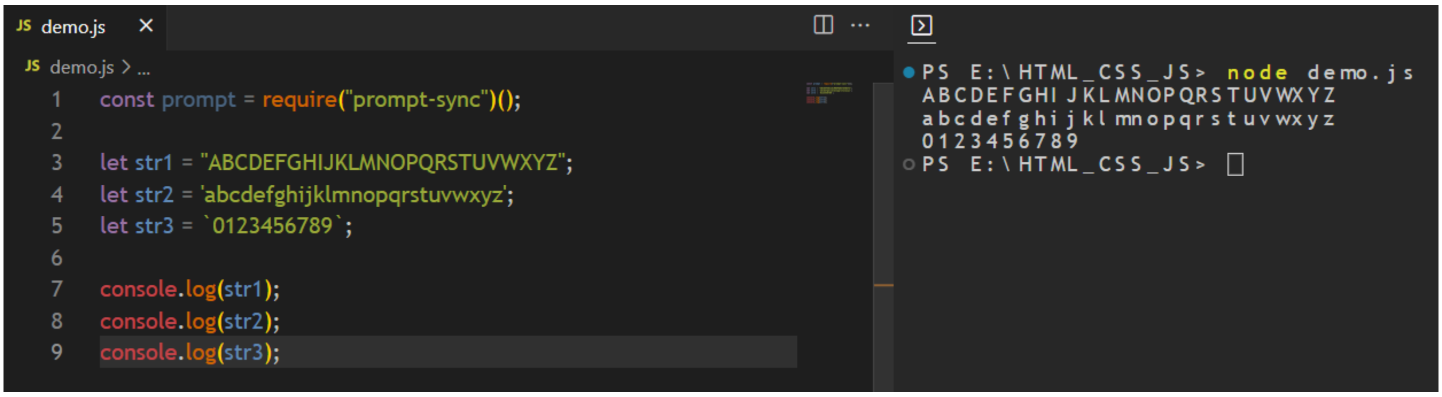
Data Structure & Algorithm

String In Java Script:

String is the character array which is used for to give String input to the system. in JavaScript you can give a String value by adding single quote mark(‘’), double quote mark(“”) and tilt symbol(``).



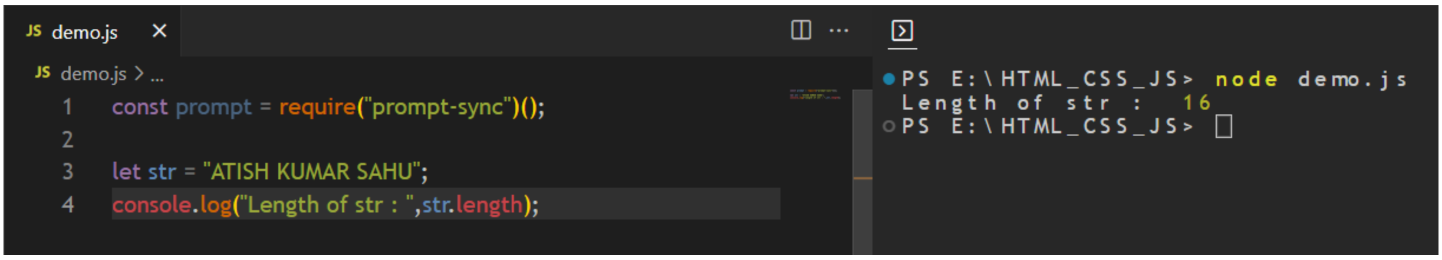
```javascript
const prompt = require("prompt-sync")();

let str1 = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
let str2 = 'abcdefghijklmnopqrstuvwxyz';
let str3 = `0123456789`;

console.log(str1);
console.log(str2);
console.log(str3);
```

```
PS E:\HTML_CSS_JS> node demo.js
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
0123456789
PS E:\HTML_CSS_JS>
```

Strings enclosed in single quotes work just like strings enclosed in double quotes. They are interchangeable in JavaScript. Like single quotes, double quotes are used to define strings. They are also interchangeable with single quotes in JavaScript. Backticks, also known as template literals, were introduced in ECMAScript 6 (ES6). They allow for more advanced string formatting. One significant feature is the ability to include placeholders for variables directly within the string using ${}

String Methods:

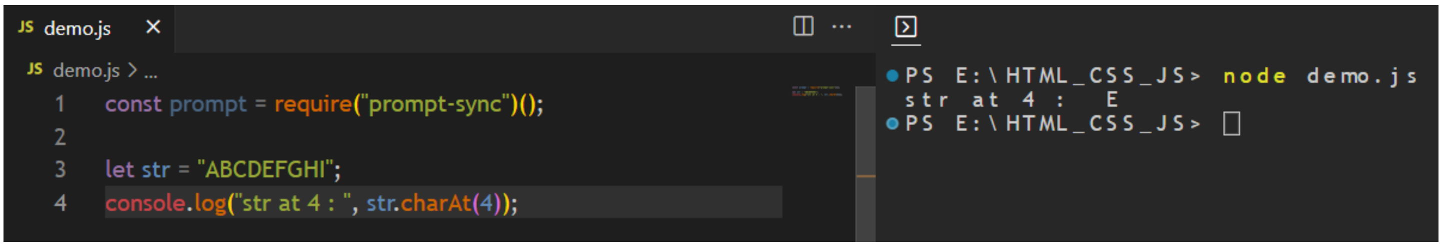
length: length method is used for to find the length of the string.



```javascript
const prompt = require("prompt-sync")();

let str = "ATISH KUMAR SAHU";
console.log("Length of str : ",str.length);
```

```
PS E:\HTML_CSS_JS> node demo.js
Length of str :  16
PS E:\HTML_CSS_JS>
```

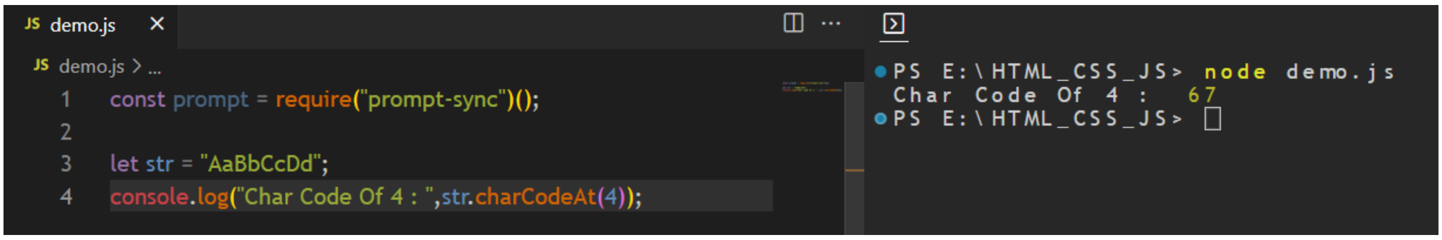
charAt(): this function is used for to find the value of that character in the given number count starts from 0.



```javascript
const prompt = require("prompt-sync")();

let str = "ABCDEFGHI";
console.log("str at 4 : ", str.charAt(4));
```

```
PS E:\HTML_CSS_JS> node demo.js
str at 4 :  E
PS E:\HTML_CSS_JS>
```

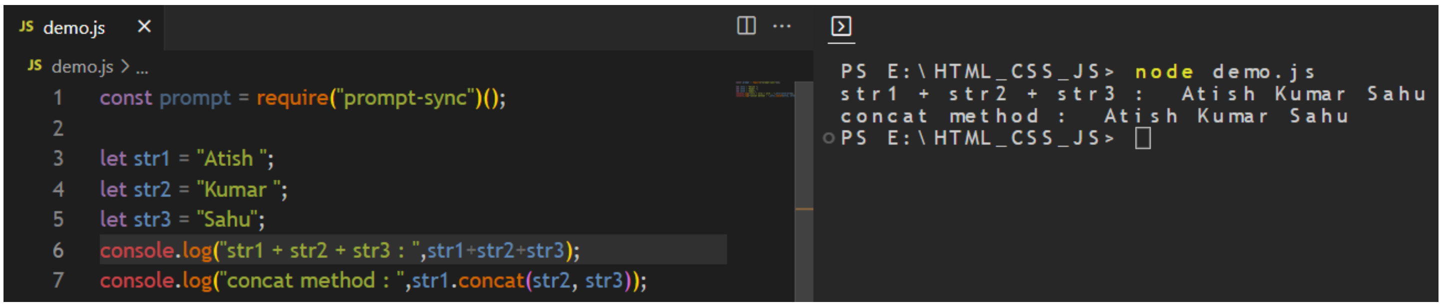
charCodeAt(): this function is used for to find the ascii code of that character in the given number count starts from 0.



```javascript
const prompt = require("prompt-sync")();

let str = "AaBbCcDd";
console.log("Char Code Of 4 : ",str.charCodeAt(4));
```

```
PS E:\HTML_CSS_JS> node demo.js
Char Code Of 4 :  67
PS E:\HTML_CSS_JS>
```

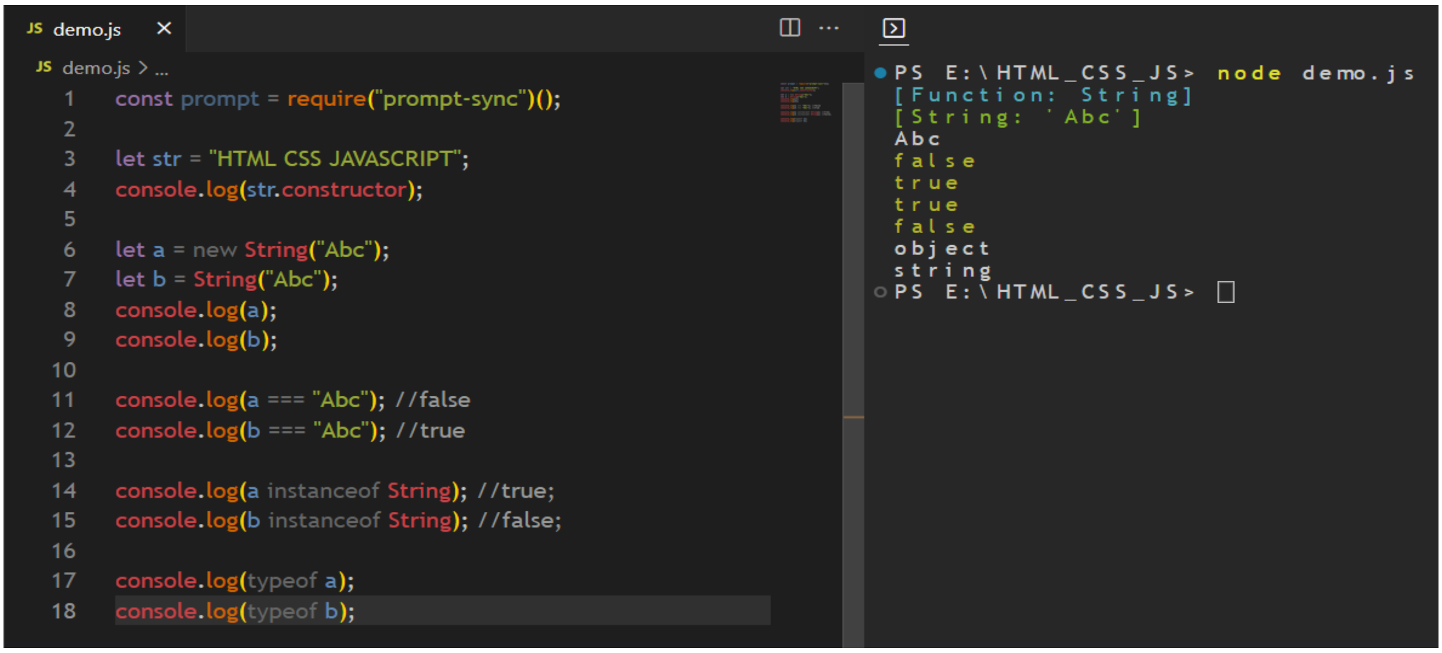
concat(): this function is used for to add two string values



```javascript
const prompt = require("prompt-sync")();

let str1 = "Atish ";
let str2 = "Kumar ";
let str3 = "Sahu";
console.log("str1 + str2 + str3 : ",str1+str2+str3);
console.log("concat method : ",str1.concat(str2, str3));
```

```
PS E:\HTML_CSS_JS> node demo.js
str1 + str2 + str3 :  Atish Kumar Sahu
concat method :  Atish Kumar Sahu
PS E:\HTML_CSS_JS>
```

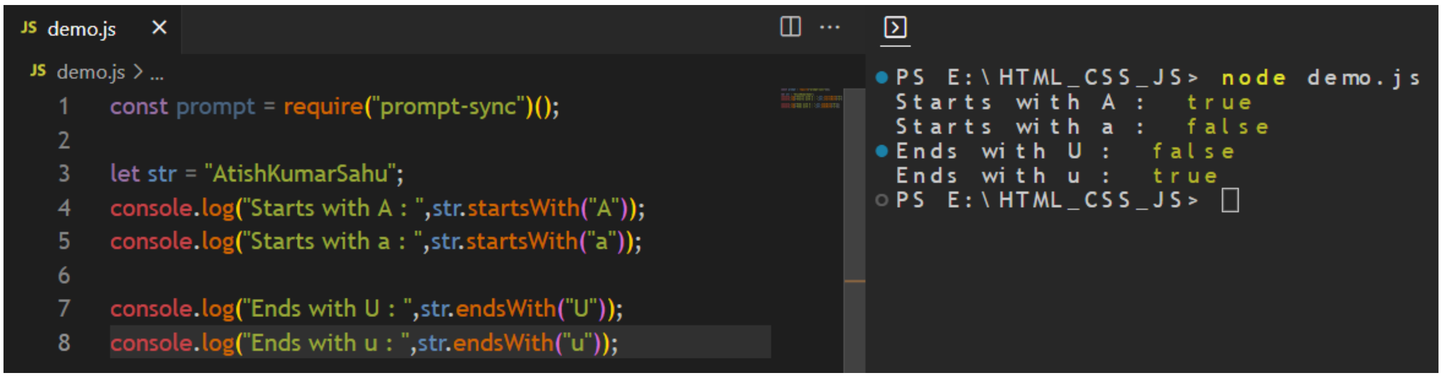
Constructor: the constructor property returns the function that created the String Prototype.



```javascript
const prompt = require("prompt-sync")();

let str = "HTML CSS JAVASCRIPT";
console.log(str.constructor);

let a = new String("Abc");
let b = String("Abc");
console.log(a);
console.log(b);

console.log(a === "Abc"); //false
console.log(b === "Abc"); //true

console.log(a instanceof String); //true;
console.log(b instanceof String); //false;

console.log(typeof a);
console.log(typeof b);
```

```
PS E:\HTML_CSS_JS> node demo.js
[Function: String]
[String: 'Abc']
Abc
false
true
true
false
object
string
PS E:\HTML_CSS_JS>
```

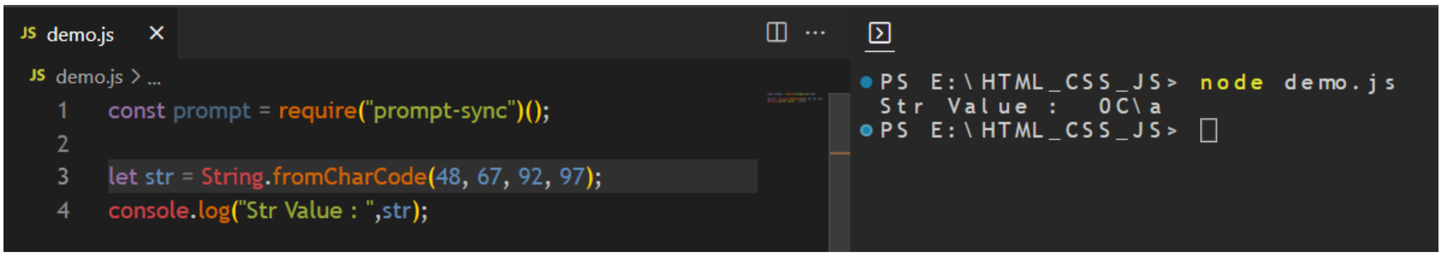
startsWith() & endsWith(): startsWith() function used for whether the String value is starts with the given value or not. endsWith() function used for whether the String is ends with the given value or not.



```javascript
const prompt = require("prompt-sync")();

let str = "AtishKumarSahu";
console.log("Starts with A : ",str.startsWith("A"));
console.log("Starts with a : ",str.startsWith("a"));

console.log("Ends with U : ",str.endsWith("U"));
console.log("Ends with u : ",str.endsWith("u"));
```

```
PS E:\HTML_CSS_JS> node demo.js
Starts with A :  true
Starts with a :  false
Ends with U :  false
Ends with u :  true
PS E:\HTML_CSS_JS>
```

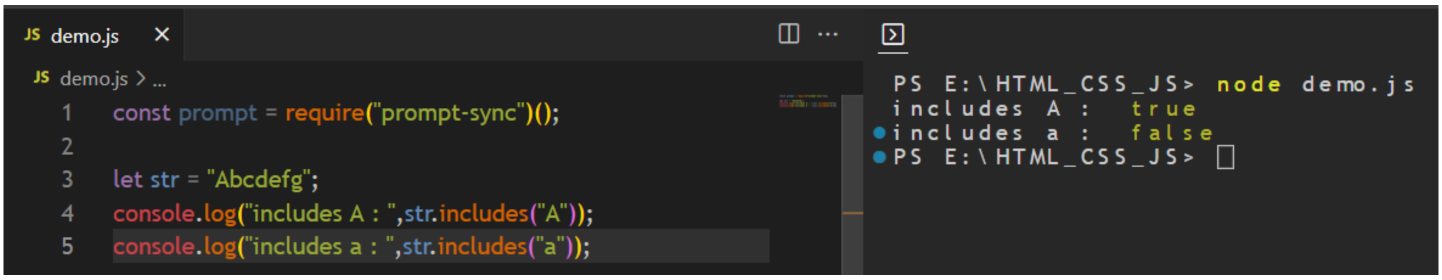
fromCharCode(): this function is used for to show the String value from the ascii code.



```js
const prompt = require("prompt-sync")();

let str = String.fromCharCode(48, 67, 92, 97);
console.log("Str Value : ",str);
```

```
PS E:\HTML_CSS_JS> node demo.js
Str Value :  0C\a
PS E:\HTML_CSS_JS>
```

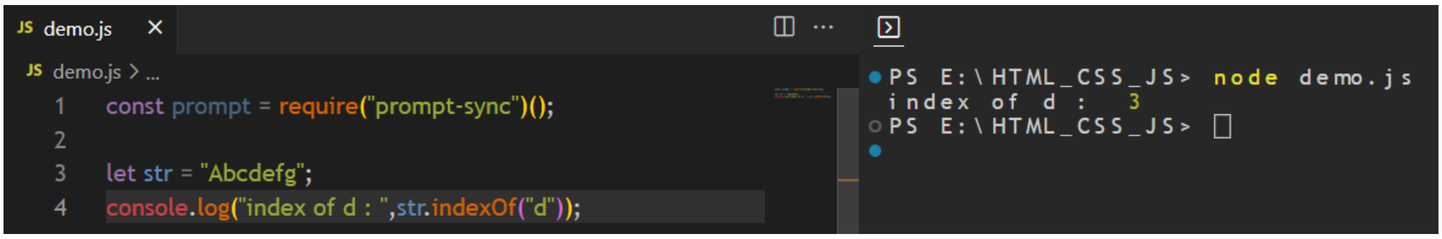
includes(): this function is used for to check whether the given character is inside or not.



```js
const prompt = require("prompt-sync")();

let str = "Abcdefg";
console.log("includes A : ",str.includes("A"));
console.log("includes a : ",str.includes("a"));
```

```
PS E:\HTML_CSS_JS> node demo.js
includes A :  true
includes a :  false
PS E:\HTML_CSS_JS>
```

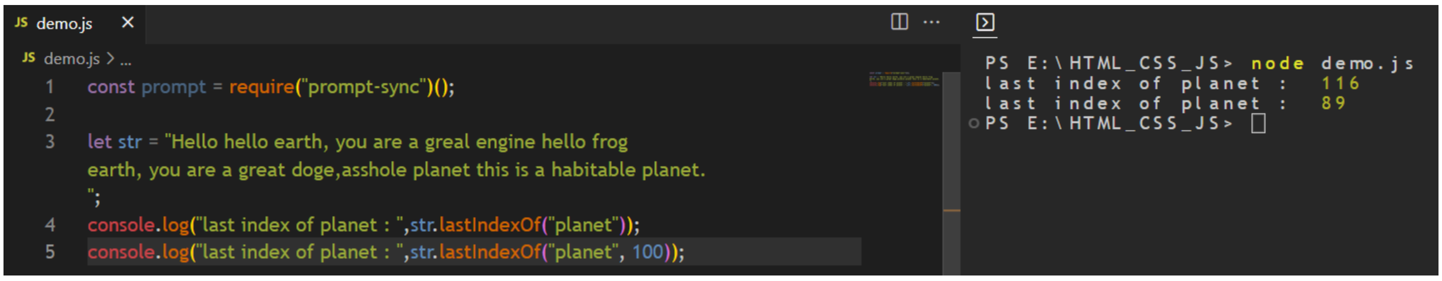
indexOf(): this function is used for to find the index value of given String value and count starts from 0.



```js
const prompt = require("prompt-sync")();

let str = "Abcdefg";
console.log("index of d : ",str.indexOf("d"));
```

```
PS E:\HTML_CSS_JS> node demo.js
index of d :  3
PS E:\HTML_CSS_JS>
```

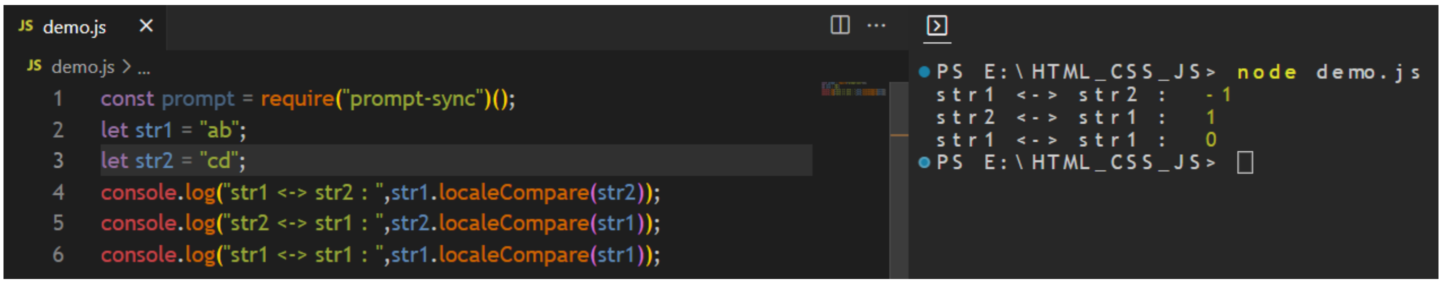
lastIndexOf(): this method used for to find the last index of the given String value.



```js
const prompt = require("prompt-sync")();

let str = "Hello hello earth, you are a greal engine hello frog
earth, you are a great doge,asshole planet this is a habitable planet.
";
console.log("last index of planet : ",str.lastIndexOf("planet"));
console.log("last index of planet : ",str.lastIndexOf("planet", 100));
```

```
PS E:\HTML_CSS_JS> node demo.js
last index of planet :  116
last index of planet :  89
PS E:\HTML_CSS_JS>
```

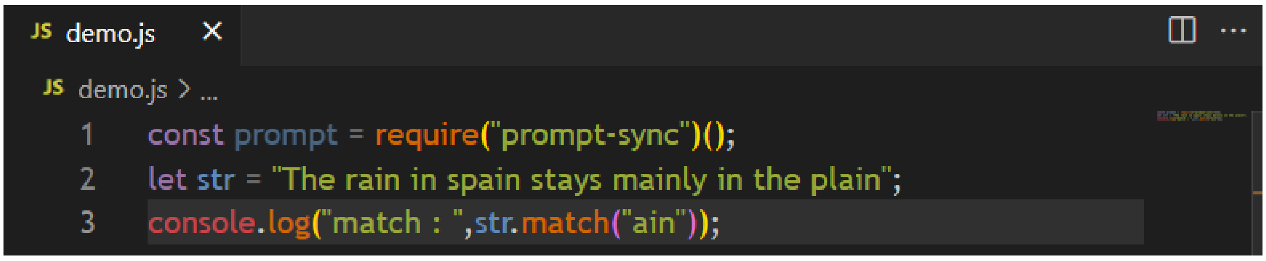
localeCompare(): this method compare two strings in the current locale. The localeCompare() method returns the sort order -1, 1 or 0 (for before, after or equal).



