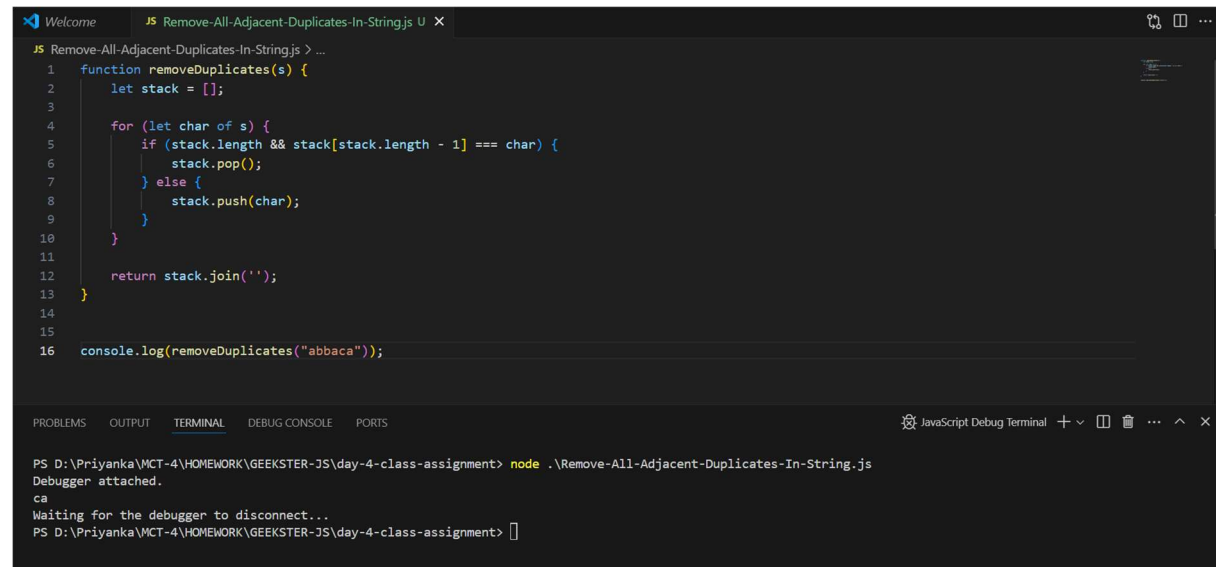


Remove all adjacent duplicates from a string:- <https://leetcode.com/problems/remove-all-adjacent-duplicates-in-string>

