Integer to English Words

## Description

Convert a non-negative integer num to its English words representation.

Example 1:

Input: num = 123

Output: "One Hundred Twenty Three"

Example 2:

Input: num = 12345

Output: "Twelve Thousand Three Hundred Forty Five"

Example 3:

Input: num = 1234567

Output: "One Million Two Hundred Thirty Four Thousand Five Hundred Sixty Seven"

## Algorithm

1.) **Initialize Constants**:

- Define two arrays: belowTwenty for numbers from 0 to 19 and tens for multiples of ten from 20 to 90.

2.) **Base Case for Zero**:

- If the input number num is 0, return the string "Zero".

3.) **Helper Function**:

- Use a recursive helper function to construct the English words representation of the number:
    - For numbers less than 20, use the belowTwenty array.
    - For numbers less than 100, use the tens array and recurse for the remainder.
    - For numbers less than 1000, use "Hundred" and recurse for the hundreds and remainder.
    - For larger numbers, use "Thousand", "Million", and "Billion", dividing the number and recursing accordingly.

4.) **Return the Result**:

- The helper function returns the constructed string, and numberToWords trims any extra spaces and returns the final result.

## Pseudocode

```
function numberToWords(num: int) -> String:
  if num == 0:
    return "Zero"

  return helper(num).trim()

function helper(num: int) -> String:
  result = ""

  if num < 20:
    result = belowTwenty[num]
  else if num < 100:
    result = tens[num / 10] + " " + belowTwenty[num % 10]
  else if num < 1000:
    result = helper(num / 100) + " Hundred " + helper(num % 100)
  else if num < 1000000:
    result = helper(num / 1000) + " Thousand " + helper(num % 1000)
  else if num < 1000000000:
    result = helper(num / 1000000) + " Million " + helper(num % 1000000)
  else:
    result = helper(num / 1000000000) + " Billion " + helper(num % 1000000000)

  return result.trim()
```

## Code

```
class Solution {
    private final String[] belowTwenty = {"", "One", "Two", "Three", "Four", "Five", "Six",
"Seven", "Eight", "Nine",
            "Ten", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen",
"Eighteen", "Nineteen"};
    private final String[] tens = {"", "", "Twenty", "Thirty", "Forty", "Fifty", "Sixty", "Seventy",
"Eighty", "Ninety"};

    public String numberToWords(int num) {
        if (num == 0) {
            return "Zero";
        }
        return helper(num);
    }

    private String helper(int num) {
        StringBuilder result = new StringBuilder();
        if (num < 20) {
            result.append(belowTwenty[num]);
        } else if (num < 100) {
            result.append(tens[num / 10]).append(" ").append(belowTwenty[num % 10]);
        } else if (num < 1000) {
            result.append(helper(num / 100)).append(" Hundred ").append(helper(num % 100));
        } else if (num < 1000000) {
            result.append(helper(num / 1000)).append(" Thousand ").append(helper(num %
1000));
        } else if (num < 1000000000) {
            result.append(helper(num / 1000000)).append(" Million ").append(helper(num %
1000000));
        } else {
            result.append(helper(num / 1000000000)).append(" Billion ").append(helper(num % 1000000000));
        }
        return result.toString().trim();
    }
}
```

## Conclusion

The numberToWords function converts an integer to its English words representation by using a recursive approach. It handles different ranges of numbers (units, tens, hundreds, thousands, millions, billions) by dividing the number and recursively processing each part. The use of arrays for numbers below 20 and multiples of ten simplifies the construction of the English words representation. This approach ensures that the function can handle any integer within the valid range and return its correct English words representation.