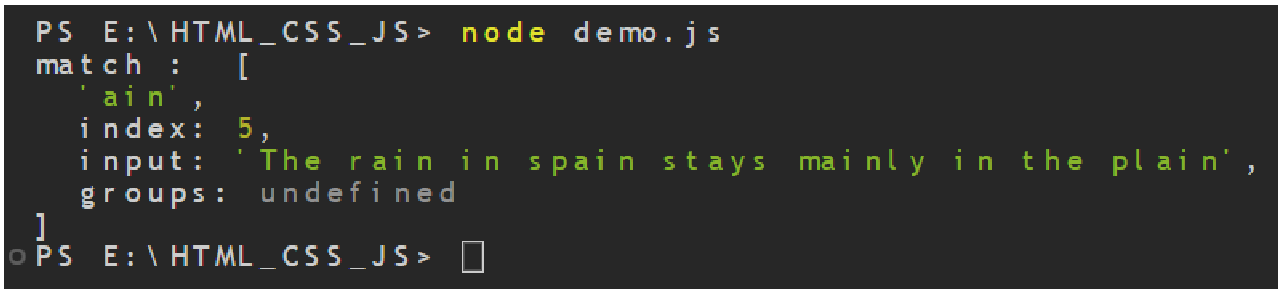
```js
const prompt = require("prompt-sync")();
let str1 = "ab";
let str2 = "cd";
console.log("str1 <-> str2 : ",str1.localeCompare(str2));
console.log("str2 <-> str1 : ",str2.localeCompare(str1));
console.log("str1 <-> str1 : ",str1.localeCompare(str1));
```

```
PS E:\HTML_CSS_JS> node demo.js
str1 <-> str2 :  -1
str2 <-> str1 :  1
str1 <-> str1 :  0
PS E:\HTML_CSS_JS>
```

match(): this function matches a string against a regular expression. The match() method returns an array with the matches. The match() method returns null if no match is found.

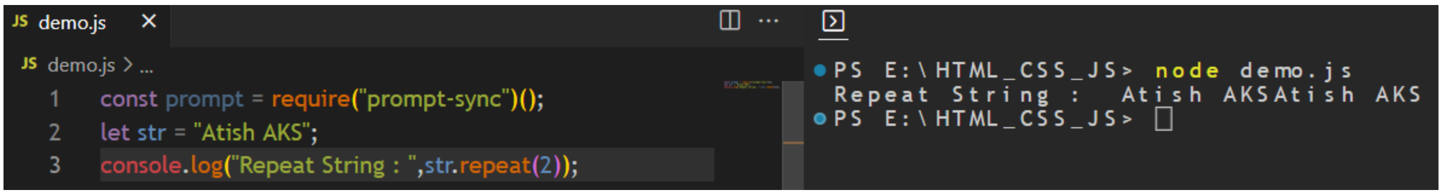


```
const prompt = require("prompt-sync")();
let str = "The rain in spain stays mainly in the plain";
console.log("match : ",str.match("ain"));
```



```
PS E:\HTML_CSS_JS> node demo.js
match :  [
  'ain',
  index: 5,
  input: 'The rain in spain stays mainly in the plain',
  groups: undefined
]
PS E:\HTML_CSS_JS> 
```

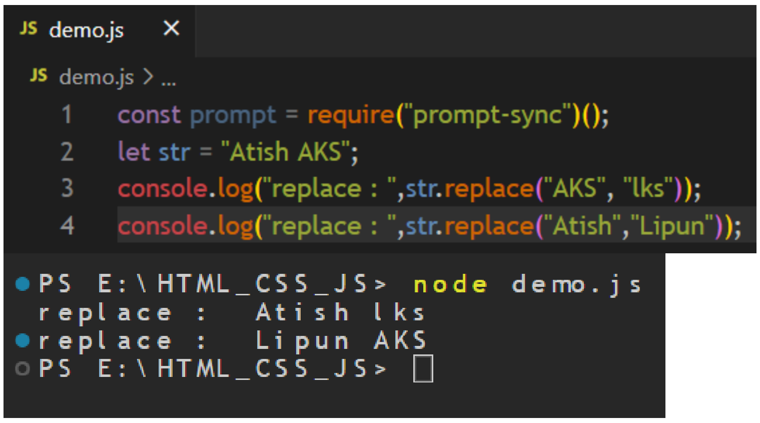
Repeat(): the repeat function returns a string with a number of copies of a string. the repeat() method returns a new String. the repeat() method does not change the original String.



```
const prompt = require("prompt-sync")();
let str = "Atish AKS";
console.log("Repeat String : ",str.repeat(2));
```

```
PS E:\HTML_CSS_JS> node demo.js
Repeat String :  Atish AKSAtish AKS
PS E:\HTML_CSS_JS> 
```

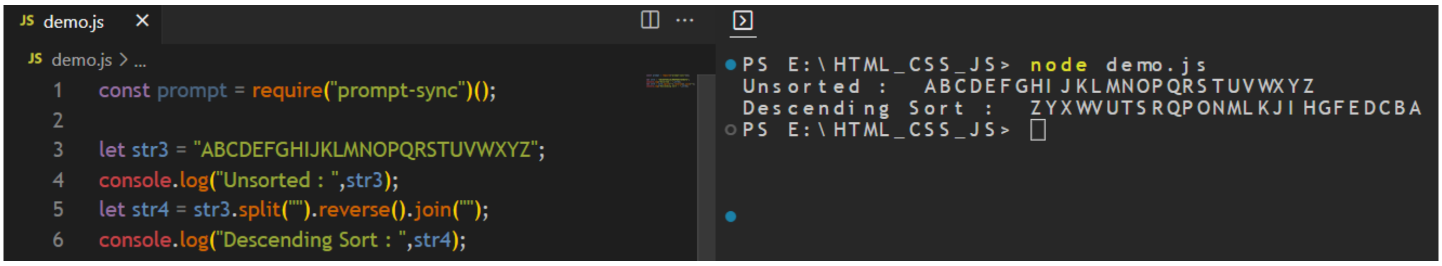
replace(): this method searches a string for a value or a regular expression. The replace() method returns a new string with the values(s) replaced. The replace() method does not change the original String.



```
const prompt = require("prompt-sync")();
let str = "Atish AKS";
console.log("replace : ",str.replace("AKS", "lks"));
console.log("replace : ",str.replace("Atish","Lipun"));
```

```
PS E:\HTML_CSS_JS> node demo.js
replace :  Atish lks
replace :  Lipun AKS
PS E:\HTML_CSS_JS> 
```

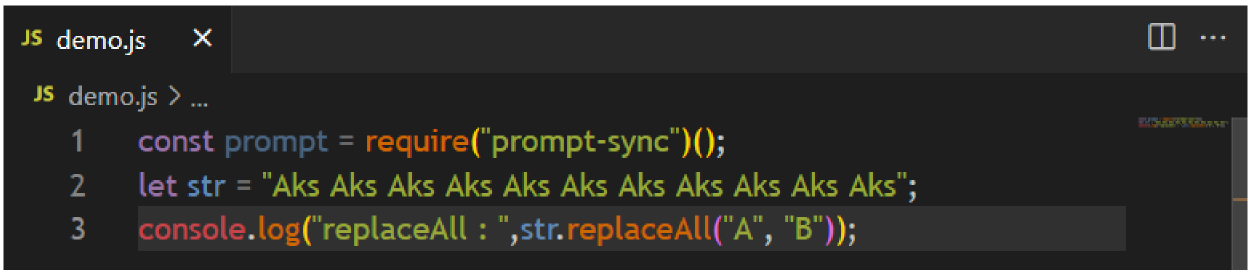
Reverse(): reverse a String.



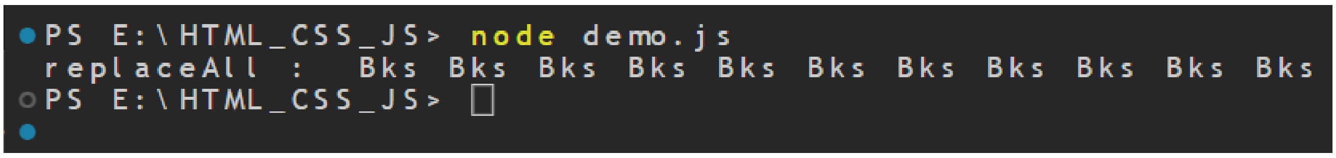
```
const prompt = require("prompt-sync")();

let str3 = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
console.log("Unsorted : ",str3);
let str4 = str3.split("").reverse().join("");
console.log("Descending Sort : ",str4);
```

```
PS E:\HTML_CSS_JS> node demo.js
Unsorted :  ABCDEFGHIJKLMNOPQRSTUVWXYZ
Descending Sort :  ZYXWVUTSRQPONMLKJIHGFEDCBA
PS E:\HTML_CSS_JS> 
```

replaceAll(): this function searches a string for a value or a regular expression. The replace all() method returns a new string with all values replaced. The replaceAll() method does not change the original string. the replaceAll() method was introduced in JavaScript 2021. The replaceAll() method does not work in internet explorer.

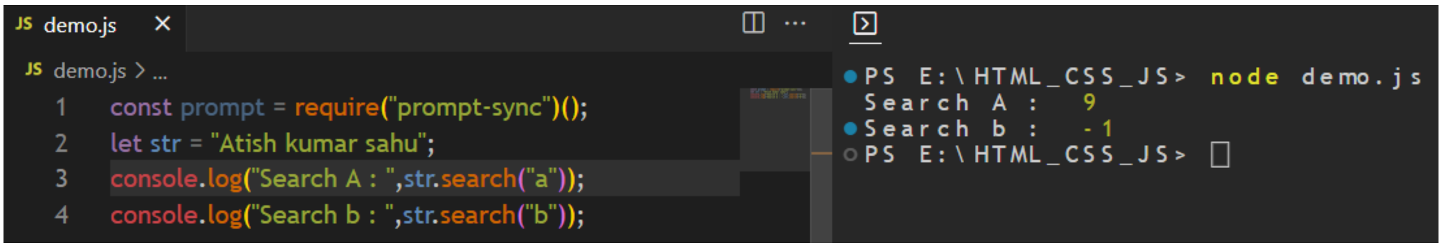


```
demo.js > ...
const prompt = require("prompt-sync")();
let str = "Aks Aks Aks Aks Aks Aks Aks Aks Aks Aks Aks";
console.log("replaceAll : ",str.replaceAll("A", "B"));
```



```
PS E:\HTML_CSS_JS> node demo.js
replaceAll :  Bks Bks Bks Bks Bks Bks Bks Bks Bks Bks Bks
PS E:\HTML_CSS_JS>
```

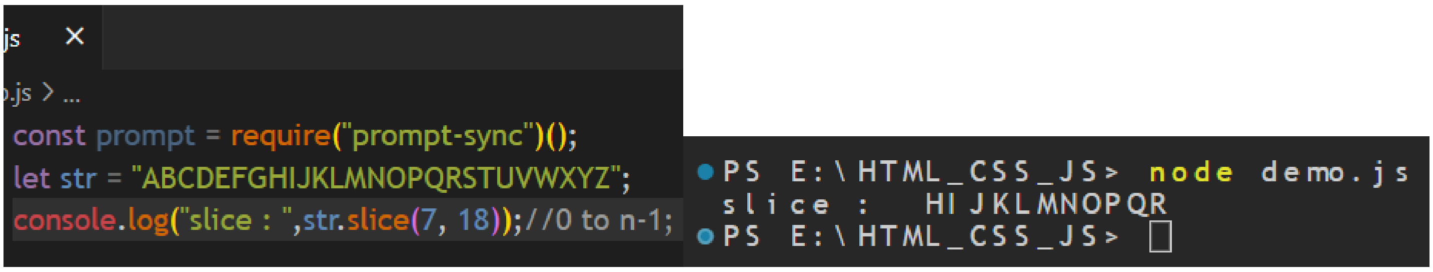
Search(): this method matches a string against a regular expression. The search() method returns the index position of the first match. the search() method returns -1 if no match is found. The search() method is case sensitive.



```
demo.js > ...
const prompt = require("prompt-sync")();
let str = "Atish kumar sahu";
console.log("Search A : ",str.search("a"));
console.log("Search b : ",str.search("b"));
```

```
PS E:\HTML_CSS_JS> node demo.js
Search A :  9
Search b :  -1
PS E:\HTML_CSS_JS>
```

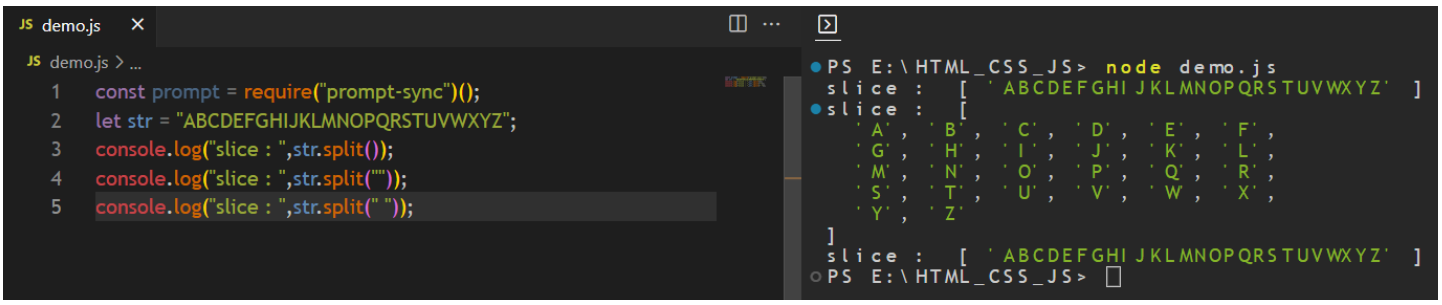
slice(): this method extract a part of a string. the slice() method returns the extracted part in a new string. the slice() method does not change the original string. the start and end parameters specifies the part of the string to extract. The first position is 0 the second is 1, A negative number selects from the end of the string.



```
js
.js > ...
const prompt = require("prompt-sync")();
let str = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
console.log("slice : ",str.slice(7, 18));//0 to n-1;
```

```
PS E:\HTML_CSS_JS> node demo.js
slice :  HIJKLMNOPQR
PS E:\HTML_CSS_JS>
```

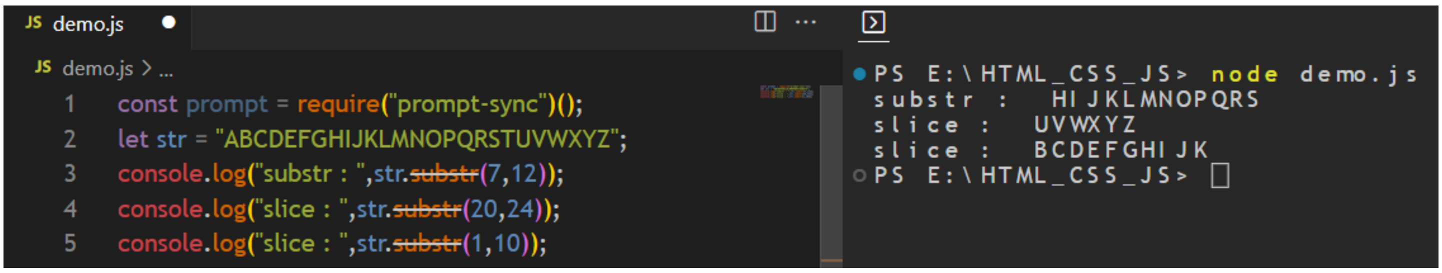
Split(): this function splits a string into an array of substrings. The split() method returns the new array. the split() method does not change the original string. if(" ") is used as separator, the string is split between words.



```
demo.js > ...
const prompt = require("prompt-sync")();
let str = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
console.log("slice : ",str.split());
console.log("slice : ",str.split(""));
console.log("slice : ",str.split(" "));
```

```
PS E:\HTML_CSS_JS> node demo.js
slice :  [ 'ABCDEFGHIJKLMNOPQRSTUVWXYZ' ]
slice :  [
  'A', 'B', 'C', 'D', 'E', 'F',
  'G', 'H', 'I', 'J', 'K', 'L',
  'M', 'N', 'O', 'P', 'Q', 'R',
  'S', 'T', 'U', 'V', 'W', 'X',
  'Y', 'Z'
]
slice :  [ 'ABCDEFGHIJKLMNOPQRSTUVWXYZ' ]
PS E:\HTML_CSS_JS>
```

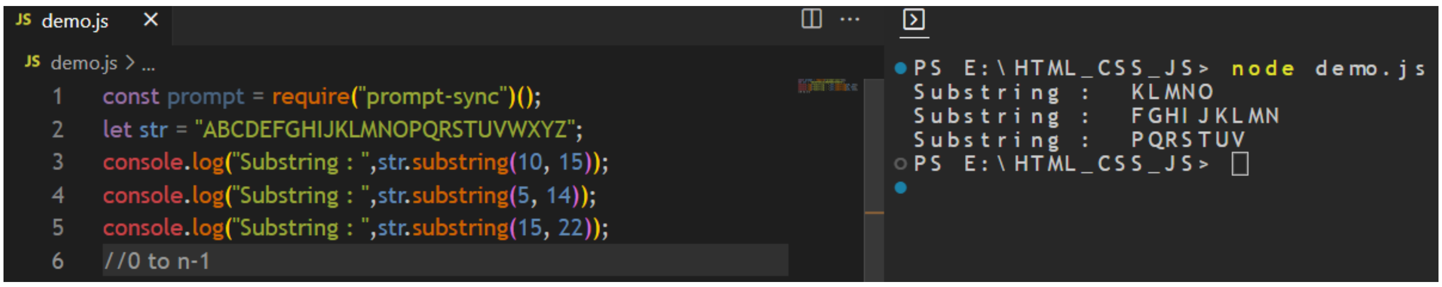
substr(): this method extracts a part of a string. this function begins at a specified position, and returns a specified number of characters. The function does not change the original string. to extract characters from the end of the string use a negative start position.



```
const prompt = require("prompt-sync")();
let str = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
console.log("substr : ",str.substr(7,12));
console.log("slice : ",str.substr(20,24));
console.log("slice : ",str.substr(1,10));
```

```
PS E:\HTML_CSS_JS> node demo.js
substr :  HIJKLMNOPQRS
slice :  UVWXYZ
slice :  BCDEFGHIJK
PS E:\HTML_CSS_JS>
```

Substring(): this method extracts character between two indices position from a string and return the substring. The substring() method extracts characters from start to end. The method does not change the original string. if start is greater than end, arguments are swapped: (4, 1) = (1, 4) starts of end values less than 0 are treated as 0.

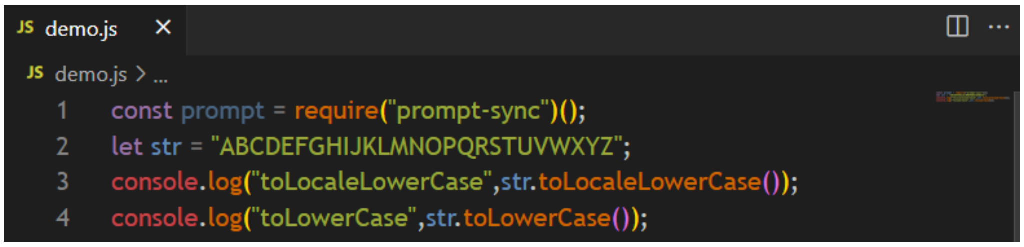


```
const prompt = require("prompt-sync")();
let str = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
console.log("Substring : ",str.substring(10, 15));
console.log("Substring : ",str.substring(5, 14));
console.log("Substring : ",str.substring(15, 22));
//0 to n-1
```

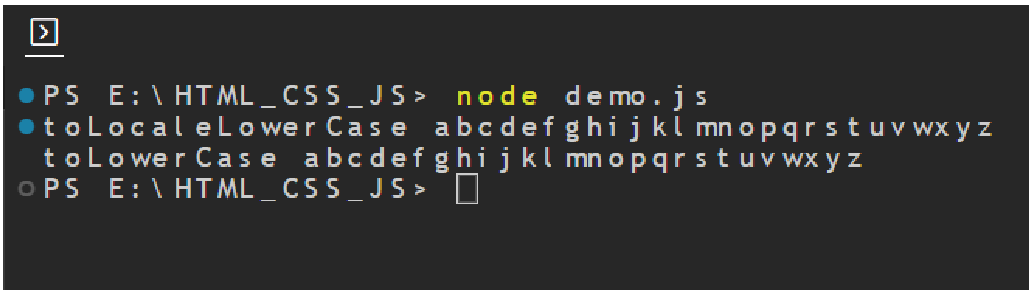
```
PS E:\HTML_CSS_JS> node demo.js
Substring :  KLMNO
Substring :  FGHIJKLMN
Substring :  PQRSTUV
PS E:\HTML_CSS_JS>
```

toLocalLowerCase(): this method converts a string to lowercase letters, using current locale. The locale is based on the language settings of the browser. The toLocalLowerCase() method does not change the original string. the toLocalLowerCase() returns the same result as toLowerCase(), except for locales that conflict with the regular Unicode case mapping such as Turkish.

toLowerCase(): this method converts a string to lower case letters. This function does not change the original string.