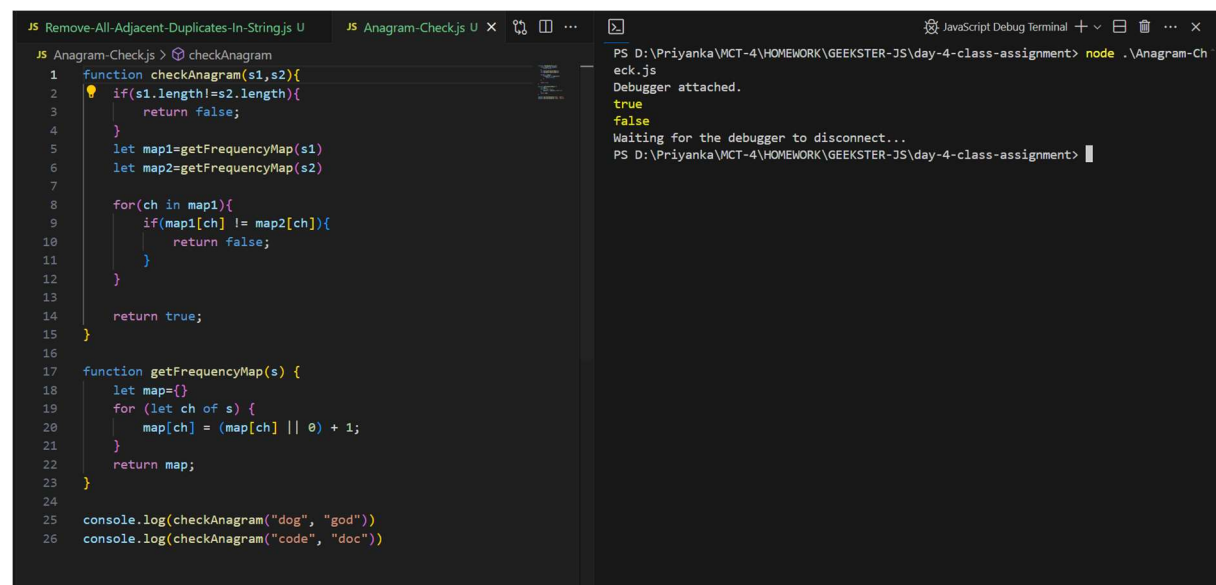


```
JS Remove-All-Adjacent-Duplicates-In-String.js > ...
1 function removeDuplicates(s) {
2   let stack = [];
3
4   for (let char of s) {
5     if (stack.length && stack[stack.length - 1] === char) {
6       stack.pop();
7     } else {
8       stack.push(char);
9     }
10  }
11
12  return stack.join('');
13 }
14
15
16 console.log(removeDuplicates("abbaca"));
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE PORTS JavaScript Debug Terminal + - [] ... ^ x

PS D:\Priyanka\MCT-4\HOMEWORK\GEEKSTER-JS\day-4-class-assignment> node .\Remove-All-Adjacent-Duplicates-In-String.js
Debugger attached.
ca
Waiting for the debugger to disconnect...
PS D:\Priyanka\MCT-4\HOMEWORK\GEEKSTER-JS\day-4-class-assignment> |

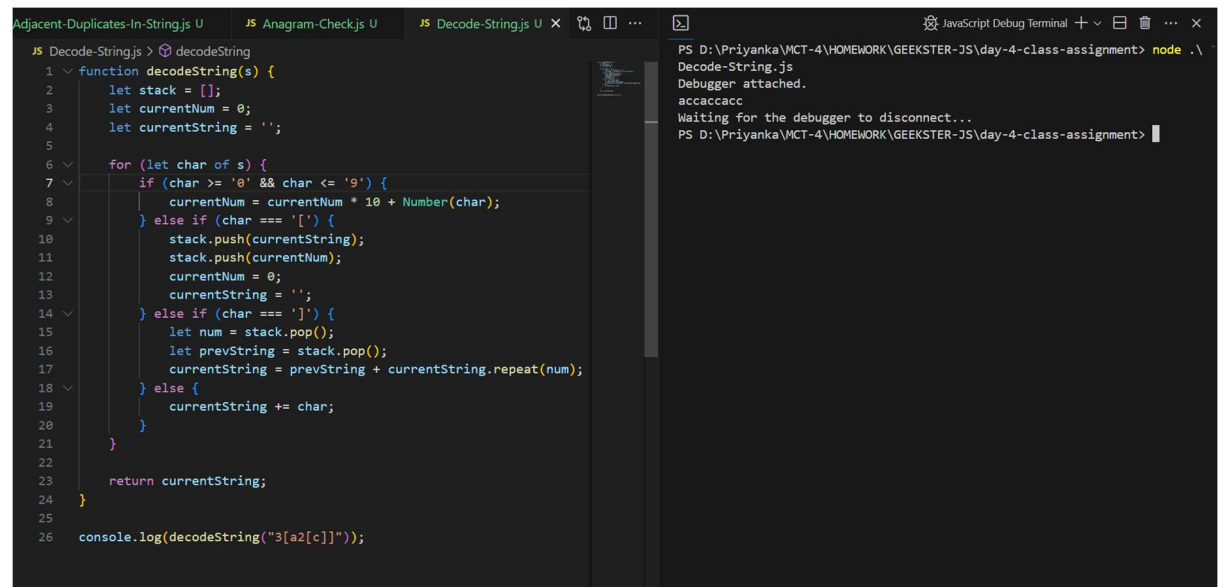
Valid Anagram:- <https://leetcode.com/problems/valid-anagram>



```
JS Remove-All-Adjacent-Duplicates-In-String.js U JS Anagram-Check.js U x [ ] ... [ ] JavaScript Debug Terminal + - [ ] ... x
JS Anagram-Check.js > checkAnagram
1 function checkAnagram(s1,s2){
2   if(s1.length!=s2.length){
3     return false;
4   }
5   let map1=getFrequencyMap(s1)
6   let map2=getFrequencyMap(s2)
7
8   for(ch in map1){
9     if(map1[ch] != map2[ch]){
10      return false;
11    }
12  }
13
14  return true;
15 }
16
17 function getFrequencyMap(s) {
18   let map={}
19   for (let ch of s) {
20     map[ch] = (map[ch] || 0) + 1;
21   }
22   return map;
23 }
24
25 console.log(checkAnagram("dog", "god"))
26 console.log(checkAnagram("code", "doc"))
```

PS D:\Priyanka\MCT-4\HOMEWORK\GEEKSTER-JS\day-4-class-assignment> node .\Anagram-Check.js
Debugger attached.
true
false
Waiting for the debugger to disconnect...
PS D:\Priyanka\MCT-4\HOMEWORK\GEEKSTER-JS\day-4-class-assignment> |

Decode String:- <https://leetcode.com/problems/decode-string/>

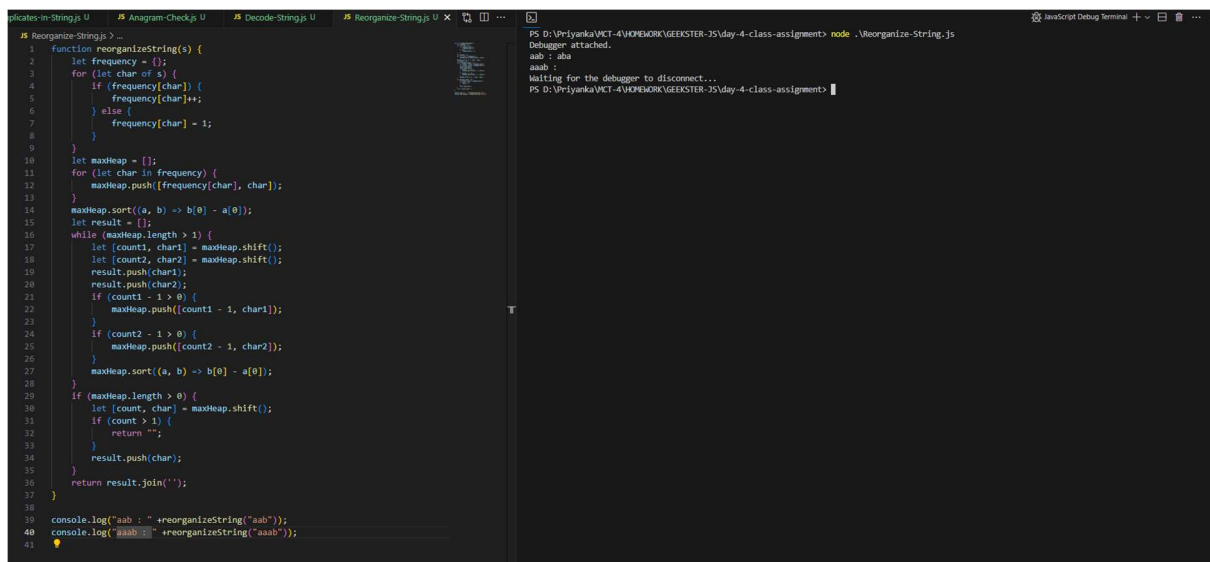


The screenshot shows a VS Code editor with a file named `Decode-String.js`. The code implements a function `decodeString(s)` that decodes a string with escape sequences like `3[a2[c]]`. The function uses a stack to handle nested sequences. The terminal on the right shows the command `node .\Decode-String.js` being executed, resulting in the output `accaccacc`.

