Decode a string (encoded with number followed by string)

Posted on August 28, 2019 | by Prashant Yadav

Posted in Algorithms, Stack, String | Tagged medium

Practically prepare for your JavaScript interview

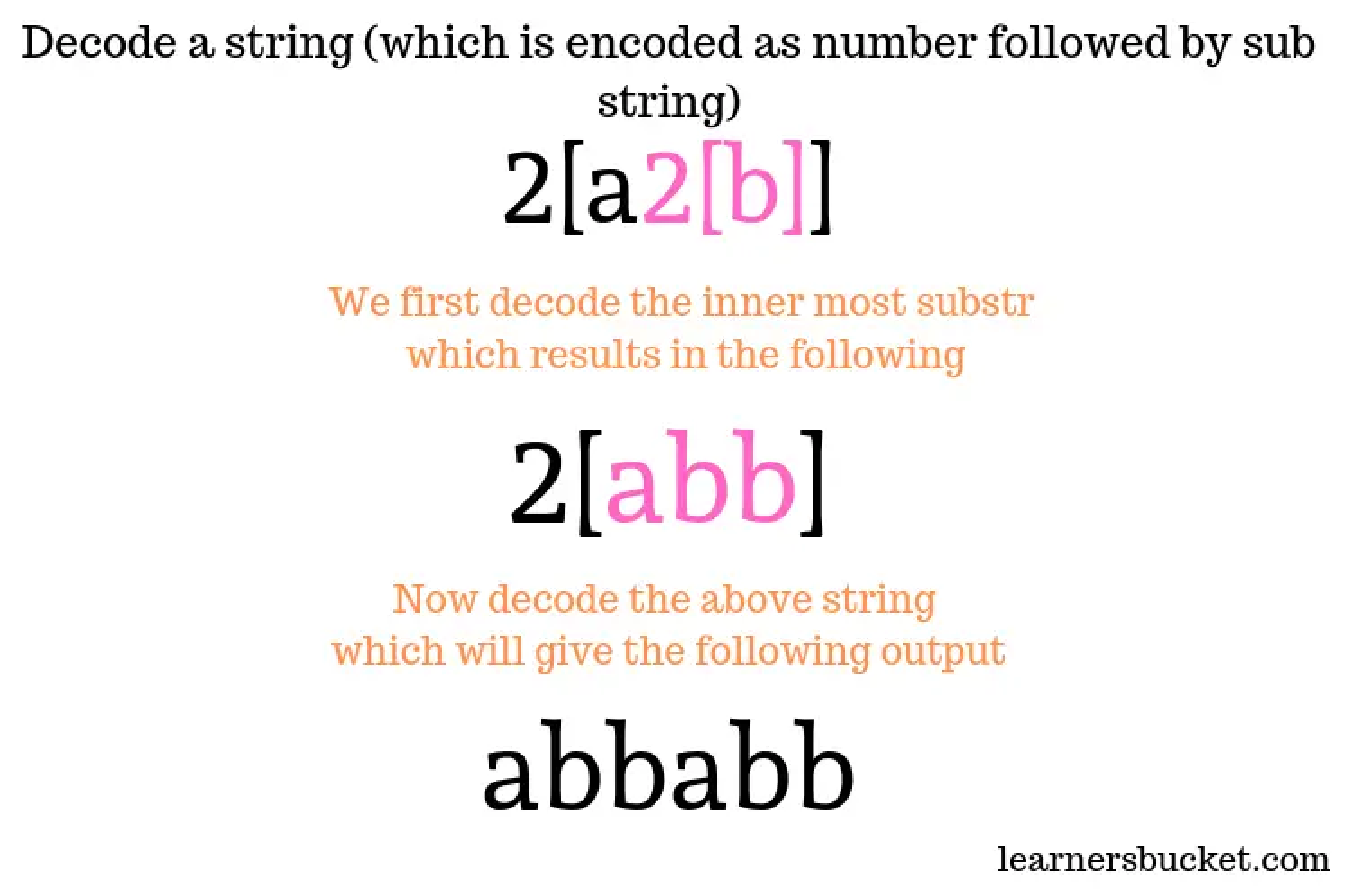
- JavaScript Revision
- JavaScript-Concept Based Problems
- Data Structures
- Algorithms
- Machine Coding
- Web Fundamentals

An algorithm to decode a string which is encoded in a pattern where a substring is wrapped in square brackets lead by a number.