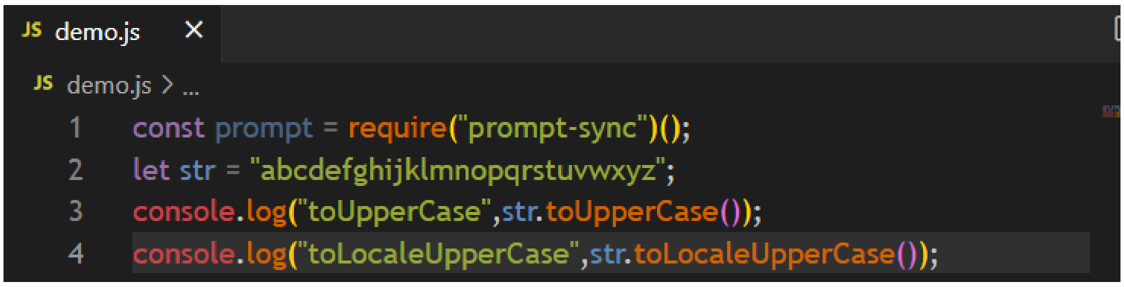
```
const prompt = require("prompt-sync")();
let str = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
console.log("toLocaleLowerCase",str.toLocaleLowerCase());
console.log("toLowerCase",str.toLowerCase());
```



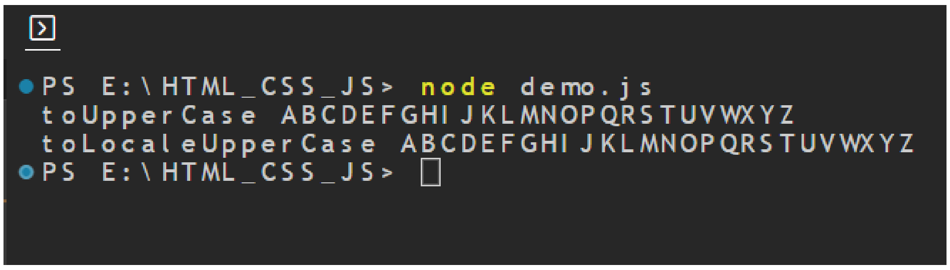
```
PS E:\HTML_CSS_JS> node demo.js
toLocaleLowerCase abcdefghijklmnopqrstuvwxyz
toLowerCase abcdefghijklmnopqrstuvwxyz
PS E:\HTML_CSS_JS>
```

toLocalUpperCase(): this method converts a string to uppercase letters, using current locale. The locale is based on the language settings of the browser. The toLocalUpperCase() method doesn't change the original string. the toLocalUpperCase() returns the same result as toUpperCase(). Except for locales that conflict with the regular Unicode case mappings like turkish.

toUpperCase(): this method converts a string to uppercase letters. The toUpperCase method does not change the orginal string.

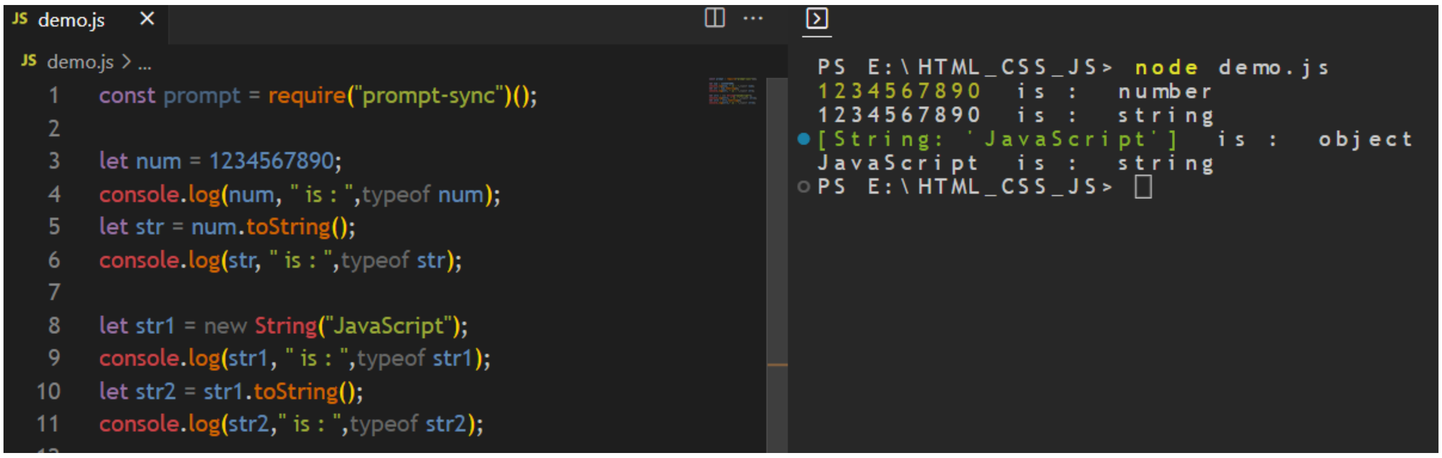


```
JS demo.js > ...
const prompt = require("prompt-sync")();
let str = "abcdefghijklmnopqrstuvwxyz";
console.log("toUpperCase",str.toUpperCase());
console.log("toLocaleUpperCase",str.toLocaleUpperCase());
```



```
PS E:\HTML_CSS_JS> node demo.js
toUpperCase ABCDEFGHIJKLMNOPQRSTUVWXYZ
toLocaleUpperCase ABCDEFGHIJKLMNOPQRSTUVWXYZ
PS E:\HTML_CSS_JS>
```

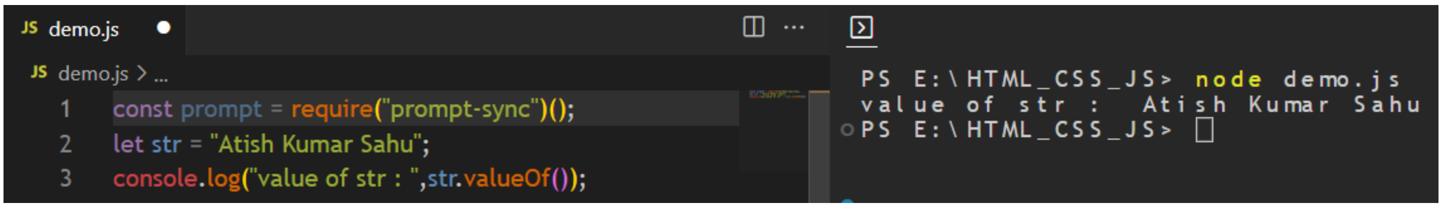
toString(): this method returns a string as a string. the toString() method does not change the original string. the method can be used to convert a String Object into a String.



```
JS demo.js > ...
const prompt = require("prompt-sync")();

let num = 1234567890;
console.log(num, " is : ",typeof num);
let str = num.toString();
console.log(str, " is : ",typeof str);

let str1 = new String("JavaScript");
console.log(str1, " is : ",typeof str1);
let str2 = str1.toString();
console.log(str2," is : ",typeof str2);
```

```
PS E:\HTML_CSS_JS> node demo.js
1234567890  is :  number
1234567890  is :  string
[String: 'JavaScript']  is :  object
JavaScript  is :  string
PS E:\HTML_CSS_JS>
```

valueOf(): this method returns the primitive value of a String. the valueOf() method doesn't does not change the original string. this method can be used to convert a string object into a String.



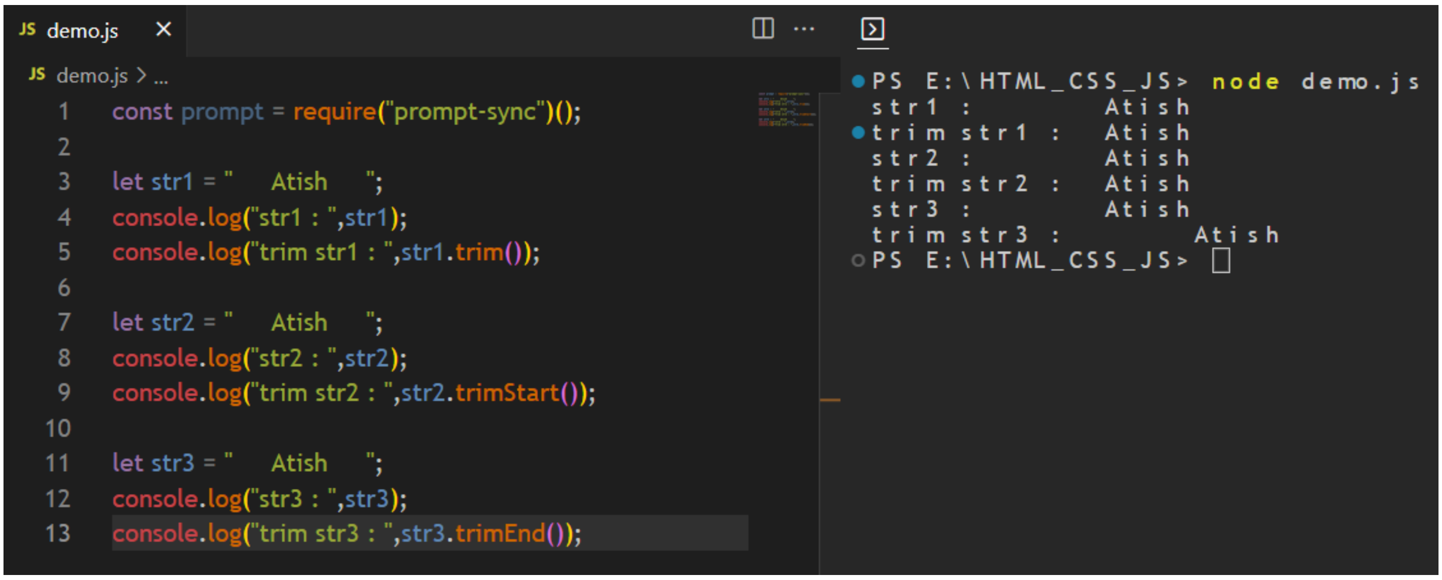
```
JS demo.js > ...
const prompt = require("prompt-sync")();
let str = "Atish Kumar Sahu";
console.log("value of str : ",str.valueOf());
```

```
PS E:\HTML_CSS_JS> node demo.js
value of str :  Atish Kumar Sahu
PS E:\HTML_CSS_JS>
```

trim(): this method removes whitespace from both sides of a string. this function does not change the original string.

trimEnd(): this method removes whitespace from the end of a String. the trimEnd() method does not change the original String the trimEnd() method works like trim(), but removes whitespace only from the end of a string.

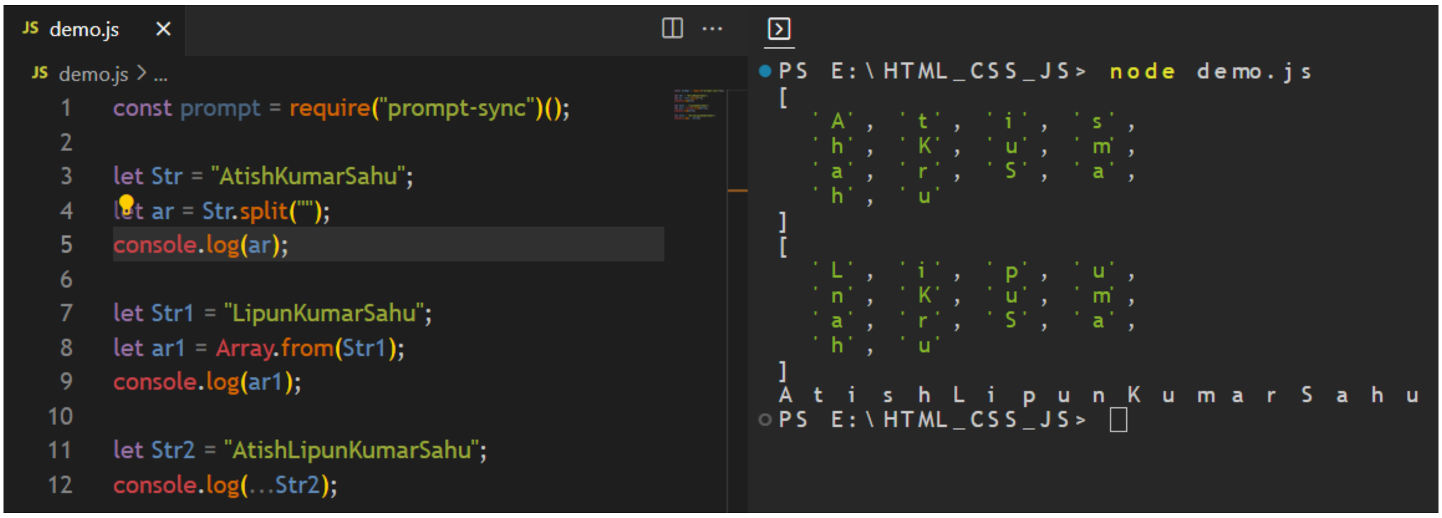
trimStart(): this method removes whitespace from the beginning of a String. the trimStart() function doesn't change the original String. this method works like trim() but removes whitespace only from the Start of a string.



```javascript
const prompt = require("prompt-sync")();

let str1 = "     Atish     ";
console.log("str1 : ",str1);
console.log("trim str1 : ",str1.trim());

let str2 = "     Atish     ";
console.log("str2 : ",str2);
console.log("trim str2 : ",str2.trimStart());

let str3 = "     Atish     ";
console.log("str3 : ",str3);
console.log("trim str3 : ",str3.trimEnd());
```

```
PS E:\HTML_CSS_JS> node demo.js
str1 :       Atish
trim str1 :  Atish
str2 :       Atish
trim str2 :  Atish
str3 :       Atish
trim str3 :         Atish
PS E:\HTML_CSS_JS>
```

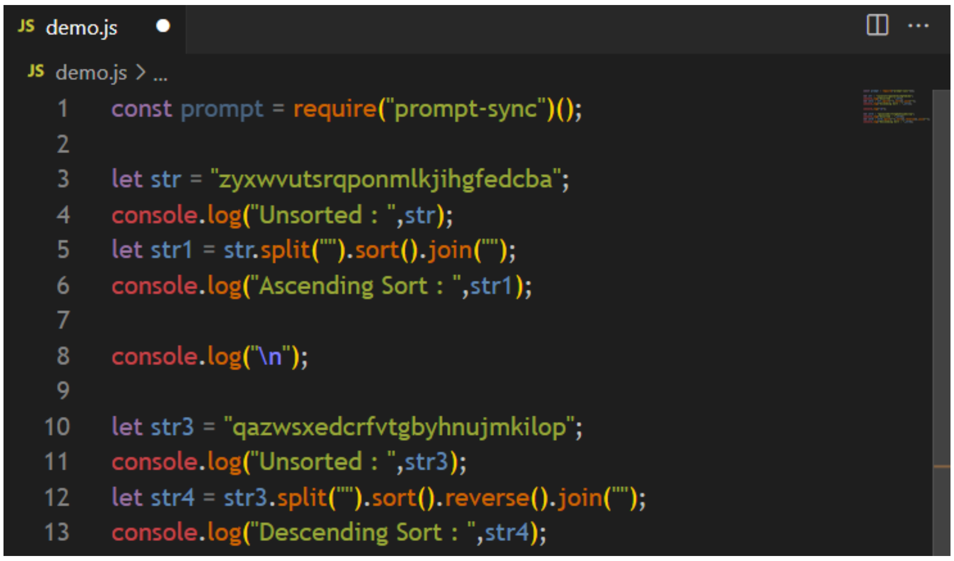
<u>String to Array Conversion:</u>



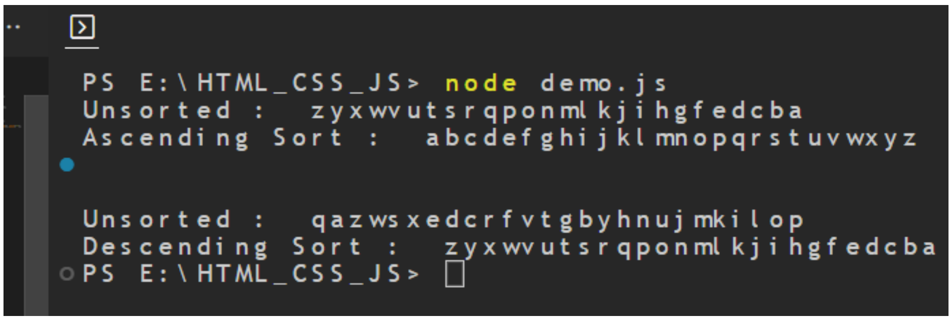
```javascript
const prompt = require("prompt-sync")();

let Str = "AtishKumarSahu";
let ar = Str.split("");
console.log(ar);

let Str1 = "LipunKumarSahu";
let ar1 = Array.from(Str1);
console.log(ar1);

let Str2 = "AtishLipunKumarSahu";
console.log(...Str2);
```

```
PS E:\HTML_CSS_JS> node demo.js
[
  'A', 't', 'i', 's',
  'h', 'K', 'u', 'm',
  'a', 'r', 'S', 'a',
  'h', 'u'
]
[
  'L', 'i', 'p', 'u',
  'n', 'K', 'u', 'm',
  'a', 'r', 'S', 'a',
  'h', 'u'
]
A t i s h L i p u n K u m a r S a h u
PS E:\HTML_CSS_JS>
```

String Sorting:

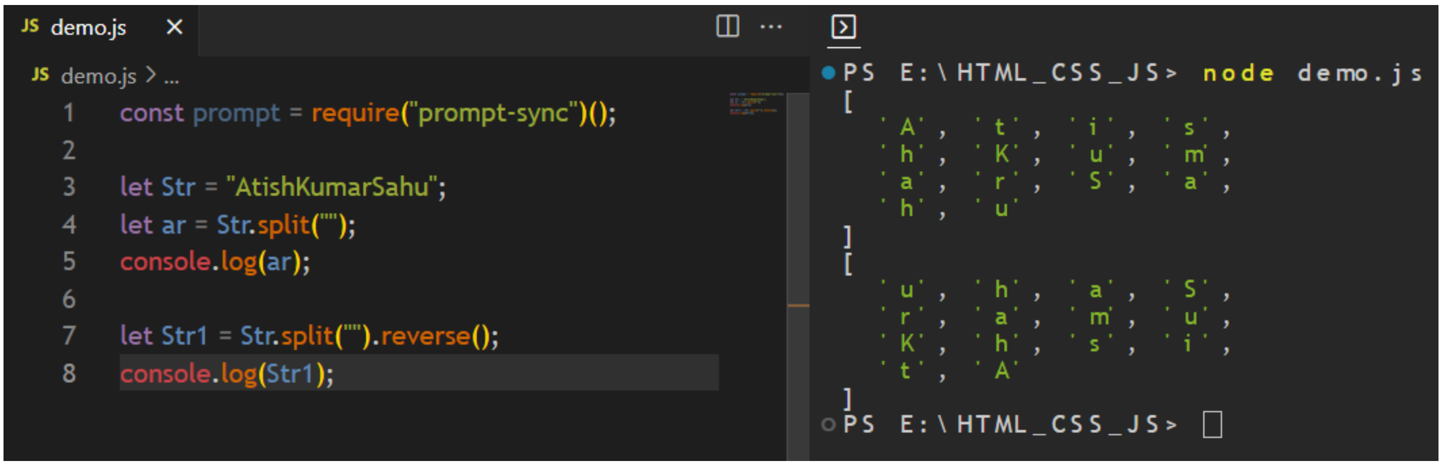


```javascript
const prompt = require("prompt-sync")();

let str = "zyxwvutsrqponmlkjihgfedcba";
console.log("Unsorted : ",str);
let str1 = str.split("").sort().join("");
console.log("Ascending Sort : ",str1);

console.log("\n");

let str3 = "qazwsxedcrfvtgbyhnujmkilop";
console.log("Unsorted : ",str3);
let str4 = str3.split("").sort().reverse().join("");
console.log("Descending Sort : ",str4);
```



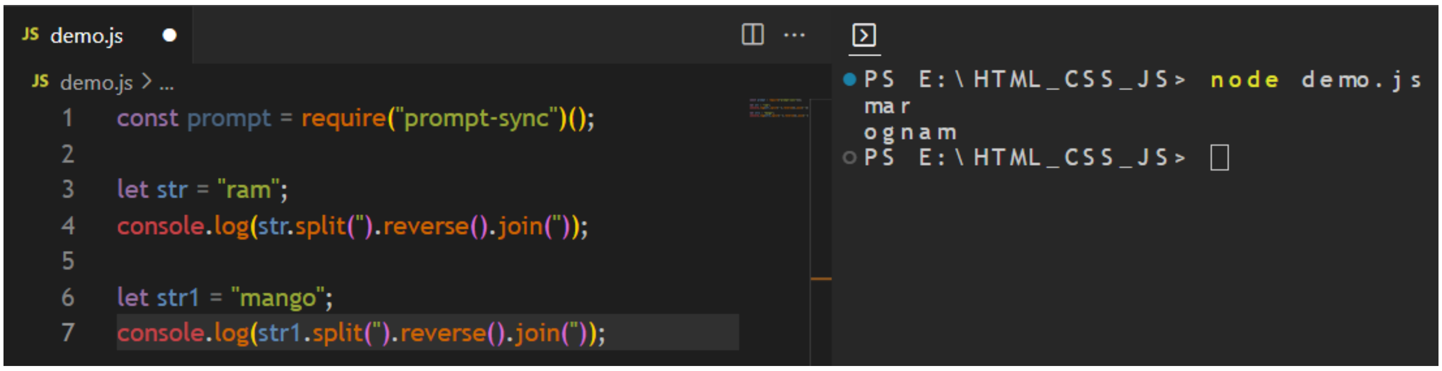
```
PS E:\HTML_CSS_JS> node demo.js
Unsorted :  zyxwvutsrqponmlkjihgfedcba
Ascending Sort :  abcdefghijklmnopqrstuvwxyz


Unsorted :  qazwsxedcrfvtgbyhnujmkilop
Descending Sort :  zyxwvutsrqponmlkjihgfedcba
PS E:\HTML_CSS_JS> 
```

String Reverse:



```javascript
const prompt = require("prompt-sync")();

let Str = "AtishKumarSahu";
let ar = Str.split("");
console.log(ar);

let Str1 = Str.split("").reverse();
console.log(Str1);
```

```
PS E:\HTML_CSS_JS> node demo.js
[
  'A', 't', 'i', 's',
  'h', 'K', 'u', 'm',
  'a', 'r', 'S', 'a',
  'h', 'u'
]
[
  'u', 'h', 'a', 'S',
  'r', 'a', 'm', 'u',
  'K', 'h', 's', 'i',
  't', 'A'
]
PS E:\HTML_CSS_JS> 
```

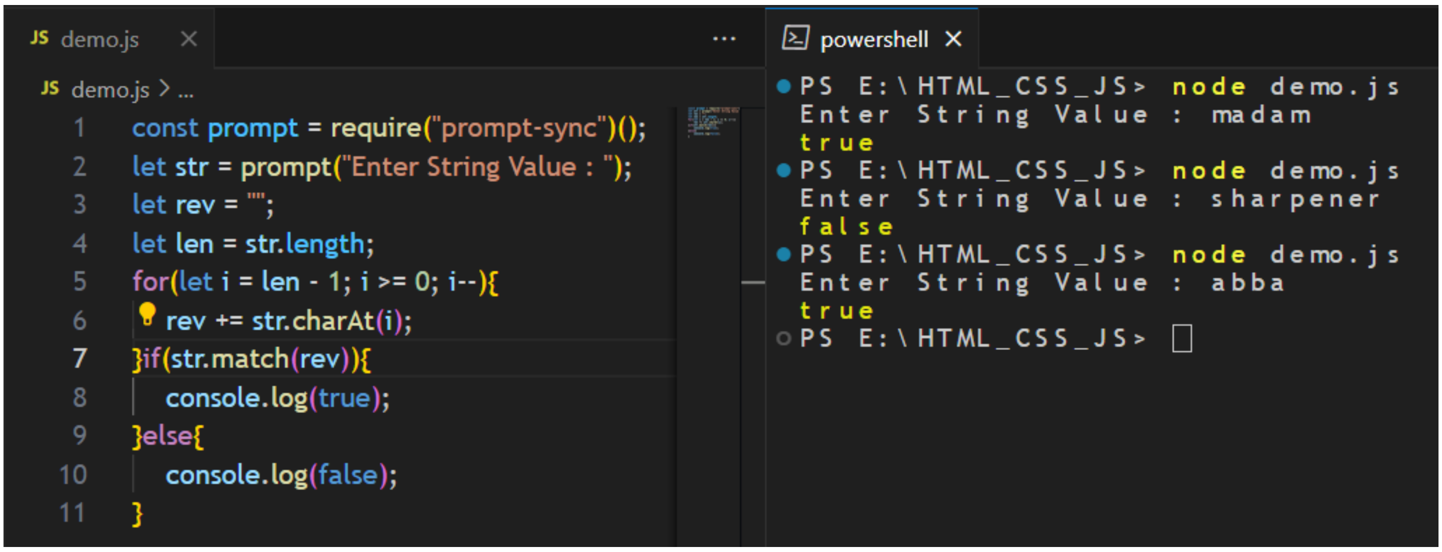


```javascript
const prompt = require("prompt-sync")();

let str = "ram";
console.log(str.split('').reverse().join(''));

let str1 = "mango";
console.log(str1.split('').reverse().join(''));
```

```
PS E:\HTML_CSS_JS> node demo.js
mar
ognam
PS E:\HTML_CSS_JS> 
```

