Decode Variations

Decode Variations

A letter can be encoded to a number in the following way:

```
'A' -> '1', 'B' -> '2', 'C' -> '3', ..., 'Z' -> '26'
```

A message is a string of uppercase letters, and it is encoded first using this scheme. For example, `'AZB' -> '1262'`

Given a string of digits `S` from `0-9` representing an encoded message, return the number of ways to decode it.

Examples:

```
input:  S = '1262'
output: 3
explanation: There are 3 messages that encode to '1262': 'AZB', 'ABFB',
and 'LFB'.
```

Constraints:

- **[time limit] 5000ms**
- **[input] string** `S`
  - 1 ≤ S.length ≤ 12
- **[output] integer**

Approach

Let's consider the String in questions is given as below

```
const string = '1262';
```

This input string has 3 possibilities

Possibilities	Characters
'1', '2', '6', '2'	'A', 'B', 'F', 'B'
'1','26','2'	'A', 'Z', 'B'
'12','6','2'	'L', 'F', 'B'

So the possibile occurance of Characters as 3 => 'ABFB', 'AZB', 'LFB'

If we look closely on how we figured this out, We approached the group the array as follows:

1. As individual numbers first - `[1,2,6,2]`
2. As pairs of numbers second - `[26 & 12]`

This problem can be broken down into 2 functions

- Function to convert String to Character Array

```
const arr = str.split(''); //gives out the character array
```

- Function to recursively traverse through the array in the following manner
    - As single characters -> '1', '2', '6', '2'
    - As pairs of characters -> '12', '26'

```
const decodeVarAlg = (arr, n=arr.length) => {
    // base case
    if (n == '0' || n == '1') return 1;
    let count = 0;
    // traverse through the array as individual characters
    if (arr[n-1] > '0') count  = decodeVarAlg(arr, n-1);
    // traverse through the array as pairs of characters
    if (arr[n-2] == '1' || arr[n-2] == '2' && arr[n-1] < '7') count +=
decodeVarAlg(arr, n-2);
    return count;
};

function decodeVariations(str) {
    const arr = str.split('');
    return decodeVarAlg(arr);
}


console.log(decodeVariations('1262')); // 3
```