Joshua Bilsland

**Codewars Solutions**

A collection of solutions and things I have learned.

Contents

[8 Kyu Difficulty Challenges 2](#_Toc109635947)

[Even or Odd – 25/07/2022 2](#_Toc109635948)

[7 Kyu Difficulty Challenges 4](#_Toc109635949)

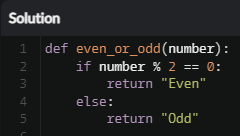
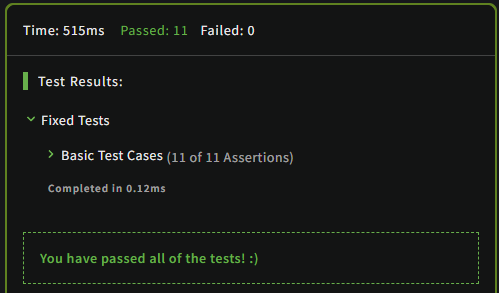
[Disemvowel Trolls – 23/01/2022 4](#_Toc109635950)

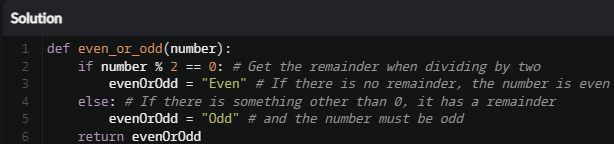
# 8 Kyu Difficulty Challenges

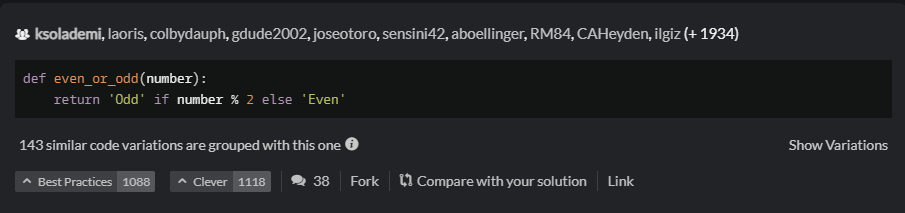
## Challenge descriptionEven or Odd – 25/07/2022

This challenge wants me to write a program that can take an integer, work out if it is odd or even, and then return the result.

I plan to make use of the modulus operator (%) to divide the integer by 2 and work out if the number is odd or even based on the returned value (which will be the remainder). My first solution can be seen below.

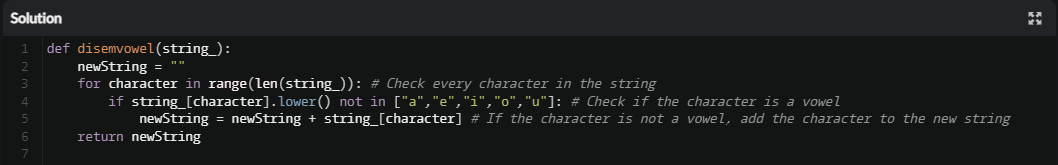
This code works, but it uses multiple return statements. For a program this small it doesn’t really matter but it is good practice to only have one return statement to make a function more readable. I have changed my program so that it follows the rule of ‘single exit’.

I submitted this solution and looked at some of the solutions submitted by the community. Most people had done something very similar to my solution. I also saw solutions that had similar logic but were condensed onto a single line. An example can be seen below.

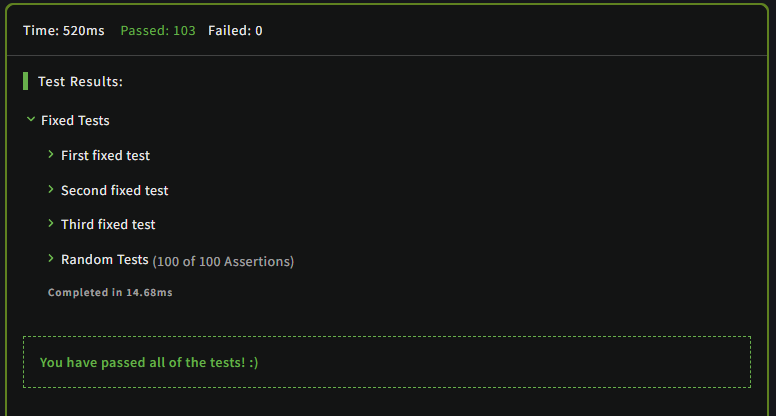
# 7 Kyu Difficulty Challenges

## Challenge descriptionDisemvowel Trolls – 23/01/2022

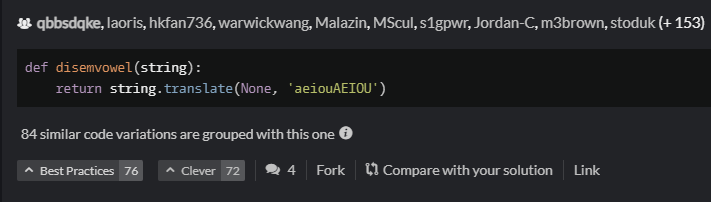
In short, this challenge wants me to create a program that can take a string, remove the vowels from it, and return the new string.

My first idea was to make use of count-controlled iteration paired with selection statements and the ‘not in’ operator. My program would go through each character in the string and check if it’s a vowel. If it is not a vowel, the character would be added to a new string variable which would be returned once every character has been checked. My solution can be seen below.

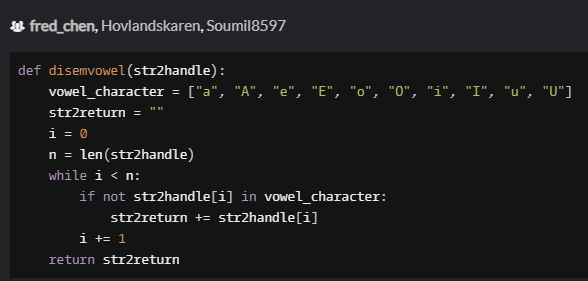
I tested the code and it worked as intended. I then submitted my solution.



I then looked at some of the solutions that the community had come up with. I found a much simpler solution which makes use of the ‘translate’ method. I have not seen this method before, so I will spend some time reading about this method and try to find a way of implementing it into my programs. The solution I found can be seen below.



I also found some solutions that seemed much more complicated than my solution and have lots of unnecessary and inefficient parts to them (not saying that mine is perfect). An example can be seen below.



One issue with this solution is that it checks all the vowels in their lowercase and uppercase forms via a list. This will mean that there will be ten comparisons being made for each character in the list. A better way of doing this would be to only put lower case letters in the list and use the ‘lower’ method on each character when making the comparisons. This will half the number of comparisons being made, save memory, and make the program faster.

## 