String Question 01:

Given A String check whether on reversal it is the same or not. Return True if yes otherwise return False. Input:- word = "madam" Output : true



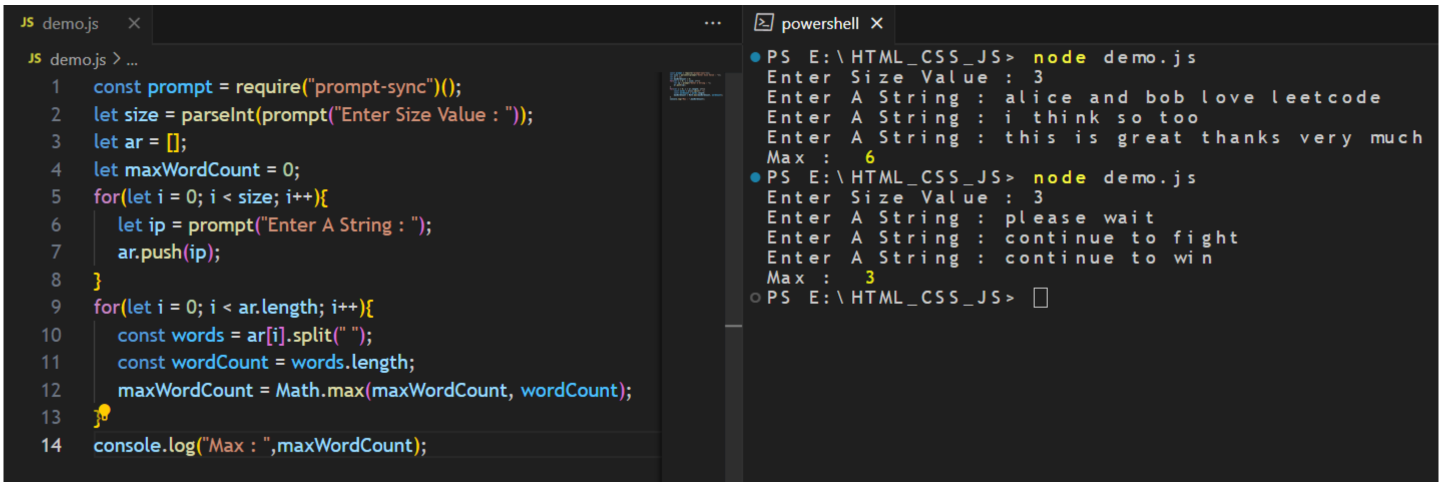
```javascript
const prompt = require("prompt-sync")();
let str = prompt("Enter String Value : ");
let rev = "";
let len = str.length;
for(let i = len - 1; i >= 0; i--){
    rev += str.charAt(i);
}if(str.match(rev)){
    console.log(true);
}else{
    console.log(false);
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : madam
true
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : sharpener
false
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : abba
true
PS E:\HTML_CSS_JS>
```

String Question 02:

A sentence is a list of words that are separated by a single space with no leading or trailing spaces. You are given an array of strings sentences, where each sentences[i] represents a single sentence. Return the maximum number of words that appear in a single sentence.

Input: [“alice and bob love leetcode”, “I think so too”, “this is great thanks very much”]
Output: 6

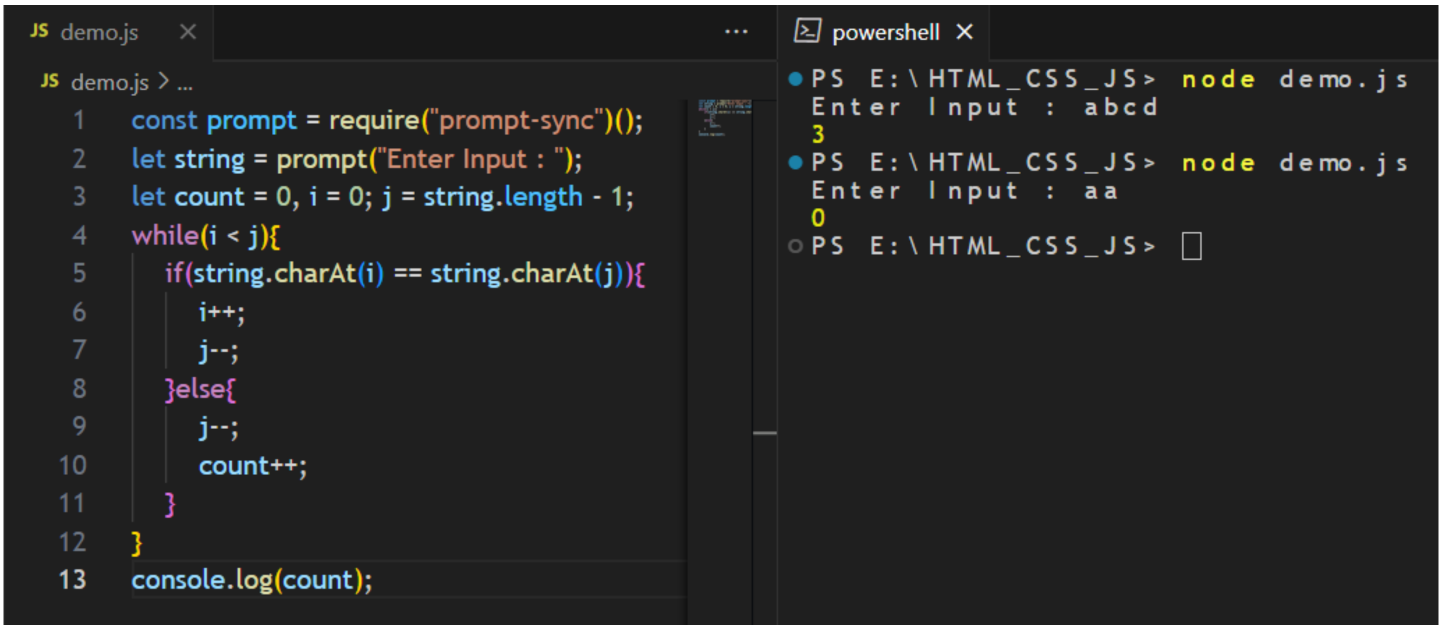


```javascript
const prompt = require("prompt-sync")();
let size = parseInt(prompt("Enter Size Value : "));
let ar = [];
let maxWordCount = 0;
for(let i = 0; i < size; i++){
    let ip = prompt("Enter A String : ");
    ar.push(ip);
}
for(let i = 0; i < ar.length; i++){
    const words = ar[i].split(" ");
    const wordCount = words.length;
    maxWordCount = Math.max(maxWordCount, wordCount);
}
console.log("Max : ",maxWordCount);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Size Value : 3
Enter A String : alice and bob love leetcode
Enter A String : i think so too
Enter A String : this is great thanks very much
Max :  6
PS E:\HTML_CSS_JS> node demo.js
Enter Size Value : 3
Enter A String : please wait
Enter A String : continue to fight
Enter A String : continue to win
Max :  3
PS E:\HTML_CSS_JS>
```

String Question 03:

You have given a string , You have to add characters at start of string to make it a palindrome . return the minimum number of characters required to add to make it a palindrome. Input: abcd Output: 3 Input: aa Output: 0



demo.js

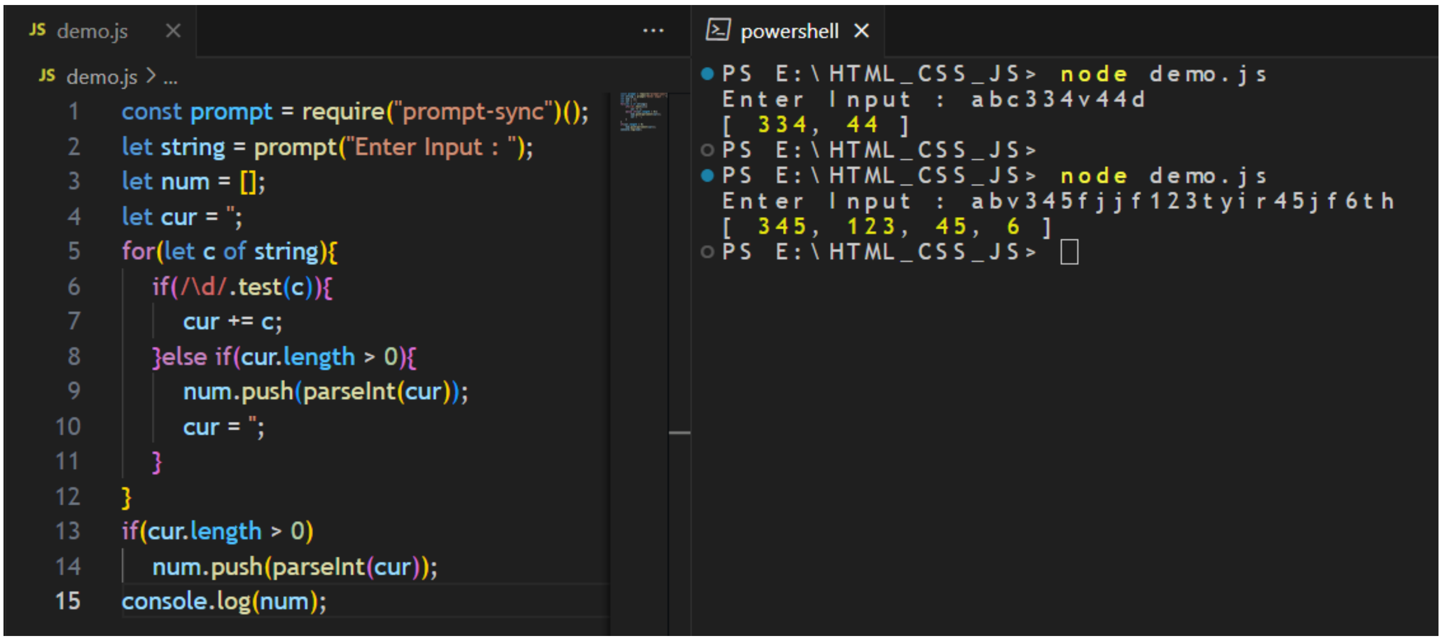
```javascript
const prompt = require("prompt-sync")();
let string = prompt("Enter Input : ");
let count = 0, i = 0; j = string.length - 1;
while(i < j){
    if(string.charAt(i) == string.charAt(j)){
        i++;
        j--;
    }else{
        j--;
        count++;
    }
}
console.log(count);
```

powershell

```
PS E:\HTML_CSS_JS> node demo.js
Enter Input : abcd
3
PS E:\HTML_CSS_JS> node demo.js
Enter Input : aa
0
PS E:\HTML_CSS_JS>
```

String Question 04:

Given a String Extract all numbers from it and store it inside an array. Return the Array Once extraction is completed. Note that if the string is "abc334vf" then the number is 334 and not 3,3,4 as Input: abc334v44d Output: [334, 44];



demo.js

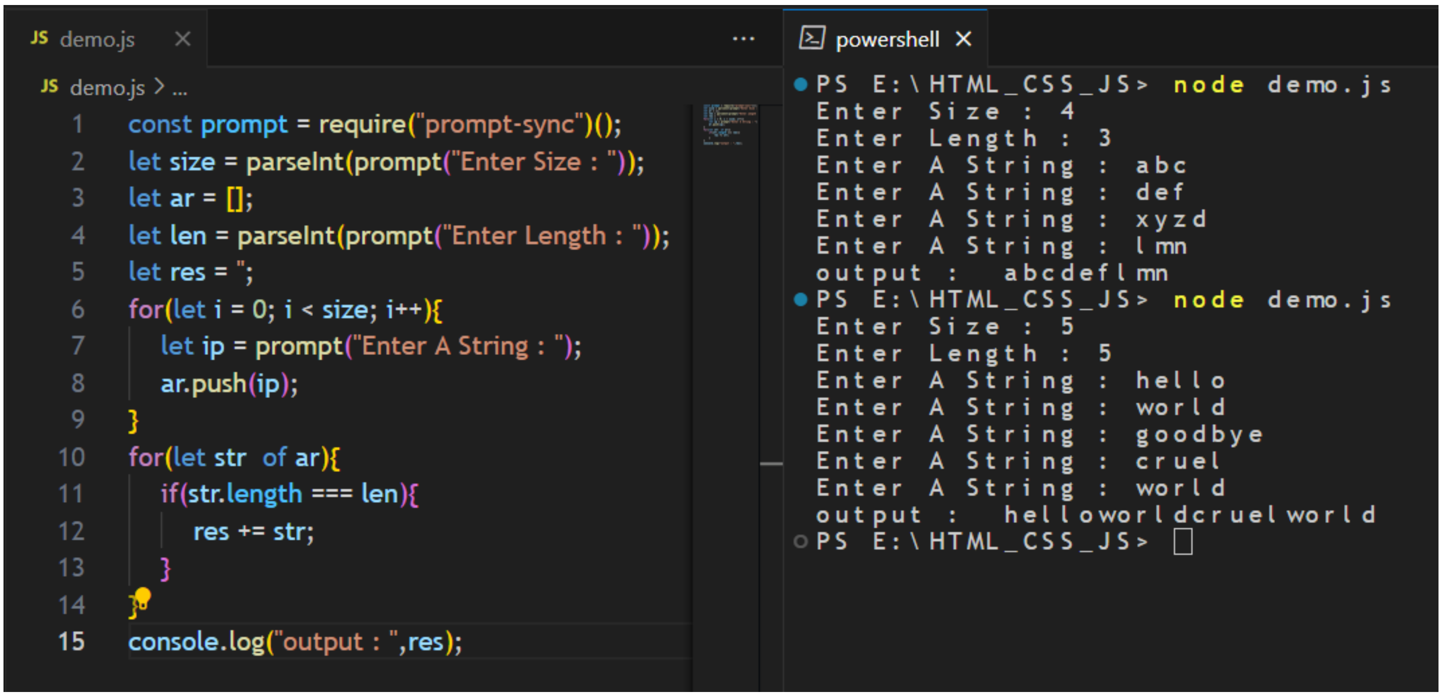
```javascript
const prompt = require("prompt-sync")();
let string = prompt("Enter Input : ");
let num = [];
let cur = '';
for(let c of string){
    if(/\d/.test(c)){
        cur += c;
    }else if(cur.length > 0){
        num.push(parseInt(cur));
        cur = '';
    }
}
if(cur.length > 0)
    num.push(parseInt(cur));
console.log(num);
```

powershell

```
PS E:\HTML_CSS_JS> node demo.js
Enter Input : abc334v44d
[ 334, 44 ]
PS E:\HTML_CSS_JS>
PS E:\HTML_CSS_JS> node demo.js
Enter Input : abv345fjjf123tyir45jf6th
[ 345, 123, 45, 6 ]
PS E:\HTML_CSS_JS>
```

String Question 05:

Given an array of strings and an integer len . Concat the string such whose length is equal to len.   Input: [abc, def, xyzd, lmn]   Output: abcdeflmn



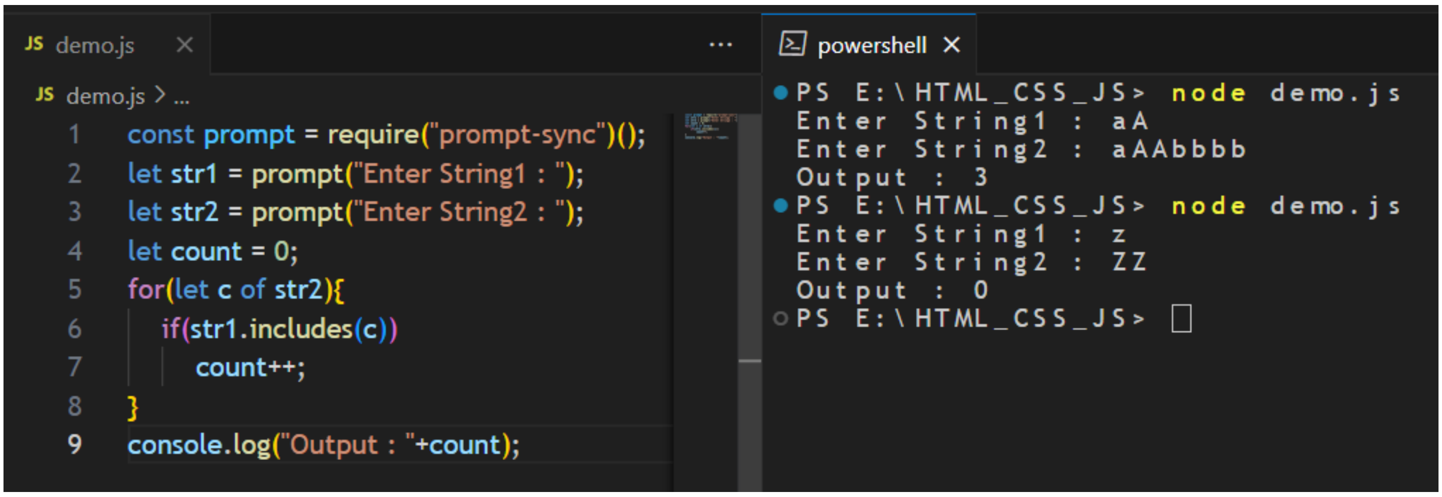
```javascript
const prompt = require("prompt-sync")();
let size = parseInt(prompt("Enter Size : "));
let ar = [];
let len = parseInt(prompt("Enter Length : "));
let res = '';
for(let i = 0; i < size; i++){
    let ip = prompt("Enter A String : ");
    ar.push(ip);
}
for(let str  of ar){
    if(str.length === len){
        res += str;
    }
}
console.log("output : ",res);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Size : 4
Enter Length : 3
Enter A String : abc
Enter A String : def
Enter A String : xyzd
Enter A String : lmn
output :  abcdeflmn
PS E:\HTML_CSS_JS> node demo.js
Enter Size : 5
Enter Length : 5
Enter A String : hello
Enter A String : world
Enter A String : goodbye
Enter A String : cruel
Enter A String : world
output :  helloworldcruelworld
PS E:\HTML_CSS_JS>
```

String Question 06:

You're given strings jewels representing the types of stones that are jewels, and stones representing the stones you have. Each character in stones is a type of stone you have. You want to know how many of the stones you have are also jewels. Letters are case sensitive, so "a" is considered a different type of stone from "A".

Input: aA, aAAbbbb   Output: 3   Input: z, ZZ   Output: 0

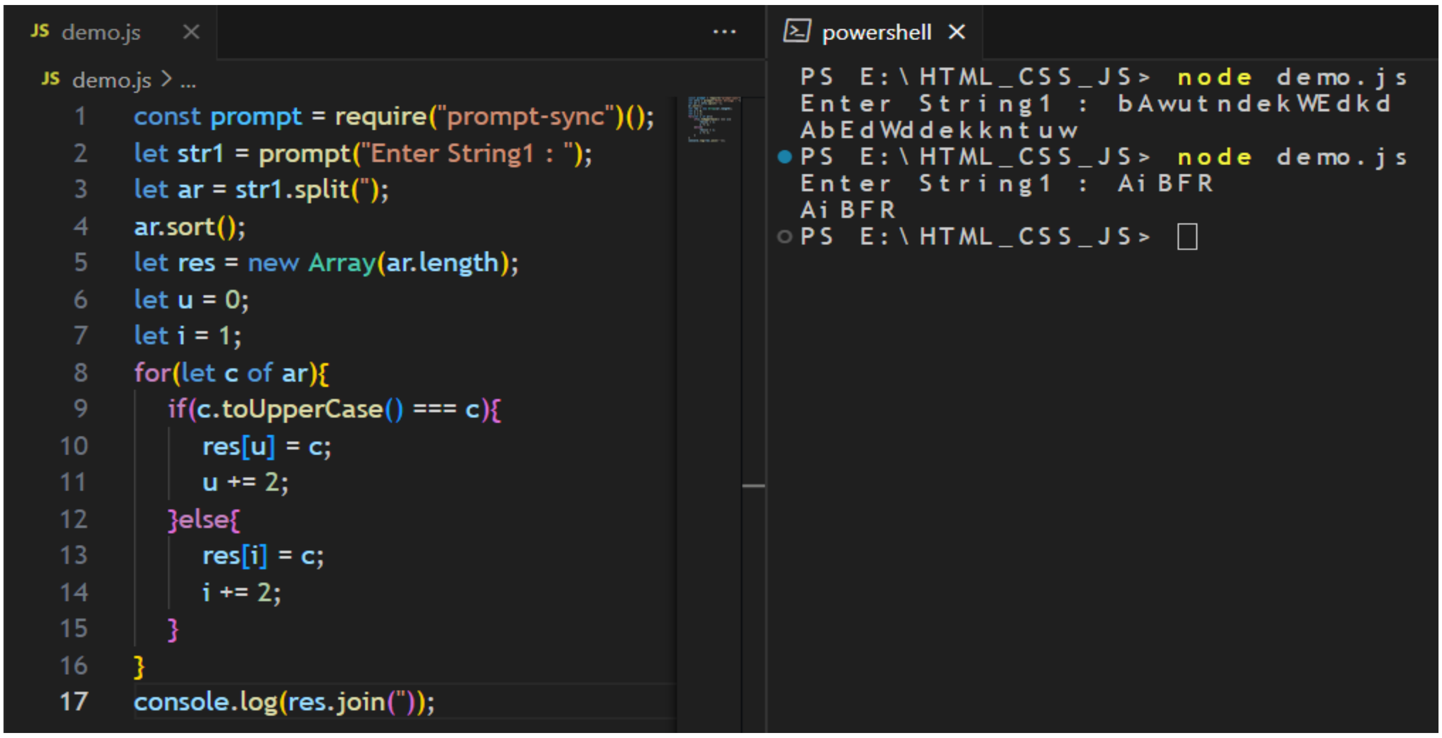


```javascript
const prompt = require("prompt-sync")();
let str1 = prompt("Enter String1 : ");
let str2 = prompt("Enter String2 : ");
let count = 0;
for(let c of str2){
    if(str1.includes(c))
        count++;
}
console.log("Output : "+count);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String1 : aA
Enter String2 : aAAbbbb
Output : 3
PS E:\HTML_CSS_JS> node demo.js
Enter String1 : z
Enter String2 : ZZ
Output : 0
PS E:\HTML_CSS_JS>
```

String Question 07:

