# Exercises: Strings

These exercises will test your knowledge of string functions, conditionals, and arrays. For many of them, you will want to consult the JavaScript strings reference to find useful string methods to call.

## DrEvil

Create a function called `DrEvil`. It should take a single argument, an amount, and return ' dollars', except it will add '(pinky)' at the end if the amount is 1 million. For example:

```
  DrEvil(10): 10 dollars
  DrEvil(1000000): 1000000 dollars (pinky)
```

# MixUp

Create a function called `mixUp`. It should take in two strings, and return the concatenation of the two strings (separated by a space) slicing out and swapping the first 2 characters of each. You can assume that the strings are at least 2 characters long. For example:

```
  mixUp('mix', pod'): 'pox mid'
  mixUp('dog', 'dinner'): 'dig donner'
```

Look up the JavaScript string reference to find methods which may be useful!

## FixStart

Create a function called `fixStart`. It should take a single argument, a string, and return a version where all occurences of its first character have been replaced with '*', except for the first character itself. You can assume that the string is at least one character long. For example:

```
fixStart('babble'): 'ba**le'
```

## Verbing

Create a function called `verbing`. It should take a single argument, a string. If its length is at least 3, it should add 'ing' to its end, unless it already ends in 'ing', in which case it should add 'ly' instead. If the string length is less than 3, it should leave it unchanged. For example:

```
verbing('swim'): 'swimming'
verbing('swimming'): 'swimmingly'
verbing('go'): 'go'
```

## Not Bad

- Create a function called `notBad` that takes a single argument, a string.
- It should find the first appearance of the substring 'not' and 'bad'.
- If the 'bad' follows the 'not', then it should replace the whole 'not'...'bad' substring with 'good' and return the result.
- If it doesn't find 'not' and 'bad' in the right sequence (or at all), just return the original sentence.

For example:

```
notBad('This dinner is not that bad!'): 'This dinner is good!'
notBad('This movie is not so bad!'): 'This movie is good!'
notBad('This dinner is bad!'): 'This dinner is bad!'
```