Example

```
Input:
"2[a2[b]]"
"3[b2[ca]]"

output:
"abbabb"
"bcacabcacabcaca"
```


Decode a string (which is encoded as number followed by sub string)
2[a2[b]]
We first decode the inner most substr which results in the following
2[abb]
Now decode the above string which will give the following output
abbabb
learnersbucket.com

Using two stacks to decode a string.

Implementation

- We will use two different stacks, one to store the count i.e numbers `numStack` and second to the store the sub string `charStack`.
- We will iterate the whole string on each character and check
- If the current char is number then push it to the `numStack`.
- Else if the character is opening square bracket `'['` then check if there is a count number assigned to it or not. If it is then it may have already been pushed in the first condition so just add the character to `charStack` else add the current character to `charStack` and `1` to the `numStack`.
- If the character is closing square bracket `']'` then get the count from `numStack` and sub string from `charStack` and decode them, then again add the decoded string back to the `charStack` so that in the next iteration decoded sub string will be repeated along with the parent sub string.
- Else the character is alphabet so add it to the `charStack`.
- In the end create a string from the `charStack` (which will have decoded substring) and return it.

Copy

```
let decodeString = (str) => {
  let numStack = new Stack();
  let charStack = new Stack();
  let decoded = "", temp ="";

  for(let i = 0; i < str.length; i++){
    let count = 0;
    let val = str[i];

    //If char is number then
    //push to numStack
    if(/^\d+$/.test(val)){

      numStack.push(val);

    }else if(val === '['){
      //Else if open bracket and previous character is number
      //Then it will already added to numStack in the above (if condition)
      //Just add the char to charStack
      if(/^\d+$/.test(str[i-1])){

        charStack.push(str[i]);

      }else{

        //Else add 1 to numstack
        //And char to charStack
        charStack.push(str[i]);
        numStack.push(1);

      }
    }else if(val === ']'){

      //If close bracket
      //Reset temp and count
      temp = "";
      count = 0;

      //Get the count from numStack
      count = !numStack.isEmpty() && numStack.pop();

      //Get the subStr from charStack
      while(!charStack.isEmpty() && charStack.peek() !== '['){
        temp = charStack.pop() + temp;
      }

      //Remove the '[' char from charStack
      if(!charStack.isEmpty() && charStack.peek() === '['){
        charStack.pop();
      }

      //Create the repeat subStr
      decoded = temp.repeat(count);

      //Push the newlyCreated subStr to charStack again
      for(let j = 0; j < decoded.length; j++){
        charStack.push(decoded[j]);
      }

      //reset the string
      decoded = "";
      } else{
        //If alpha character then add to charStack
        charStack.push(val);

      }
    }

    //Form the decoded string from charStack
    while(!charStack.isEmpty()){
      decoded = charStack.pop() + decoded;
    }

    //Return the decoded str
    return decoded;
  }
```

Copy

```
Input:
console.log(decodeString("2[a2[b]]"));
console.log(decodeString("3[b2[ca]]"));

Output:
"abbabb"
"bcacabcacabcaca"
```

Time complexity: O(n ^ 2).

Space complexity: O(n + n).

Recommended Posts:

- Word Break Problem
- Longest common subsequence | Print all LCS
- Program to print the Collatz sequence in javascript.
- Find numbers that appear twice in an array.
- Search in a sorted rotated array
- Sort a stack using another stack
- javascript program to find largest of 2 numbers
- Quick sort using linked list
- Find distinct ways to climb the stairs in javascript.
- Program to sort only positive numbers of the array

Prev Next