A string is said to be pro sorted if it is sorted and alternate uppercase and lowercase letters are there . You have given a string with uppercase and lowercase characters in it . you have to make it a pro sorted string.   Ip: AiBFR   Output: AiBFR


demo.js

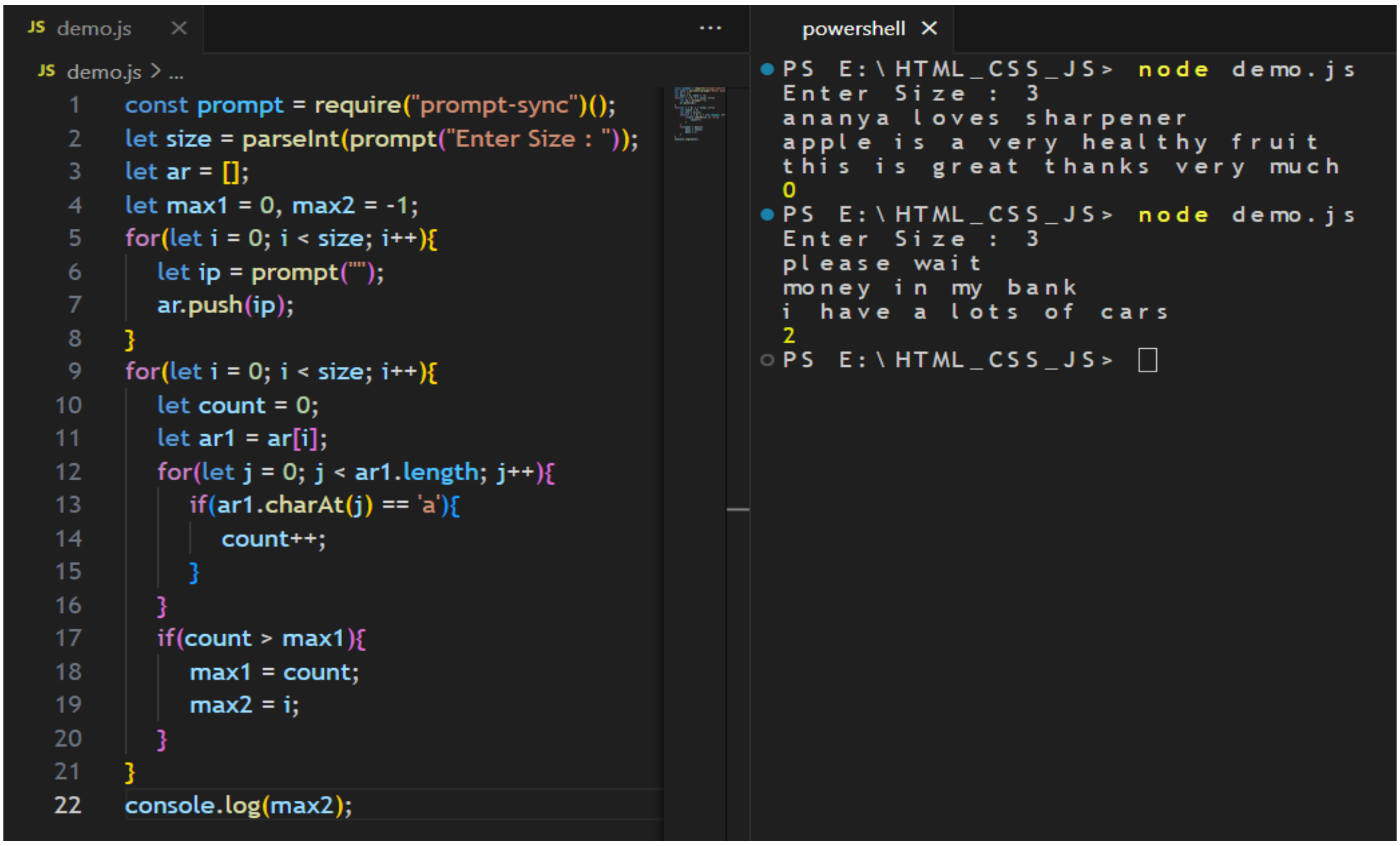
```javascript
const prompt = require("prompt-sync")();
let str1 = prompt("Enter String1 : ");
let ar = str1.split('');
ar.sort();
let res = new Array(ar.length);
let u = 0;
let i = 1;
for(let c of ar){
    if(c.toUpperCase() === c){
        res[u] = c;
        u += 2;
    }else{
        res[i] = c;
        i += 2;
    }
}
console.log(res.join(''));
```

powershell

```
PS E:\HTML_CSS_JS> node demo.js
Enter String1 : bAwutndekWEdkd
AbEdWddekkntuw
PS E:\HTML_CSS_JS> node demo.js
Enter String1 : AiBFR
AiBFR
PS E:\HTML_CSS_JS> 
```

String Question 08:

You are given an array of strings sentences, where each sentences[i] represents a single sentence. Ip: [please wait, money in my bank, i have a lots of cars]   Op: 2


demo.js

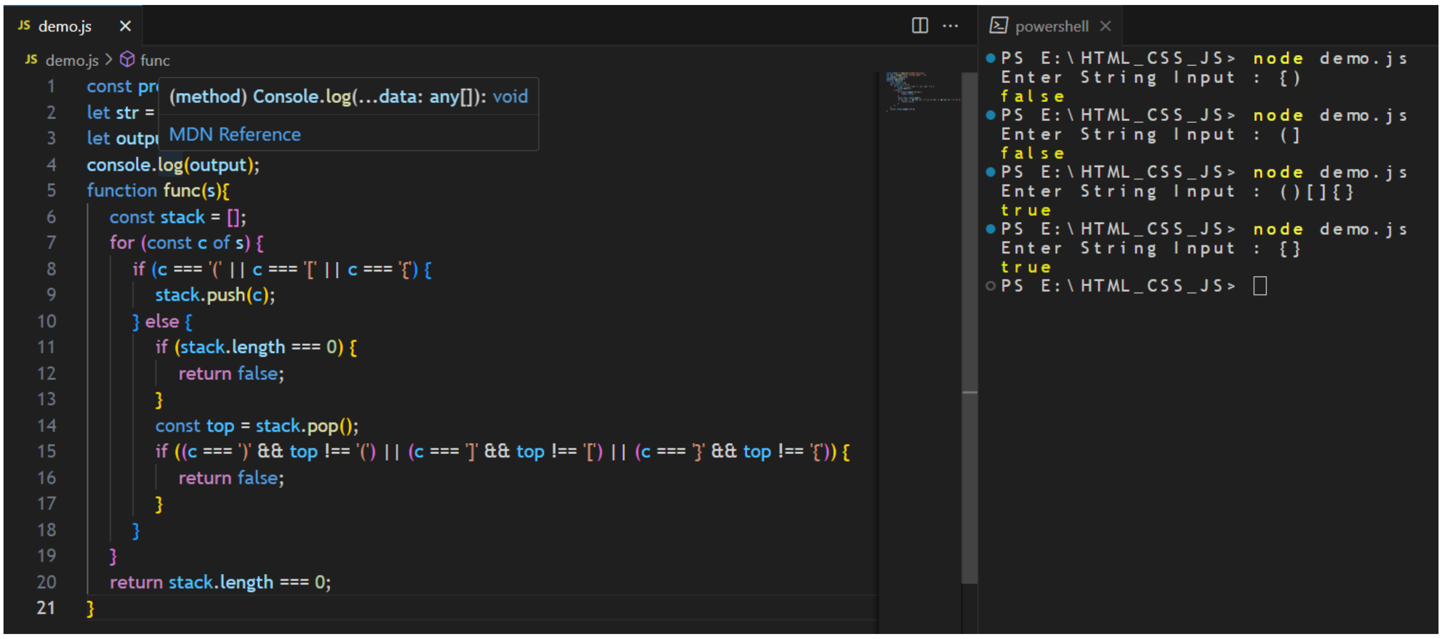
```javascript
const prompt = require("prompt-sync")();
let size = parseInt(prompt("Enter Size : "));
let ar = [];
let max1 = 0, max2 = -1;
for(let i = 0; i < size; i++){
    let ip = prompt("");
    ar.push(ip);
}
for(let i = 0; i < size; i++){
    let count = 0;
    let ar1 = ar[i];
    for(let j = 0; j < ar1.length; j++){
        if(ar1.charAt(j) == 'a'){
            count++;
        }
    }
    if(count > max1){
        max1 = count;
        max2 = i;
    }
}
console.log(max2);
```

powershell

```
PS E:\HTML_CSS_JS> node demo.js
Enter Size : 3
ananya loves sharpener
apple is a very healthy fruit
this is great thanks very much
0
PS E:\HTML_CSS_JS> node demo.js
Enter Size : 3
please wait
money in my bank
i have a lots of cars
2
PS E:\HTML_CSS_JS> 
```

String Question 09:

Given a string s containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid. An input string is valid if: Open brackets must be closed by the same type of brackets. Open brackets must be closed in the correct order. Every close bracket has a corresponding open bracket of the same type.



```
const pr
let str =
let outp
console.log(output);
function func(s){
  const stack = [];
  for (const c of s) {
    if (c === '(' || c === '[' || c === '{') {
      stack.push(c);
    } else {
      if (stack.length === 0) {
        return false;
      }
      const top = stack.pop();
      if ((c === ')' && top !== '(') || (c === ']' && top !== '[') || (c === '}' && top !== '{')) {
        return false;
      }
    }
  }
  return stack.length === 0;
}
```

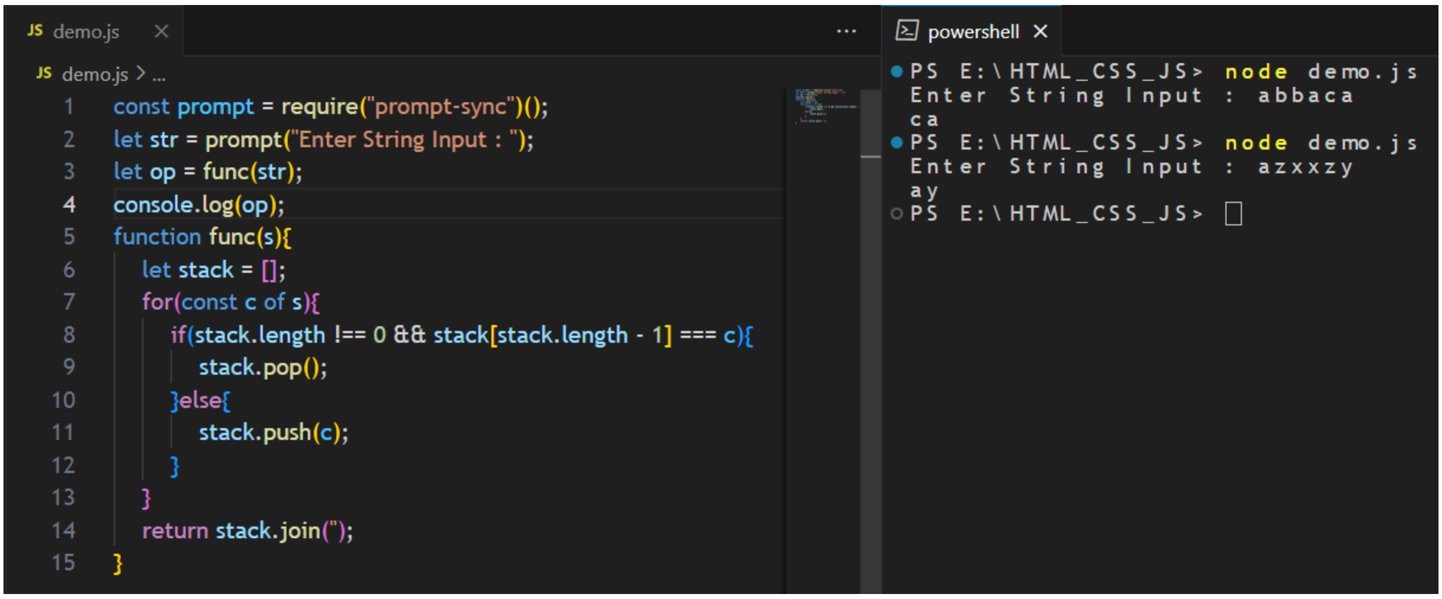
```
(method) Console.log(...data: any[]): void
MDN Reference
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Input : {)
false
PS E:\HTML_CSS_JS> node demo.js
Enter String Input : (]
false
PS E:\HTML_CSS_JS> node demo.js
Enter String Input : ()[]{}
true
PS E:\HTML_CSS_JS> node demo.js
Enter String Input : {}
true
PS E:\HTML_CSS_JS>
```

String Question 10:

You are given a string s consisting of lowercase English letters. A duplicate removal consists of choosing two adjacent and equal letters and removing them. We repeatedly make duplicate removals on s until we no longer can. Return the final string after all such duplicate removals have been made. It can be proven that the answer is unique.

Ip: abbaca Op: ca Ip: azxxzy Op: ay

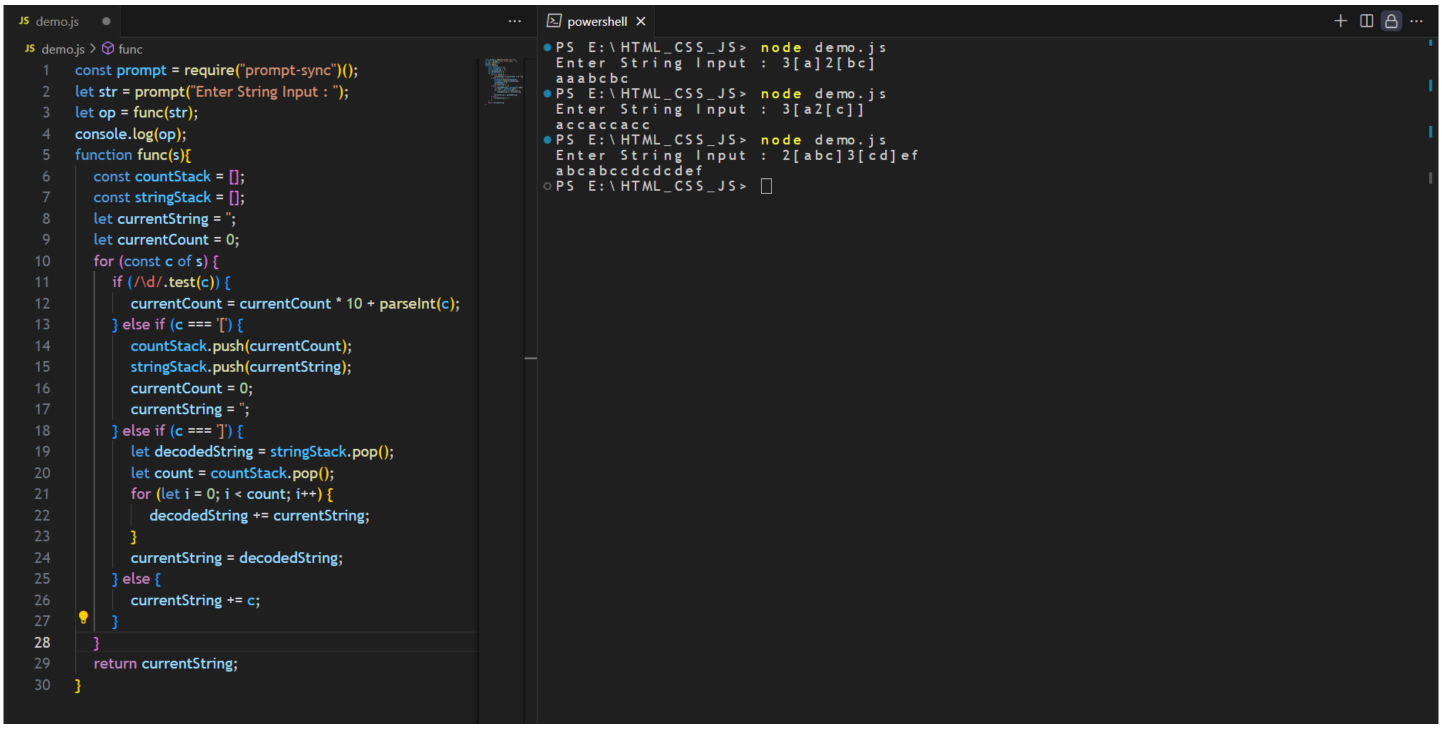


```
const prompt = require("prompt-sync")();
let str = prompt("Enter String Input : ");
let op = func(str);
console.log(op);
function func(s){
  let stack = [];
  for(const c of s){
    if(stack.length !== 0 && stack[stack.length - 1] === c){
      stack.pop();
    }else{
      stack.push(c);
    }
  }
  return stack.join('');
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Input : abbaca
ca
PS E:\HTML_CSS_JS> node demo.js
Enter String Input : azxxzy
ay
PS E:\HTML_CSS_JS>
```

String Question 11:

S = “3[a]2[bc]"   Output: "aaabcbc"   S = "3[a2[c]]"   Output: "accaccacc"



demo.js > func

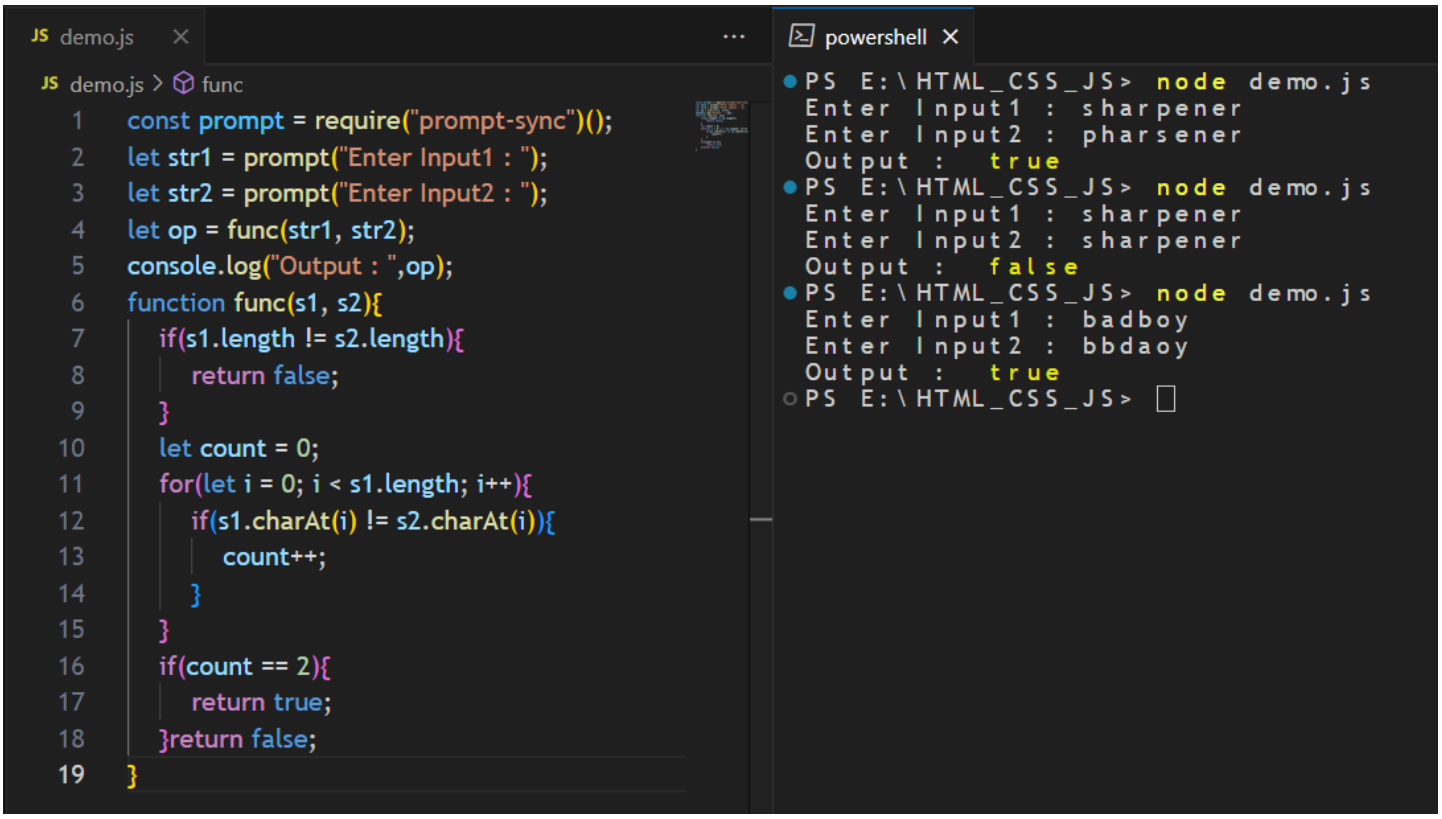
```javascript
const prompt = require("prompt-sync")();
let str = prompt("Enter String Input : ");
let op = func(str);
console.log(op);
function func(s){
   const countStack = [];
   const stringStack = [];
   let currentString = '';
   let currentCount = 0;
   for (const c of s) {
      if (/\d/.test(c)) {
         currentCount = currentCount * 10 + parseInt(c);
      } else if (c === '[') {
         countStack.push(currentCount);
         stringStack.push(currentString);
         currentCount = 0;
         currentString = '';
      } else if (c === ']') {
         let decodedString = stringStack.pop();
         let count = countStack.pop();
         for (let i = 0; i < count; i++) {
            decodedString += currentString;
         }
         currentString = decodedString;
      } else {
         currentString += c;
      }
   }
   return currentString;
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Input : 3[a]2[bc]
aaabcbc
PS E:\HTML_CSS_JS> node demo.js
Enter String Input : 3[a2[c]]
accaccacc
PS E:\HTML_CSS_JS> node demo.js
Enter String Input : 2[abc]3[cd]ef
abcabccdcdcdef
PS E:\HTML_CSS_JS>
```

String Question 12:

You have given two strings. You have to perform a single swap operation to make these strings similar. If it is possible then the strings are pro strings. If pro strings are formed then return True otherwise return False.

Ip: sharpener, pharsener   Op: True   Ip: sharpener, sharpener   Op: false



demo.js > func

```javascript
const prompt = require("prompt-sync")();
let str1 = prompt("Enter Input1 : ");
let str2 = prompt("Enter Input2 : ");
let op = func(str1, str2);
console.log("Output : ",op);
function func(s1, s2){
    if(s1.length != s2.length){
        return false;
    }
    let count = 0;
    for(let i = 0; i < s1.length; i++){
        if(s1.charAt(i) != s2.charAt(i)){
            count++;
        }
    }
    if(count == 2){
        return true;
    }return false;
}
```

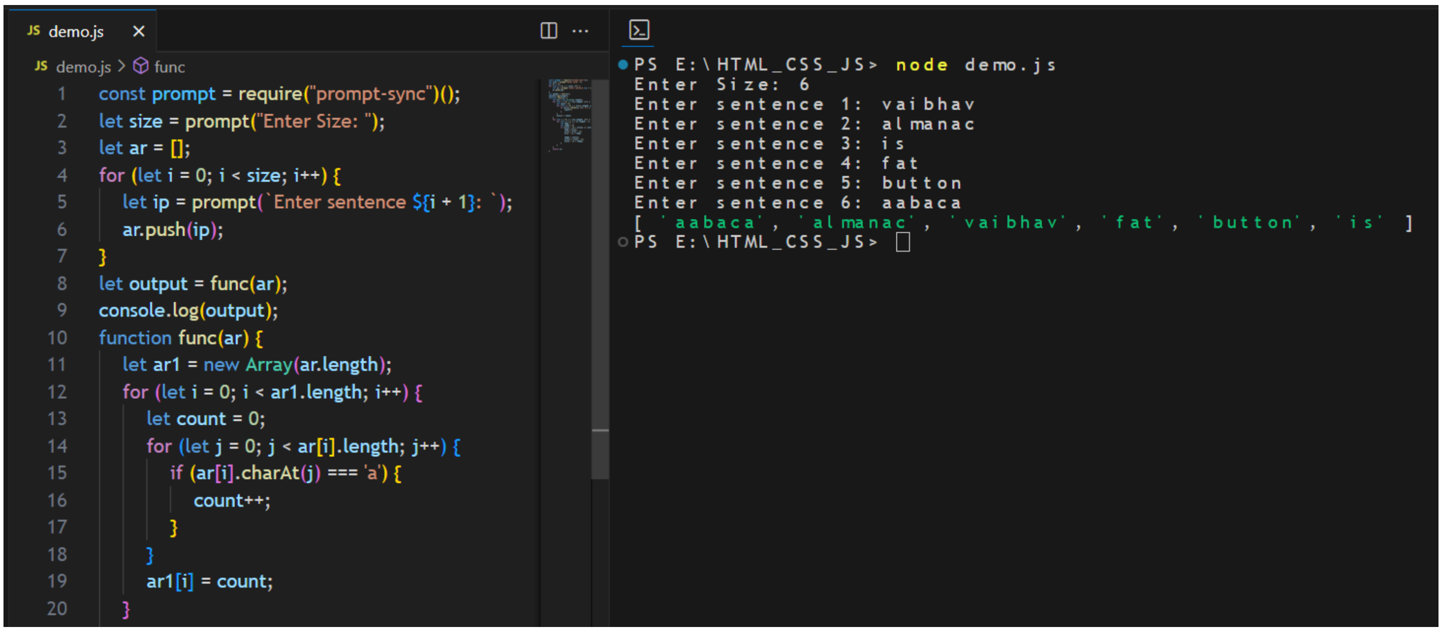
```
PS E:\HTML_CSS_JS> node demo.js
Enter Input1 : sharpener
Enter Input2 : pharsener
Output :  true
PS E:\HTML_CSS_JS> node demo.js
Enter Input1 : sharpener
Enter Input2 : sharpener
Output :  false
PS E:\HTML_CSS_JS> node demo.js
Enter Input1 : badboy
Enter Input2 : bbdaoy
Output :  true
PS E:\HTML_CSS_JS>
```

String Question 13:

Given an array Containing Strings sort on the basis of number of character 'a' present. Return the sorted array Note if number of a is same then length will be given preference.

