Integer to English Words

Convert a non-negative integer to its english words representation. Given input is guaranteed to be less than $2^{31} - 1$.

For example,

```
123 -> "One Hundred Twenty Three"
12345 -> "Twelve Thousand Three Hundred Forty Five"
1234567 -> "One Million Two Hundred Thirty Four Thousand Five Hundred Sixty Seven"
```

1. Did you see a pattern in dividing the number into chunk of words? For example, 123 and 123000.
2. Group the number by thousands (3 digits). You can write a helper function that takes a number less than 1000 and convert just that chunk to words.
3. There are many edge cases. What are some good test cases? Does your code work with input such as 0? Or 1000010? (middle chunk is zero and should not be printed out)

Solution 1

```
private final String[] lessThan20 = {"", "One", "Two", "Three", "Four", "Five", "
Six", "Seven", "Eight", "Nine", "Ten", "Eleven", "Twelve", "Thirteen", "Fourteen"
, "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Nineteen"};
private final String[] tens = {"", "Ten", "Twenty", "Thirty", "Forty", "Fifty", "
Sixty", "Seventy", "Eighty", "Ninety"};
private final String[] thousands = {"", "Thousand", "Million", "Billion"};

public String numberToWords(int num) {
    if (num == 0)
        return "Zero";
    int i = 0;
    String words = "";

    while (num > 0) {
        if (num % 1000 != 0)
            words = helper(num % 1000) + thousands[i] + " " + words;
        num /= 1000;
        i++;
    }

    return words.trim();
}

private String helper(int num) {
    if (num == 0)
        return "";
    else if (num < 20)
        return lessThan20[num] + " ";
    else if (num < 100)
        return tens[num / 10] + " " + helper(num % 10);
    else
        return lessThan20[num / 100] + " Hundred " + helper(num % 100);
}
```

written by hwy_2015 original link here

Solution 2

```
class Solution {
public:
    static string numberToWords(int n) {
        if(n == 0) return "Zero";
        else return int_string(n).substr(1);
    }
private:
    static const char * const below_20[];
    static const char * const below_100[];
    static string int_string(int n) {
        if(n >= 1000000000)   return int_string(n / 1000000000) + " Billion" + in
t_string(n - 1000000000 * (n / 1000000000));
        else if(n >= 1000000) return int_string(n / 1000000) + " Million" + int_s
tring(n - 1000000 * (n / 1000000));
        else if(n >= 1000)    return int_string(n / 1000) + " Thousand" + int_str
ing(n - 1000 * (n / 1000));
        else if(n >= 100)     return int_string(n / 100) + " Hundred" + int_strin
g(n - 100 * (n / 100));
        else if(n >= 20)      return string(" ") + below_100[n / 10 - 2] + int_s
tring(n - 10 * (n / 10));
        else if(n >= 1)       return string(" ") + below_20[n - 1];
        else return "";
        }
    }
};

const char * const Solution::below_20[] =  {"One", "Two", "Three", "Four","Five",
"Six","Seven","Eight","Nine","Ten", "Eleven","Twelve","Thirteen","Fourteen","Fift
een","Sixteen","Seventeen","Eighteen","Nineteen"};
const char * const Solution::below_100[] = {"Twenty", "Thirty", "Forty", "Fifty",
"Sixty", "Seventy", "Eighty", "Ninety"};
```

written by popffabrik original link here

Solution 3

```
public class Solution {
    private final String[] belowTen = new String[] {"", "One", "Two", "Three", "F
our", "Five", "Six", "Seven", "Eight", "Nine"};
    private final String[] belowTwenty = new String[] {"Ten", "Eleven", "Twelve",
"Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Nineteen"
};
    private final String[] belowHundred = new String[] {"", "Ten", "Twenty", "Thi
rty", "Forty", "Fifty", "Sixty", "Seventy", "Eighty", "Ninety"};

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

    private String helper(int num) {
        String result = new String();
        if (num < 10) result = belowTen[num];
        else if (num < 20) result = belowTwenty[num -10];
        else if (num < 100) result = belowHundred[num/10] + " " + helper(num % 10
);
        else if (num < 1000) result = helper(num/100) + " Hundred " +  helper(num
% 100);
        else if (num < 1000000) result = helper(num/1000) + " Thousand " +  helpe
r(num % 1000);
        else if (num < 1000000000) result = helper(num/1000000) + " Million " +
helper(num % 1000000);
        else result = helper(num/1000000000) + " Billion " + helper(num % 1000000
000);
        return result.trim();
    }
}
```

written by iuri.srb original link here