```
1 function decodeString(s) {
2   let stack = [];
3   let currentNum = 0;
4   let currentString = '';
5
6   for (let char of s) {
7     if (char >= '0' && char <= '9') {
8       currentNum = currentNum * 10 + Number(char);
9     } else if (char === '[') {
10      stack.push(currentString);
11      stack.push(currentNum);
12      currentNum = 0;
13      currentString = '';
14    } else if (char === ']') {
15      let num = stack.pop();
16      let prevString = stack.pop();
17      currentString = prevString + currentString.repeat(num);
18    } else {
19      currentString += char;
20    }
21  }
22
23  return currentString;
24 }
25
26 console.log(decodeString("3[a2[c]]"));
```

PS D:\Priyanka\MCT-4\HOMEWORK\GEEKSTER-JS\day-4-class-assignment> node .\Decode-String.js
Debugger attached.
accaccacc
Waiting for the debugger to disconnect...
PS D:\Priyanka\MCT-4\HOMEWORK\GEEKSTER-JS\day-4-class-assignment>

Reorganize String:- <https://leetcode.com/problems/reorganize-string/>



The screenshot shows a VS Code editor with a file named `Reorganize-String.js`. The code implements a function `reorganizeString(s)` that reorganizes a string so that no two adjacent characters are the same. It uses a max heap to prioritize characters with higher frequencies. The terminal on the right shows the command `node .\Reorganize-String.js` being executed, resulting in the output `aabab`.

```
1 function reorganizeString(s) {
2   let frequency = {};
3   for (let char of s) {
4     if (frequency[char]) {
5       frequency[char]++;
6     } else {
7       frequency[char] = 1;
8     }
9   }
10  let maxHeap = [];
11  for (let char in frequency) {
12    maxHeap.push([frequency[char], char]);
13  }
14  maxHeap.sort((a, b) => b[0] - a[0]);
15  let result = [];
16  while (maxHeap.length > 0) {
17    let [count1, char1] = maxHeap.shift();
18    let [count2, char2] = maxHeap.shift();
19    result.push(char1);
20    result.push(char2);
21    if (count1 - 1 > 0) {
22      maxHeap.push([count1 - 1, char1]);
23    }
24    if (count2 - 1 > 0) {
25      maxHeap.push([count2 - 1, char2]);
26    }
27    maxHeap.sort((a, b) => b[0] - a[0]);
28  }
29  if (maxHeap.length > 0) {
30    let [count, char] = maxHeap.shift();
31    if (count > 1) {
32      return "";
33    }
34    result.push(char);
35  }
36  return result.join("");
37 }
38
39 console.log("aab : " + reorganizeString("aab"));
40 console.log("aaab : " + reorganizeString("aaab"));
41
```

PS D:\Priyanka\MCT-4\HOMEWORK\GEEKSTER-JS\day-4-class-assignment> node .\Reorganize-String.js
Debugger attached.
aab : aab
aaab :
Waiting for the debugger to disconnect...
PS D:\Priyanka\MCT-4\HOMEWORK\GEEKSTER-JS\day-4-class-assignment>