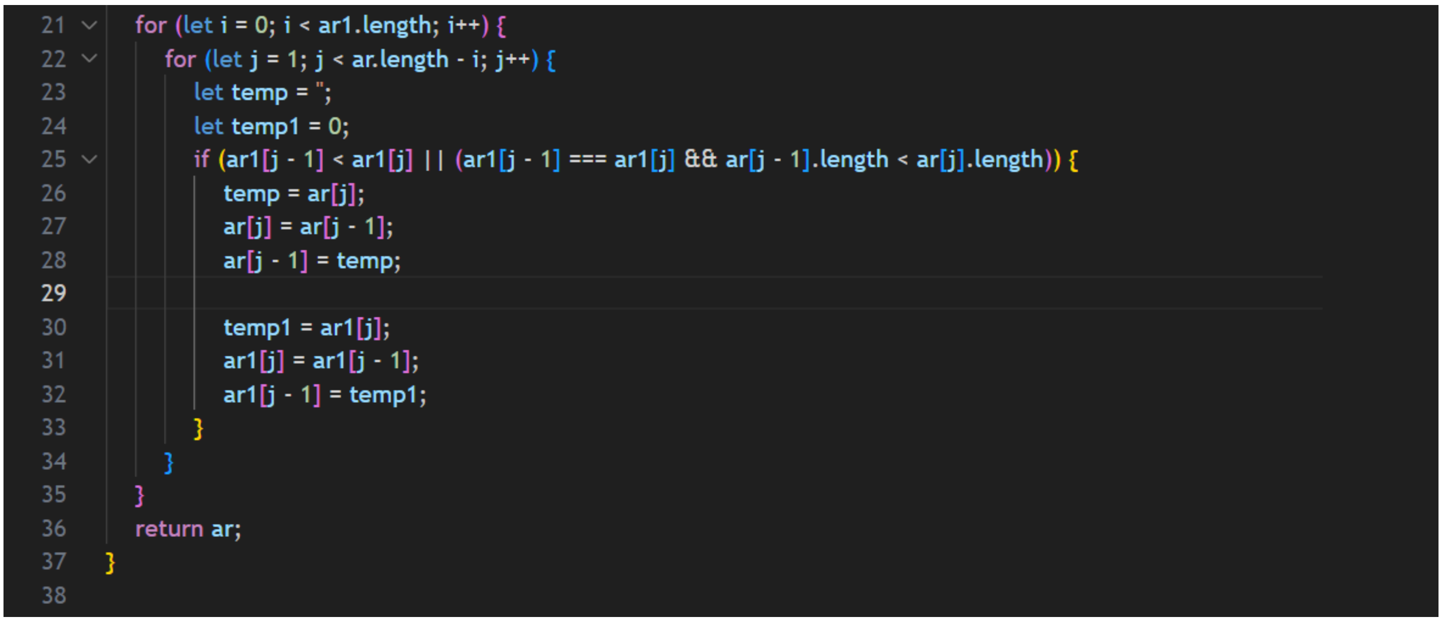
Ip: [vaibhav, almanac, is, fat, button, aabaca]

Op: [aabaca, almanac, vaibhav, fat, button, is]



```javascript
const prompt = require("prompt-sync")();
let size = prompt("Enter Size: ");
let ar = [];
for (let i = 0; i < size; i++) {
    let ip = prompt(`Enter sentence ${i + 1}: `);
    ar.push(ip);
}
let output = func(ar);
console.log(output);
function func(ar) {
    let ar1 = new Array(ar.length);
    for (let i = 0; i < ar1.length; i++) {
        let count = 0;
        for (let j = 0; j < ar[i].length; j++) {
            if (ar[i].charAt(j) === 'a') {
                count++;
            }
        }
        ar1[i] = count;
    }
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Size: 6
Enter sentence 1: vaibhav
Enter sentence 2: almanac
Enter sentence 3: is
Enter sentence 4: fat
Enter sentence 5: button
Enter sentence 6: aabaca
[ 'aabaca', 'almanac', 'vaibhav', 'fat', 'button', 'is' ]
PS E:\HTML_CSS_JS> 
```

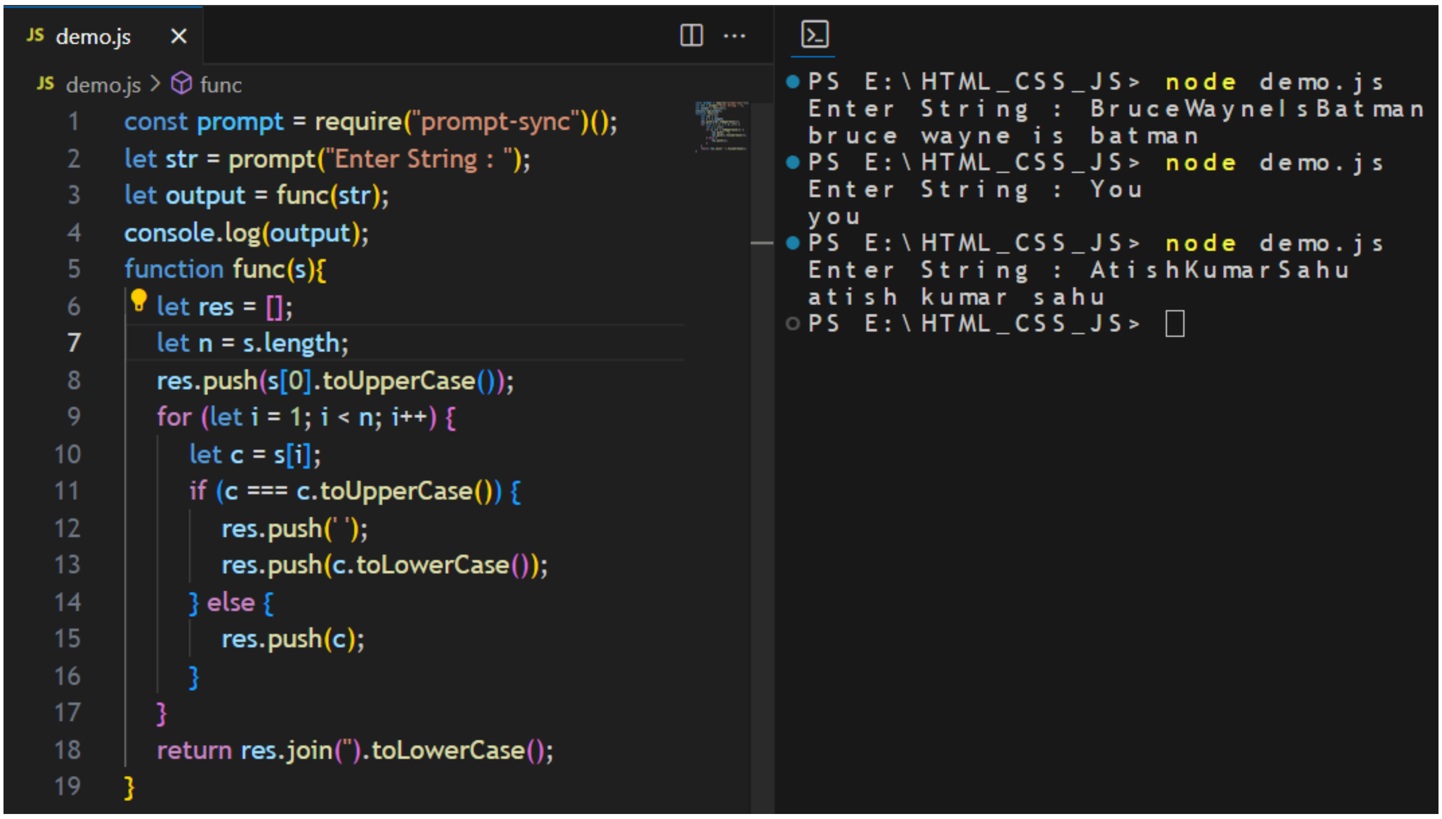


```javascript
    for (let i = 0; i < ar1.length; i++) {
        for (let j = 1; j < ar.length - i; j++) {
            let temp = '';
            let temp1 = 0;
            if (ar1[j - 1] < ar1[j] || (ar1[j - 1] === ar1[j] && ar[j - 1].length < ar[j].length)) {
                temp = ar[j];
                ar[j] = ar[j - 1];
                ar[j - 1] = temp;

                temp1 = ar1[j];
                ar1[j] = ar1[j - 1];
                ar1[j - 1] = temp1;
            }
        }
    }
    return ar;
}
```

String Question 14:

You downloaded a word file of content for your work . but some error happens and spaces between words in sentences vanished . You have given a string which can be a sentence without spaces . you have to give it spaces and change it to fully lowercase.

Ip: BruceWayneIsBatman   Op: bruce wayne is batman



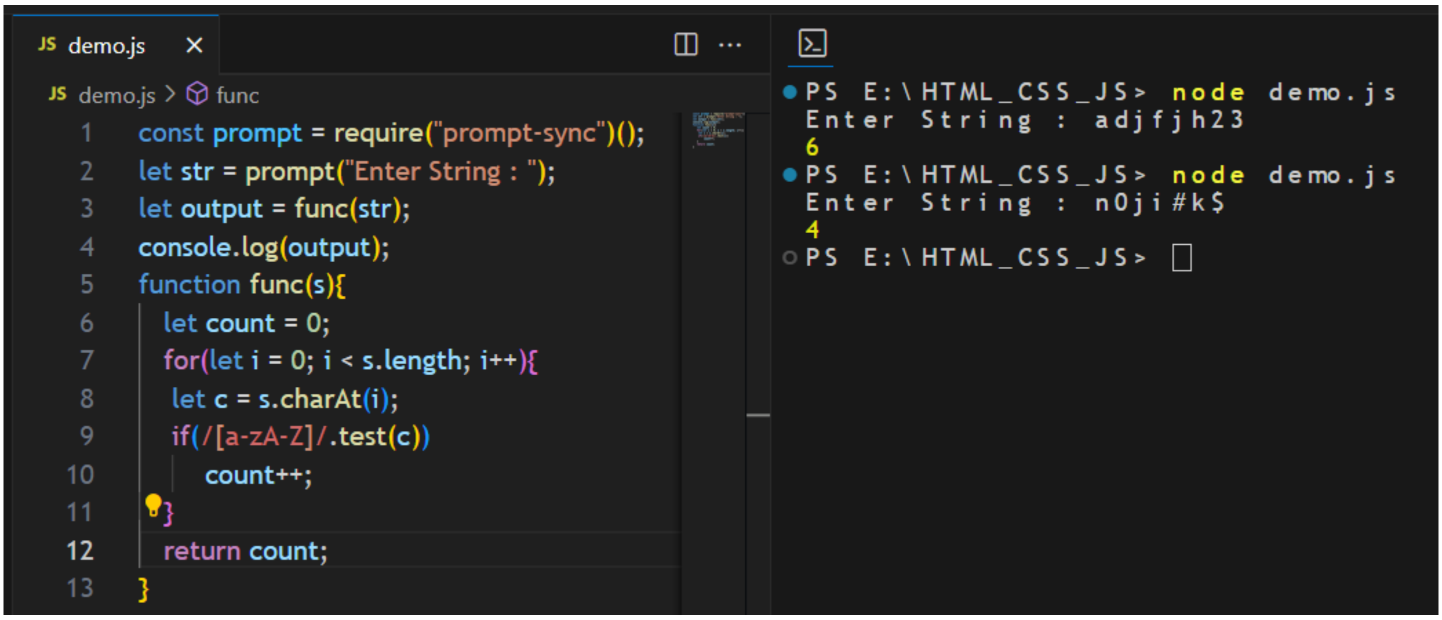
```javascript
const prompt = require("prompt-sync")();
let str = prompt("Enter String : ");
let output = func(str);
console.log(output);
function func(s){
  let res = [];
  let n = s.length;
  res.push(s[0].toUpperCase());
  for (let i = 1; i < n; i++) {
    let c = s[i];
    if (c === c.toUpperCase()) {
      res.push(' ');
      res.push(c.toLowerCase());
    } else {
      res.push(c);
    }
  }
  return res.join('').toLowerCase();
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String : BruceWayneIsBatman
bruce wayne is batman
PS E:\HTML_CSS_JS> node demo.js
Enter String : You
you
PS E:\HTML_CSS_JS> node demo.js
Enter String : AtishKumarSahu
atish kumar sahu
PS E:\HTML_CSS_JS> 
```

String Question 15:

You are design a application where for authentication you write a logic that password should contain 8 characters . but the password that user put can contain integers, special characters as well. You have to find the number of alphabet characters.

Ip: adjfjh23   Op: 6   Ip: n0ji#ks   Op: 4

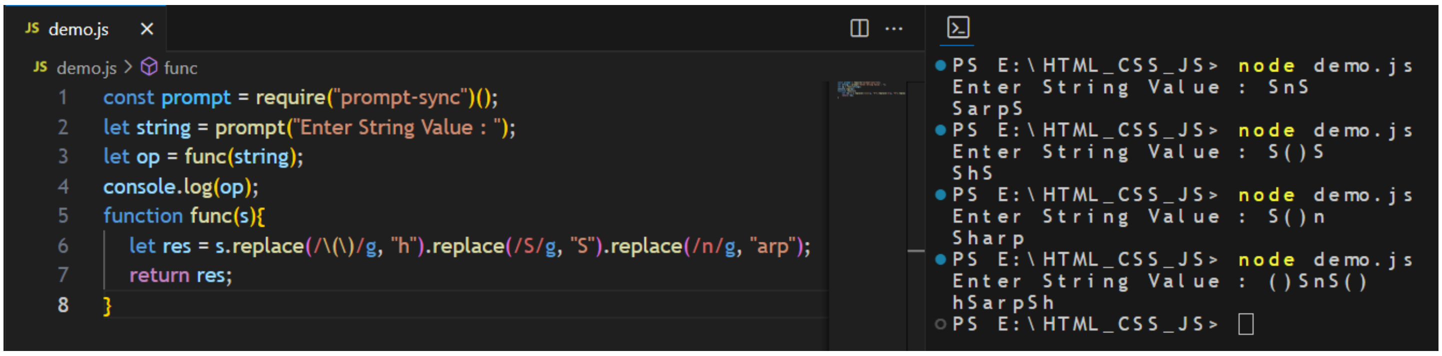


```javascript
const prompt = require("prompt-sync")();
let str = prompt("Enter String : ");
let output = func(str);
console.log(output);
function func(s){
  let count = 0;
  for(let i = 0; i < s.length; i++){
   let c = s.charAt(i);
   if(/[a-zA-Z]/.test(c))
      count++;
  }
  return count;
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String : adjfjh23
6
PS E:\HTML_CSS_JS> node demo.js
Enter String : n0ji#k$
4
PS E:\HTML_CSS_JS> 
```

String Question 16:

Ip: S()n   Op: Sharp   Ip: Sn   Op: Sarp

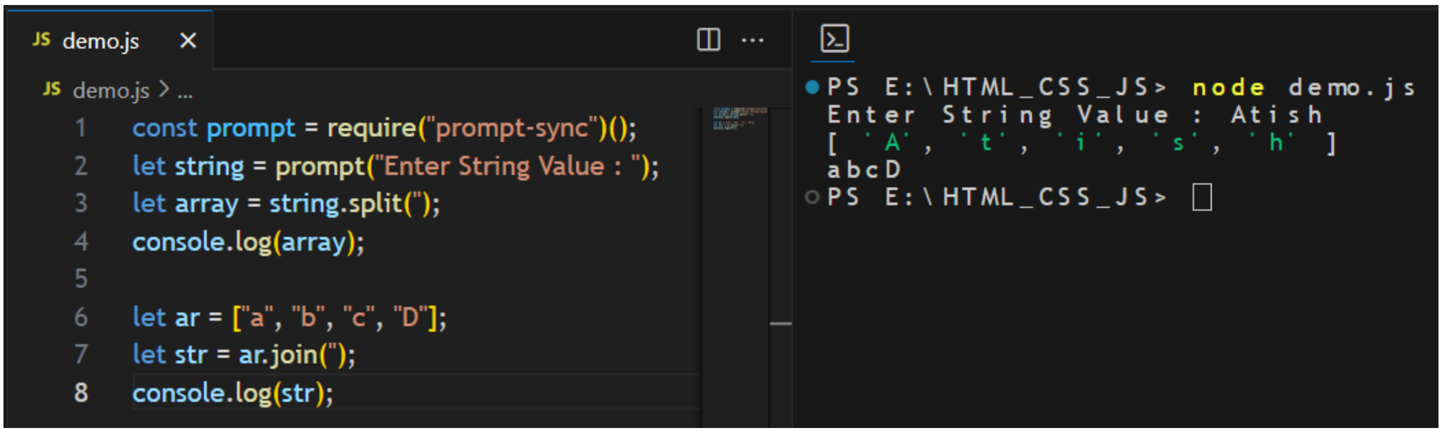


```javascript
const prompt = require("prompt-sync")();
let string = prompt("Enter String Value : ");
let op = func(string);
console.log(op);
function func(s){
    let res = s.replace(/\(\)/g, "h").replace(/S/g, "S").replace(/n/g, "arp");
    return res;
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : SnS
SarpS
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : S()S
ShS
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : S()n
Sharp
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : ()SnS()
hSarpSh
PS E:\HTML_CSS_JS>
```

String Question 17:

Convert a string to array and array to string.



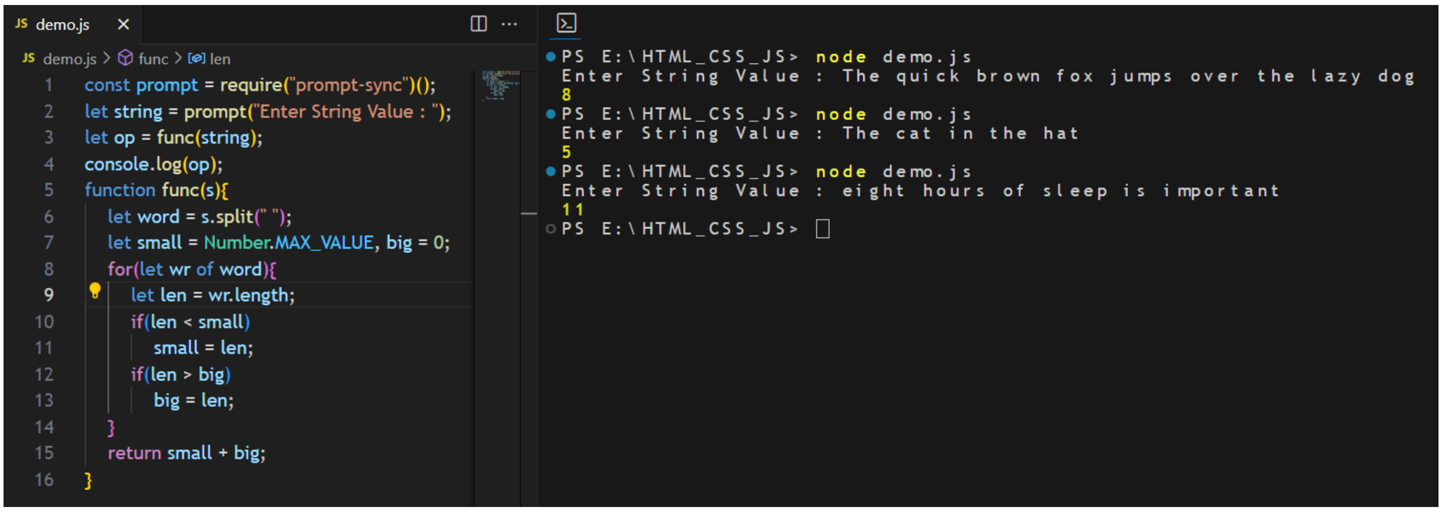
```javascript
const prompt = require("prompt-sync")();
let string = prompt("Enter String Value : ");
let array = string.split('');
console.log(array);

let ar = ["a", "b", "c", "D"];
let str = ar.join('');
console.log(str);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : Atish
[ 'A', 't', 'i', 's', 'h' ]
abcD
PS E:\HTML_CSS_JS>
```

String Question 18:

This program take a .String of text as input and finds the length of the shortest and longest word in the String. it then calculates their sum and return.

