# Recursive approach to convert Digits to Letter

The provided C++ program converts a given number into its word representation for each digit using a recursive approach. It uses an array **arr** to store the word representations of digits from "zero" to "nine."

**Recursive function to convert the digits of a number to words**

1. Inside the **sayDigits** function, there are two cases:
    - Base Case: If the **num** is 0, there are no more digits to convert, so the function returns.
    - Recursive Case: The function processes the digits from the most significant digit to the least significant digit recursively. It does this by making a recursive call to **sayDigits** with **num / 10** (which removes the least significant digit) and the same array **arr**. Then, it prints the word representation of the current digit **num % 10** using the **arr** array.
2. The program creates an array **arr** of strings to store the word representations of digits from "zero" to "nine".

**Time Complexity:**

T**he time complexity of the sayDigits function is O(log10(num)), where num is the input number. This is because, in each recursive call, the number is divided by 10,** effectively removing the least significant digit. The function makes log10(num) recursive calls until the number becomes 0, representing the number of digits in the input number.

**Space Complexity:**

**The space complexity of the program is O(log10(num)), where num is the input number. This is because the recursive calls in the sayDigits function create new frames on the call stack, and in the worst case, there can be log10(num) recursive calls, leading to O(log10(num)) space consumption on the call stack.** Additionally, the **arr** array has a constant size of 10, so it does not contribute significantly to the space complexity.

**Recursive call stack of the approach:**