**Numbers to Words Converter**

At first look, this challenge seemed to be straight forward. In that, I could simply convert an integer into a string but found out of course that the type was the only thing that changed to an integer and not the actual value.

To me there were two ways I could approach this problem; Brute force and optimised using an algorithm

**Brute Force:**

The brute force method for me meant that I would have to create a string array which would include all the word representation of the number example:

Although this method is straight forward and easy to implement, as I would only have to take the user input and output the same index value of numrepresentation. This method however is very limited, un-optimised and time consuming as I would have to include all the word representation of the number.

**Optimised:**

For me to be able to accurately convert an integer to word is to come up with an algorithm which slices the numbers to make it easier to convert since numbers only contain 0 to 9, I would need to slice them up into smaller bits and use the length of the user input to be able to determine what the number unit (units, tens, hundred, thousand, ten-thousand, hundred thousand, million, etc) it is. For example:

* A four-digit number of (1234) has a length of 4 which easily lets us know that this number is a thousand unit, from here all we need to do is slice the numbers and assign them the appropriate word.
* Similarly, a number containing six-digits (100,000) has a length of 6 which we can use to determine that this number uses a hundred-thousand unit.

All the above seems promising but the question is, how do we slice the numbers into smaller bits? Well, there are many ways to do this. One way is to convert the user input into an array and assign each single digit an index number starting from 0,1,2…. n. Since the number can only be between 0 – 9 we can assign them to exact word representation and comment the quantity of units/number units (“hundred, thousand, etc”)

Rough Example:

This however seems un-optimised to me as I would have to create a for-loop against the length of the user input so that each digit is given an index as it passes through the loop which adds more time to the operation of this program.

Instead, I eliminated the need for the for-loop by using mod(%) and divide(/) to slice the numbers where ever I want and add the other slices to make a complete statement example code:

A screenshot of a computer

Description automatically generated with medium confidence

Since the user input can have a dot ”.” separating the dollar and cents apart, I simply made an if statement to check if the input contained a “.” After which I split the input into two parts the first containing the dollar and second containing cents.