Input: The quick brown fox jumps over the lazy dog   Output: 8

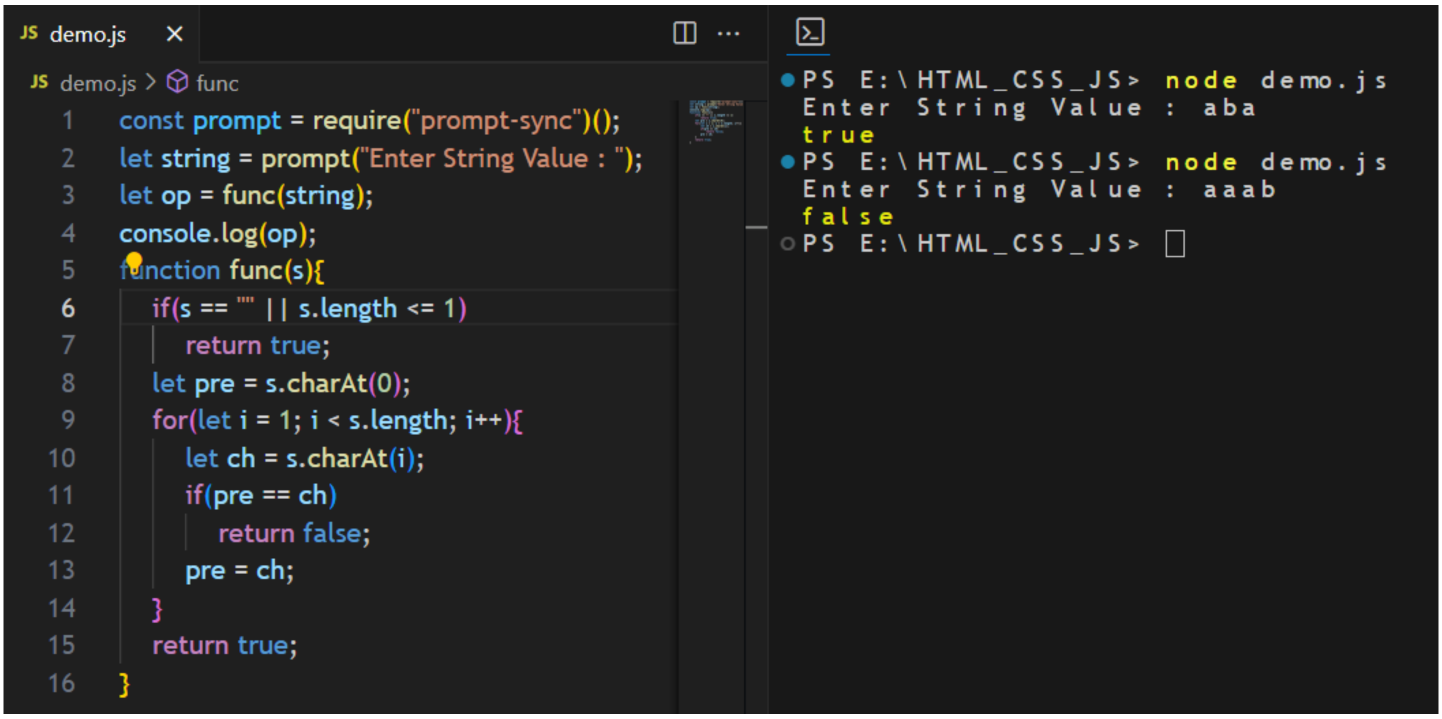


```javascript
const prompt = require("prompt-sync")();
let string = prompt("Enter String Value : ");
let op = func(string);
console.log(op);
function func(s){
    let word = s.split(" ");
    let small = Number.MAX_VALUE, big = 0;
    for(let wr of word){
        let len = wr.length;
        if(len < small)
            small = len;
        if(len > big)
            big = len;
    }
    return small + big;
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : The quick brown fox jumps over the lazy dog
8
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : The cat in the hat
5
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : eight hours of sleep is important
11
PS E:\HTML_CSS_JS>
```

String Question 19:

Given a String check whether it is an ideal String or not. Return true if it is ideal else return false. Ideal String is a type of String in which no two contiguous character are equal. Input: "aba"   output: true   input: "aaab"   output: false



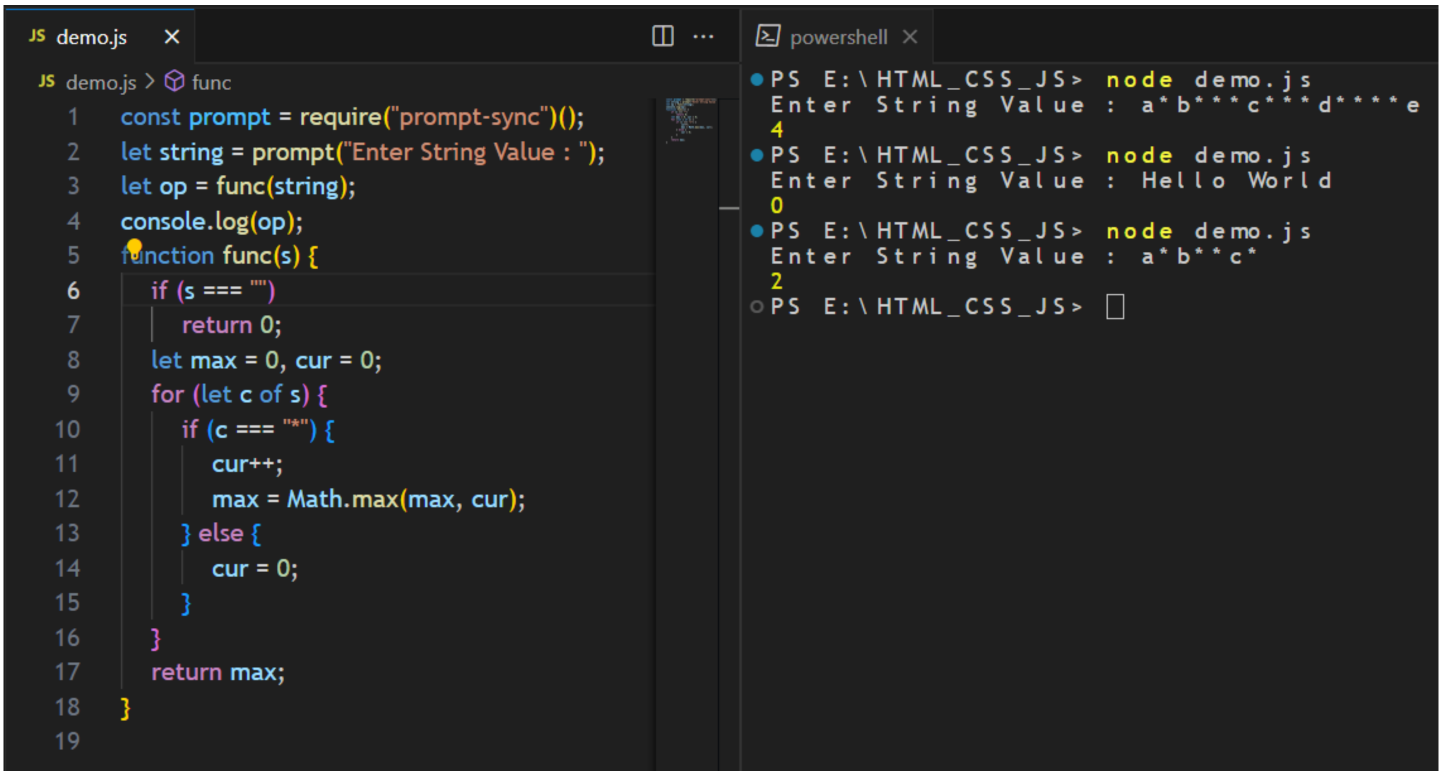
```javascript
const prompt = require("prompt-sync")();
let string = prompt("Enter String Value : ");
let op = func(string);
console.log(op);
function func(s){
    if(s == "" || s.length <= 1)
        return true;
    let pre = s.charAt(0);
    for(let i = 1; i < s.length; i++){
        let ch = s.charAt(i);
        if(pre == ch)
            return false;
        pre = ch;
    }
    return true;
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : aba
true
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : aaab
false
PS E:\HTML_CSS_JS>
```

String Question 20:

What is the maximum contiguous frequency of * in a given String?

Input: "a*b***c***d****e" output: 4   Input: "Hello World"   output: 0

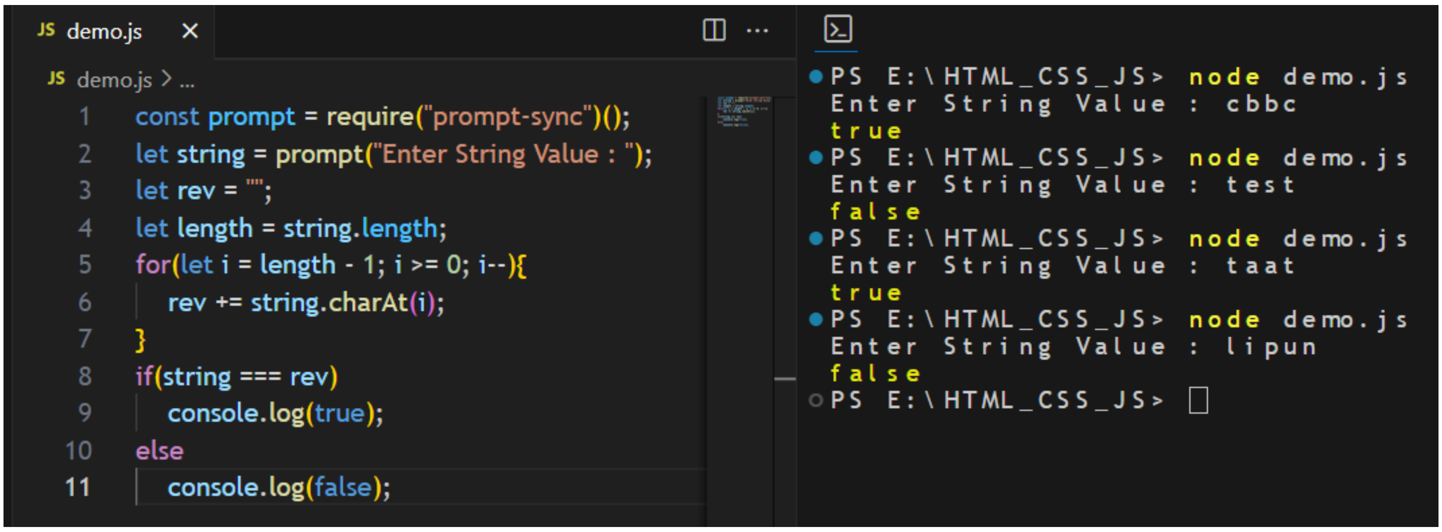


```javascript
const prompt = require("prompt-sync")();
let string = prompt("Enter String Value : ");
let op = func(string);
console.log(op);
function func(s) {
    if (s === "")
        return 0;
    let max = 0, cur = 0;
    for (let c of s) {
        if (c === "*") {
            cur++;
            max = Math.max(max, cur);
        } else {
            cur = 0;
        }
    }
    return max;
}

```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : a*b***c***d****e
4
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : Hello World
0
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : a*b**c*
2
PS E:\HTML_CSS_JS>
```

String Question 21:

String Palindrome   Ip: cbbc   Op: true



demo.js

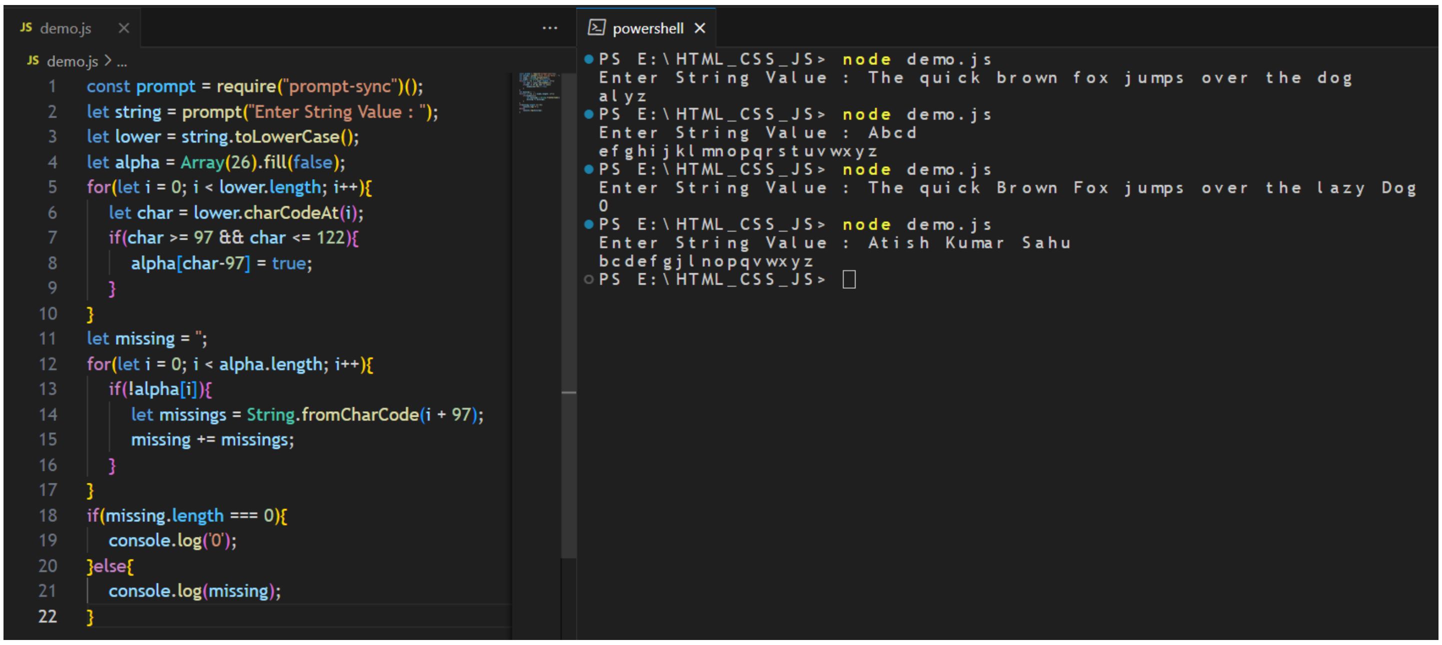
```
const prompt = require("prompt-sync")();
let string = prompt("Enter String Value : ");
let rev = "";
let length = string.length;
for(let i = length - 1; i >= 0; i--){
    rev += string.charAt(i);
}
if(string === rev)
    console.log(true);
else
    console.log(false);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : cbbc
true
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : test
false
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : taat
true
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : lipun
false
PS E:\HTML_CSS_JS>
```

String Question 22:

Given a sentence str. the task is to find whether the given sentence contains all letters of the English alphabet a to z or A to Z. if does not then print all missing letters of the alphabet otherwise print 0.

Ip: The quick brown fox jumps over the dog   Op: alyz



demo.js

```
const prompt = require("prompt-sync")();
let string = prompt("Enter String Value : ");
let lower = string.toLowerCase();
let alpha = Array(26).fill(false);
for(let i = 0; i < lower.length; i++){
    let char = lower.charCodeAt(i);
    if(char >= 97 && char <= 122){
        alpha[char-97] = true;
    }
}
let missing = '';
for(let i = 0; i < alpha.length; i++){
    if(!alpha[i]){
        let missings = String.fromCharCode(i + 97);
        missing += missings;
    }
}
if(missing.length === 0){
    console.log('0');
}else{
    console.log(missing);
}
```

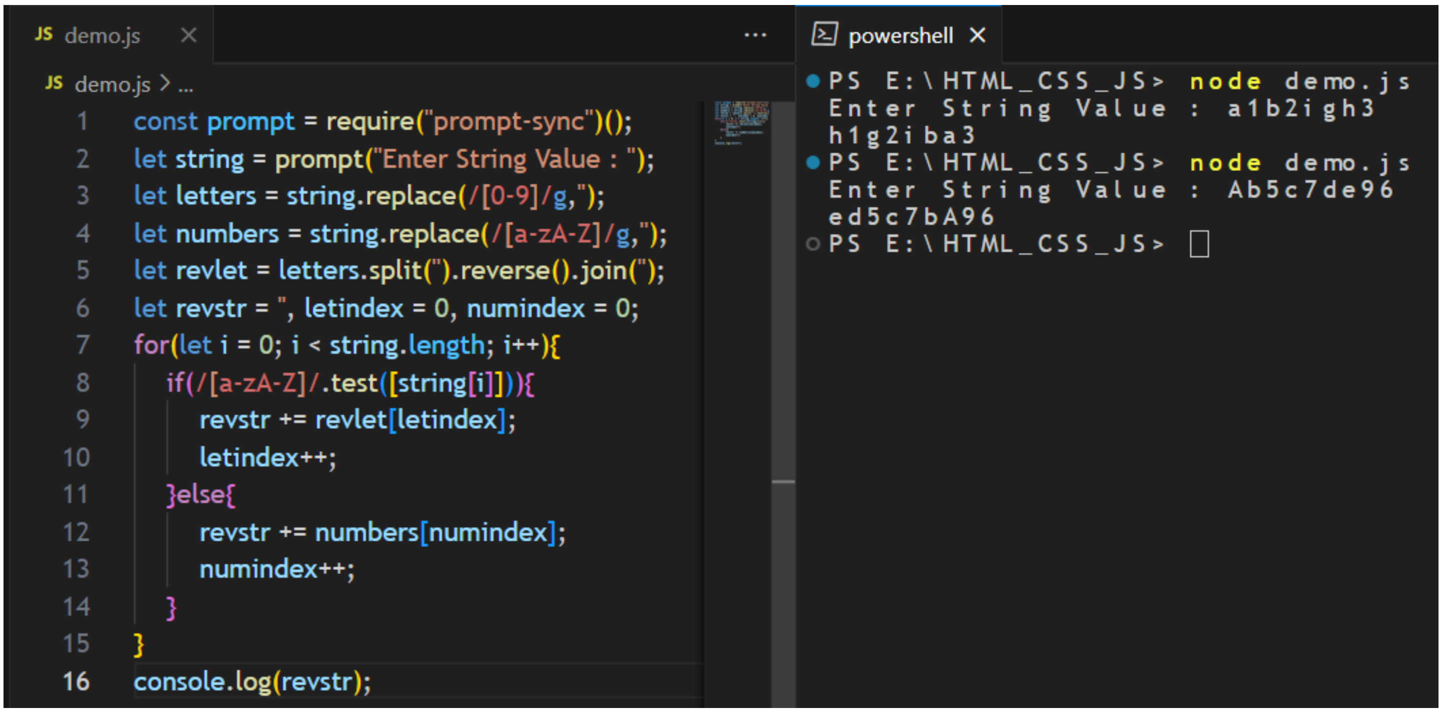
powershell

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : The quick brown fox jumps over the dog
alyz
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : Abcd
efghijklmnopqrstuvwxyz
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : The quick Brown Fox jumps over the lazy Dog
0
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : Atish Kumar Sahu
bcdefgjlnopqvwxyz
PS E:\HTML_CSS_JS>
```

String Question 23:

Given a string str which contains numbers 0 to 9 and also letters of the English alphabets a to z and A to Z the task is to reverse the string in a such a way that the position of number in the string are left unaltered.

Input: a1b2igh3 Output: h1g2iba3   Ip: Ab5c7de96   Op: ed5c7bA96

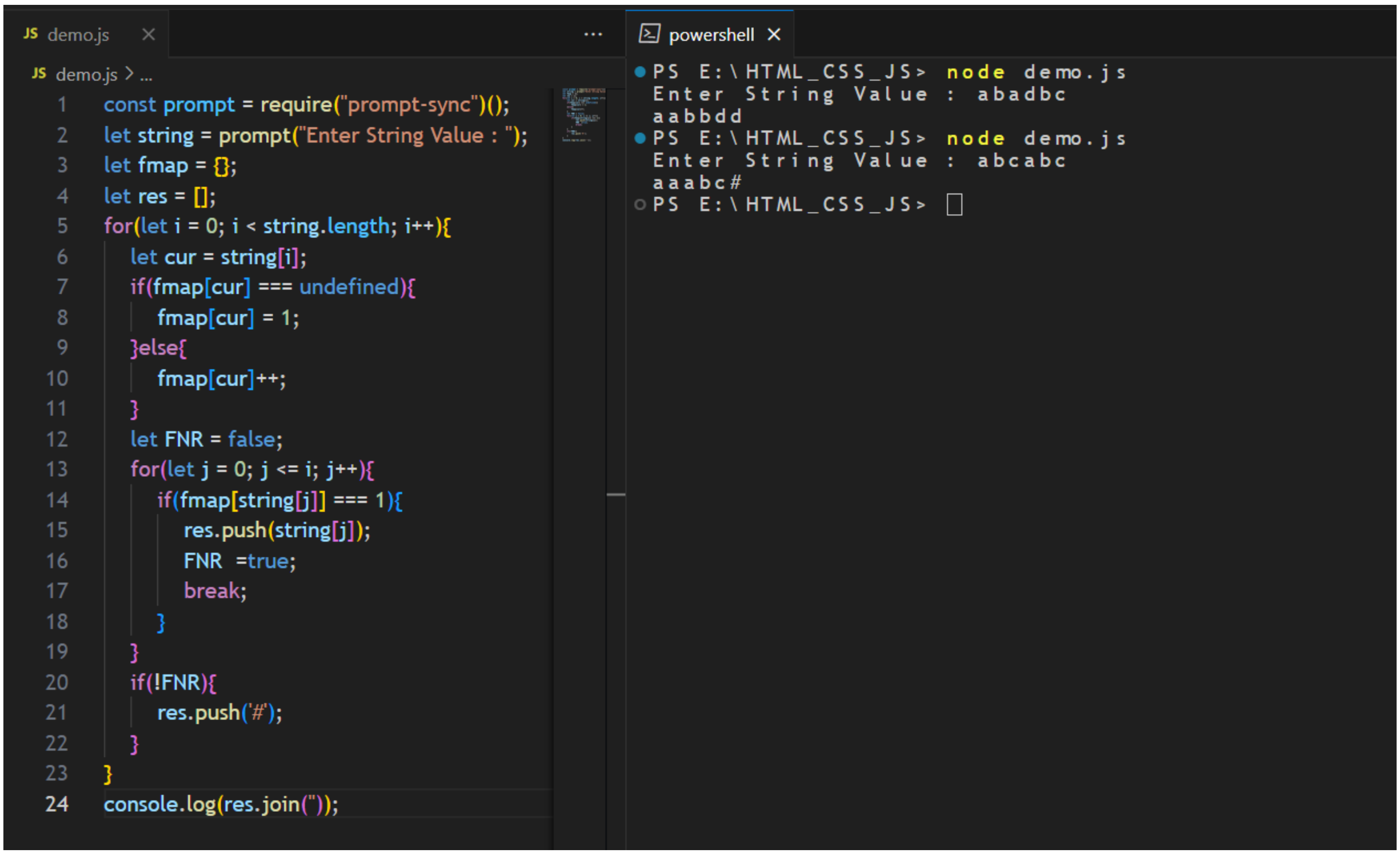

demo.js
powershell

```javascript
const prompt = require("prompt-sync")();
let string = prompt("Enter String Value : ");
let letters = string.replace(/[0-9]/g,'');
let numbers = string.replace(/[a-zA-Z]/g,'');
let revlet = letters.split('').reverse().join('');
let revstr = '', letindex = 0, numindex = 0;
for(let i = 0; i < string.length; i++){
    if(/[a-zA-Z]/.test([string[i]])){
        revstr += revlet[letindex];
        letindex++;
    }else{
        revstr += numbers[numindex];
        numindex++;
    }
}
console.log(revstr);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : a1b2igh3
h1g2iba3
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : Ab5c7de96
ed5c7bA96
PS E:\HTML_CSS_JS>
```

String Question 24:

