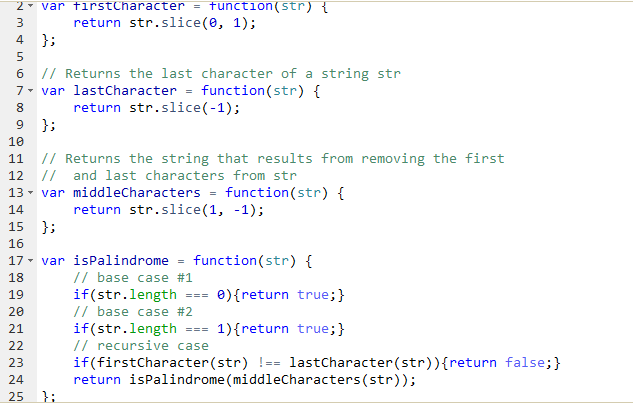
What we do here is solve a subproblem of the original problem and then use the solution of the smaller instance (n\*) to solve the rest.

1. Each recursive call should be on a smaller instance of the same problem, that is, a smaller subproblem.
2. The recursive calls must eventually reach a base case, which is solved without further recursion.

We are gonna now use recursion to figure out whether a word is a palindrome (a word that is spelled the same forward and backward).

Note : str.slice(-1) just takes the last letter of the string. str.slice(1,-1); returns all the letters barring the first and last.

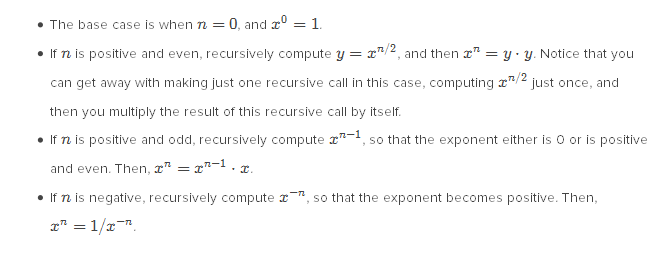
We first set up two slice functions, last and firstCharacter, which isolate respectively the first and last character of a string, and middleCharacters, which take a string a cut off the first and last character (so it’s the opposite of the two previous functions: firstCharacter(str)+middleCharacters(str)+lastCharacters(str) = str)



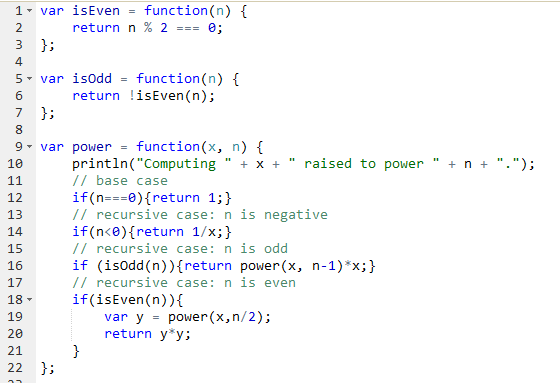
With each iteration, the program checks whether the string first and last letters are different, and, if not, cuts off both of them off the string, and runs the loop again for our new string.

Remember, we need to have set up a **base case** where the loop knows it has to stop before writing the recursion algorithm.

Computing powers of a number



Three case scenarios: n is even and positive: we compute y=x^n/2 and then do y\*y. If it is odd, y=x^n-1, and multiply y\*x. If it is negative, we compute (1/x)^n.



**The Sierpinski gasket**

This is recursion for graphical purposes (fractals)

