## **HackerRank**

## Ruby - Enumerable - collect

Beside simple methods to iterate over objects, Ruby Enumerable provides a number of important higher order constructs that we shall explore in further challenges. One of such important methods is collect method, also known as map.

map as the name may suggest, takes a function and maps (applies) it to a collection of values one by one and returns the collection of result.

That is,

$$ext{map}(f(x), [x_1, x_2, x_3, \dots, x_n]) o [f(x_1), f(x_2), \dots, f(x_n)]$$

This single powerful method helps us to operate on a large number of values at once.

For example,

```
>>> [1,2,3].map { |x| 2*x }
=> [2, 4, 6]
>>> {:a=>1, :b=>2, :c=>3}.collect { |key, value| 2*value }
=> [2, 4, 6]
```

Note that these methods are different from <a href="each">each</a> in the respect that these methods return a <a href="new">new</a> collection while former returns the original collection, irrespective of whatever happens inside the block.

In this challenge, your task is to write a method which takes an *array of strings* (containing secret enemy message bits!) and decodes its elements using ROT13 cipher system; returning an array containing the final messages.

For example, this is how ROT13 algorithm works,

## Original text:

```
Why did the chicken cross the road?
Gb trg gb gur bgure fvqr!
```

## On application of ROT13,

```
Jul qvq gur puvpxra pebff gur ebnq?
To get to the other side!
```