Input: “abadbc” input: “abcabc” output: aabbdd output: aaabc#

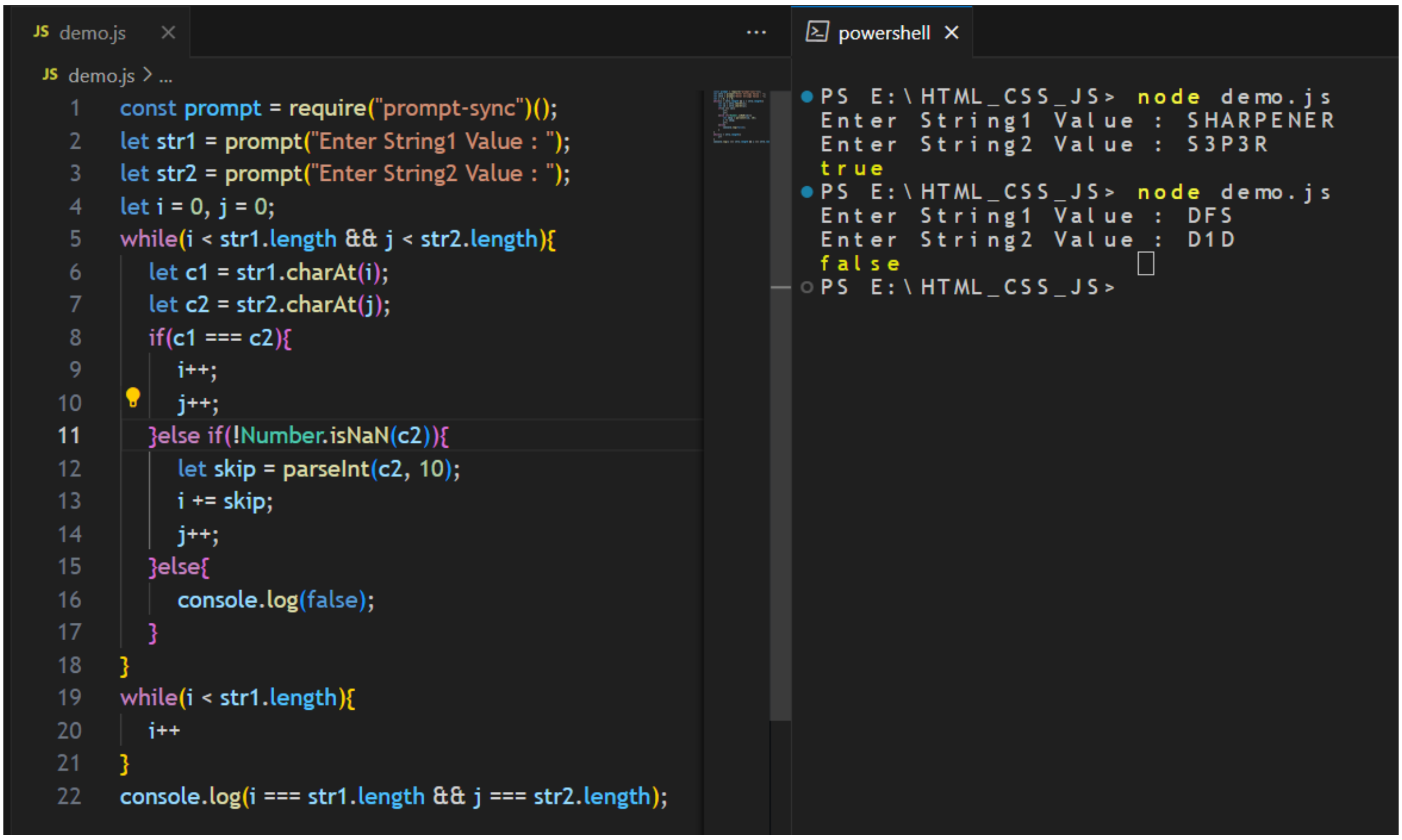

demo.js
powershell

```javascript
const prompt = require("prompt-sync")();
let string = prompt("Enter String Value : ");
let fmap = {};
let res = [];
for(let i = 0; i < string.length; i++){
    let cur = string[i];
    if(fmap[cur] === undefined){
        fmap[cur] = 1;
    }else{
        fmap[cur]++;
    }
    let FNR = false;
    for(let j = 0; j <= i; j++){
        if(fmap[string[j]] === 1){
            res.push(string[j]);
            FNR  =true;
            break;
        }
    }
    if(!FNR){
        res.push('#');
    }
}
console.log(res.join(''));
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : abadbc
aabbdd
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : abcabc
aaabc#
PS E:\HTML_CSS_JS>
```

String Question 25:

input: str1 = SHARPENER str2 = S3P3R output: true



demo.js

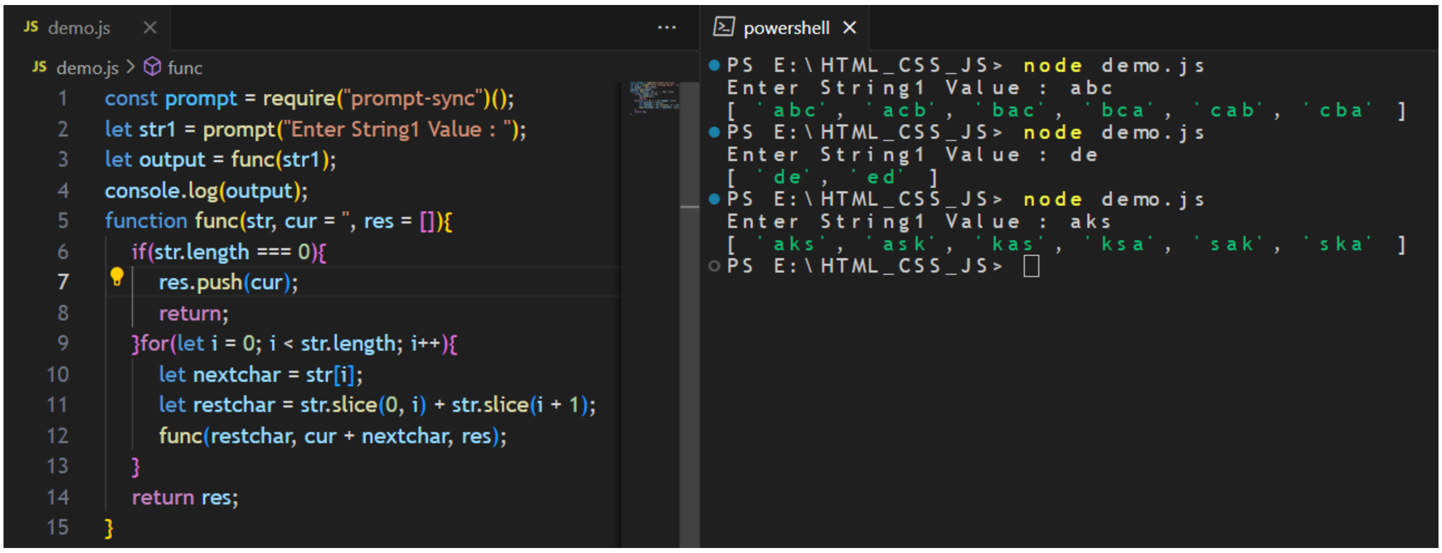
```javascript
const prompt = require("prompt-sync")();
let str1 = prompt("Enter String1 Value : ");
let str2 = prompt("Enter String2 Value : ");
let i = 0, j = 0;
while(i < str1.length && j < str2.length){
    let c1 = str1.charAt(i);
    let c2 = str2.charAt(j);
    if(c1 === c2){
        i++;
        j++;
    }else if(!Number.isNaN(c2)){
        let skip = parseInt(c2, 10);
        i += skip;
        j++;
    }else{
        console.log(false);
    }
}
while(i < str1.length){
    i++
}
console.log(i === str1.length && j === str2.length);
```

powershell

```
PS E:\HTML_CSS_JS> node demo.js
Enter String1 Value : SHARPENER
Enter String2 Value : S3P3R
true
PS E:\HTML_CSS_JS> node demo.js
Enter String1 Value : DFS
Enter String2 Value : D1D
false
PS E:\HTML_CSS_JS>
```

String Question 26:

Write a recursive program to find all the permutations of a string and store it in a lost or ArrayList or vectors. Input: abc output: abc, acb, bac, bca, cab, cba



demo.js > func

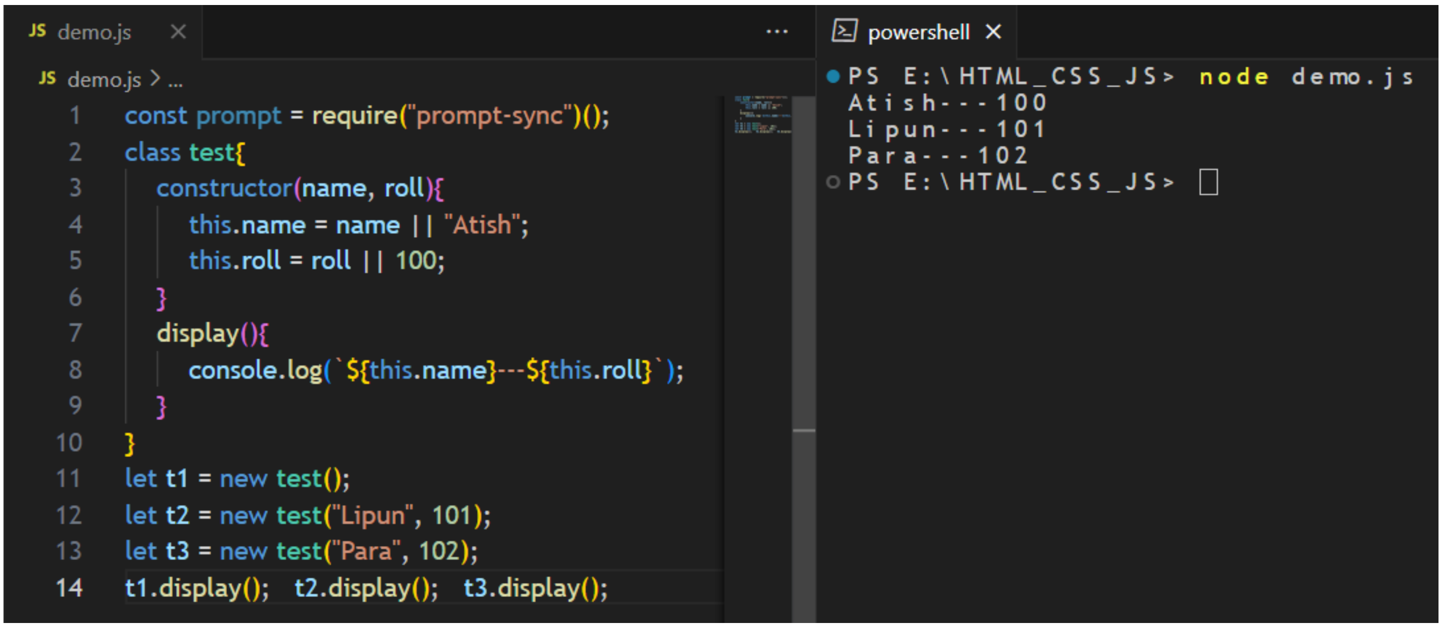
```javascript
const prompt = require("prompt-sync")();
let str1 = prompt("Enter String1 Value : ");
let output = func(str1);
console.log(output);
function func(str, cur = '', res = []){
    if(str.length === 0){
        res.push(cur);
        return;
    }for(let i = 0; i < str.length; i++){
        let nextchar = str[i];
        let restchar = str.slice(0, i) + str.slice(i + 1);
        func(restchar, cur + nextchar, res);
    }
    return res;
}
```

powershell

```
PS E:\HTML_CSS_JS> node demo.js
Enter String1 Value : abc
[ 'abc', 'acb', 'bac', 'bca', 'cab', 'cba' ]
PS E:\HTML_CSS_JS> node demo.js
Enter String1 Value : de
[ 'de', 'ed' ]
PS E:\HTML_CSS_JS> node demo.js
Enter String1 Value : aks
[ 'aks', 'ask', 'kas', 'ksa', 'sak', 'ska' ]
PS E:\HTML_CSS_JS>
```

String Question 27:

A class name with String variable name and integer variable roll and constructor is already created. Created a method display to print the attributes name and roll.



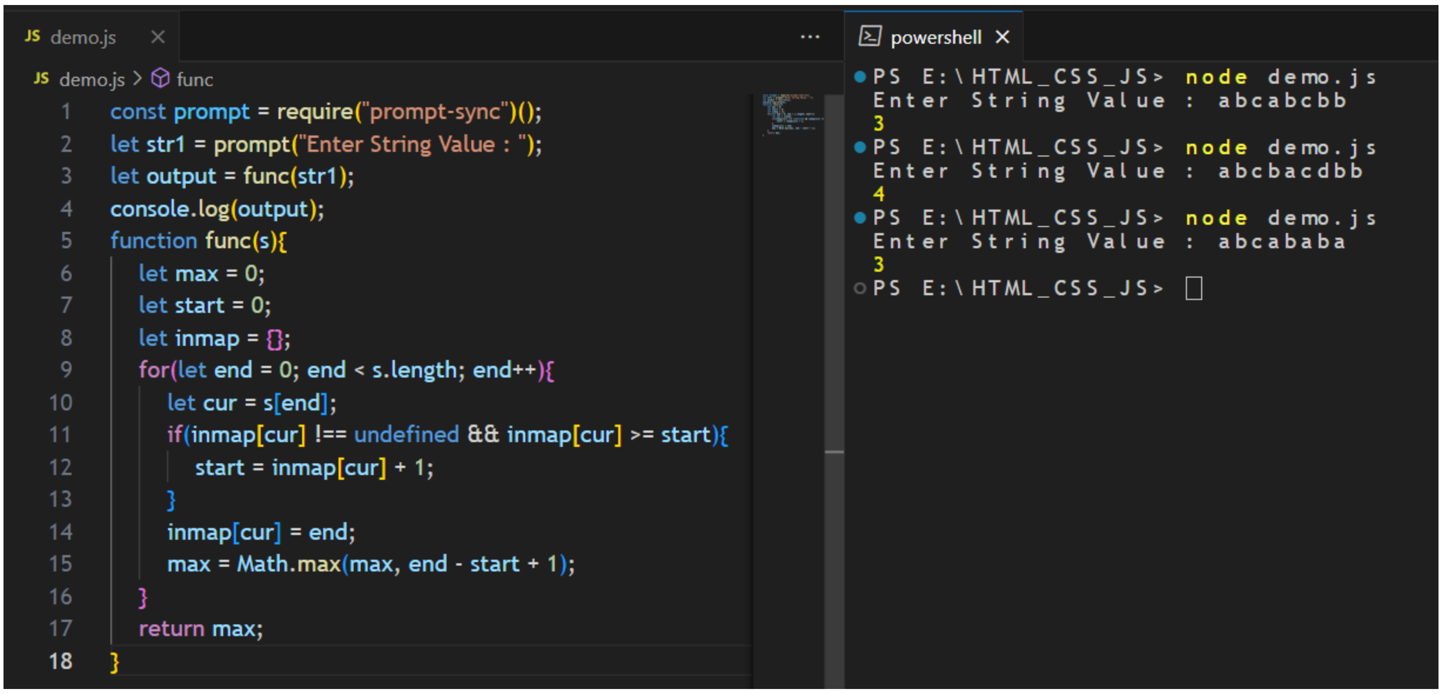
```javascript
const prompt = require("prompt-sync")();
class test{
    constructor(name, roll){
        this.name = name || "Atish";
        this.roll = roll || 100;
    }
    display(){
        console.log(`${this.name}---${this.roll}`);
    }
}
let t1 = new test();
let t2 = new test("Lipun", 101);
let t3 = new test("Para", 102);
t1.display();  t2.display();  t3.display();
```

```
PS E:\HTML_CSS_JS> node demo.js
Atish---100
Lipun---101
Para---102
PS E:\HTML_CSS_JS>
```

String Question 28:

Input: "abcabcbb" output : 3 explain: "abc" repeating character with length of 3

Input: "abcbacdbb" output: 4 explain: "bacd" repeating character with length 4



```javascript
const prompt = require("prompt-sync")();
let str1 = prompt("Enter String Value : ");
let output = func(str1);
console.log(output);
function func(s){
    let max = 0;
    let start = 0;
    let inmap = {};
    for(let end = 0; end < s.length; end++){
        let cur = s[end];
        if(inmap[cur] !== undefined && inmap[cur] >= start){
            start = inmap[cur] + 1;
        }
        inmap[cur] = end;
        max = Math.max(max, end - start + 1);
    }
    return max;
}
```

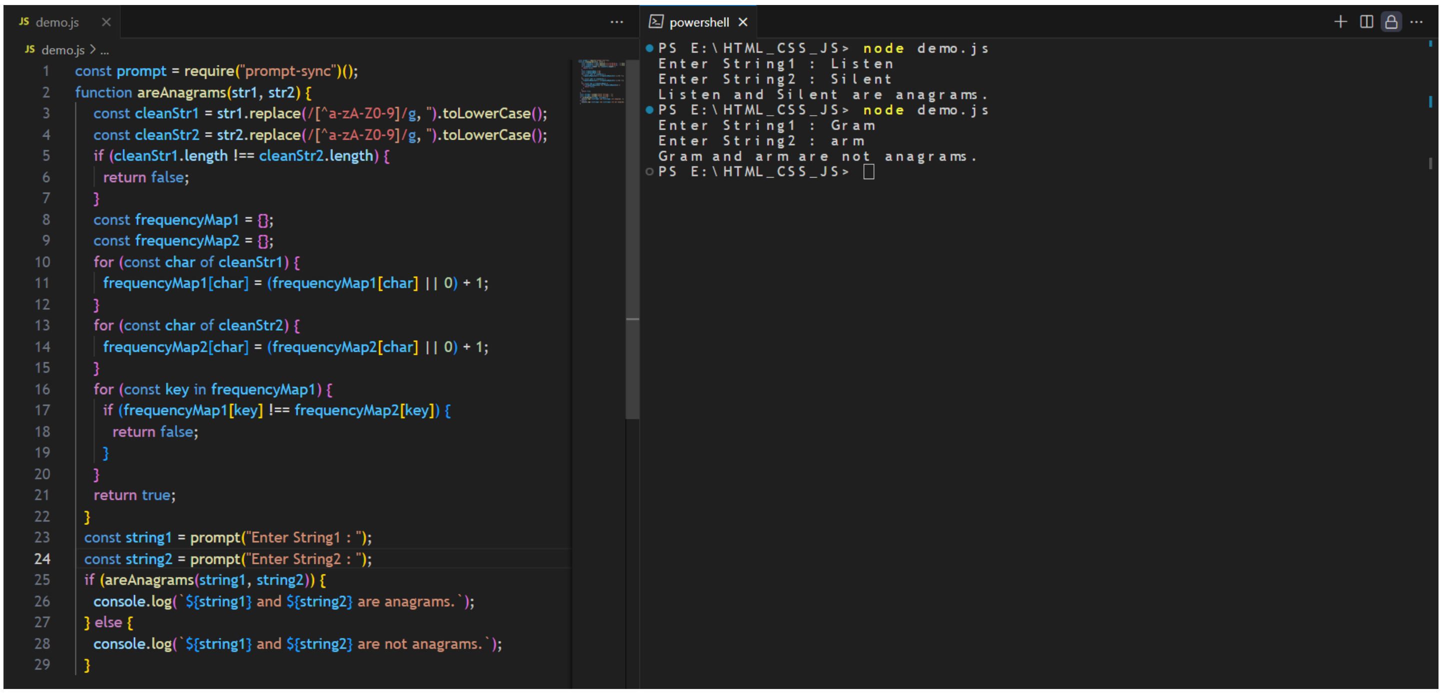
```
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : abcabcbb
3
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : abcbacdbb
4
PS E:\HTML_CSS_JS> node demo.js
Enter String Value : abcababa
3
PS E:\HTML_CSS_JS>
```

String Question 29:

Check whether two Strings are anagram of each other

Input: str1 = "listen" str2 = "silent" Output: "Anagram"

Input: str1 = "gram" str2 = "arm" Output: "Not Anagram"



```javascript
const prompt = require("prompt-sync")();
function areAnagrams(str1, str2) {
  const cleanStr1 = str1.replace(/[^a-zA-Z0-9]/g, '').toLowerCase();
  const cleanStr2 = str2.replace(/[^a-zA-Z0-9]/g, '').toLowerCase();
  if (cleanStr1.length !== cleanStr2.length) {
   return false;
  }
  const frequencyMap1 = {};
  const frequencyMap2 = {};
  for (const char of cleanStr1) {
   frequencyMap1[char] = (frequencyMap1[char] || 0) + 1;
  }
  for (const char of cleanStr2) {
   frequencyMap2[char] = (frequencyMap2[char] || 0) + 1;
  }
  for (const key in frequencyMap1) {
   if (frequencyMap1[key] !== frequencyMap2[key]) {
    return false;
   }
  }
  return true;
}
const string1 = prompt("Enter String1 : ");
const string2 = prompt("Enter String2 : ");
if (areAnagrams(string1, string2)) {
  console.log(`${string1} and ${string2} are anagrams.`);
} else {
  console.log(`${string1} and ${string2} are not anagrams.`);
}
```

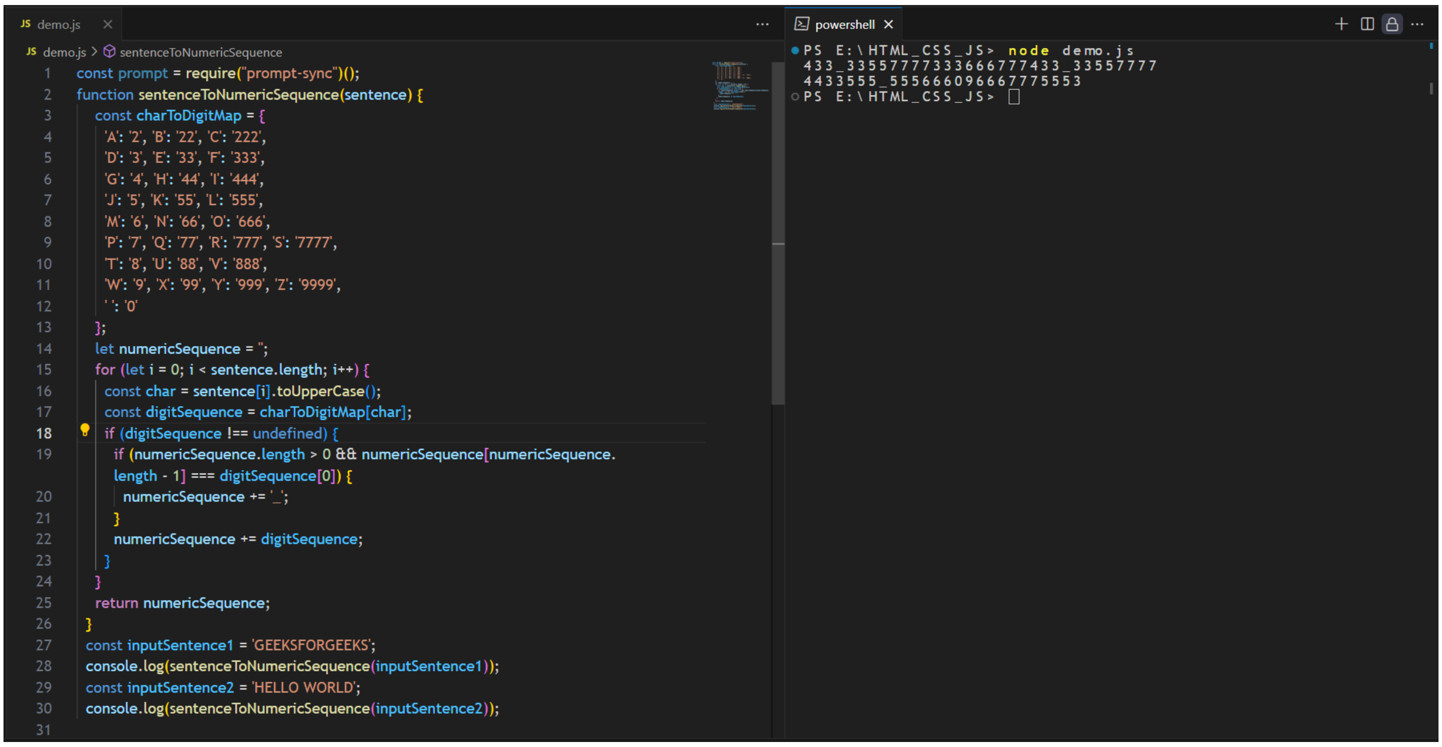
```
PS E:\HTML_CSS_JS> node demo.js
Enter String1 : Listen
Enter String2 : Silent
Listen and Silent are anagrams.
PS E:\HTML_CSS_JS> node demo.js
Enter String1 : Gram
Enter String2 : arm
Gram and arm are not anagrams.
PS E:\HTML_CSS_JS>
```

String Question 30:

Convert a sentence into its equivalent mobile numeric keypad sequence

Input: GEEKSFORGEEKS Output: 4333355777733366677743333557777

Input : HELLO WORLD Output : 4433555555666096667775553



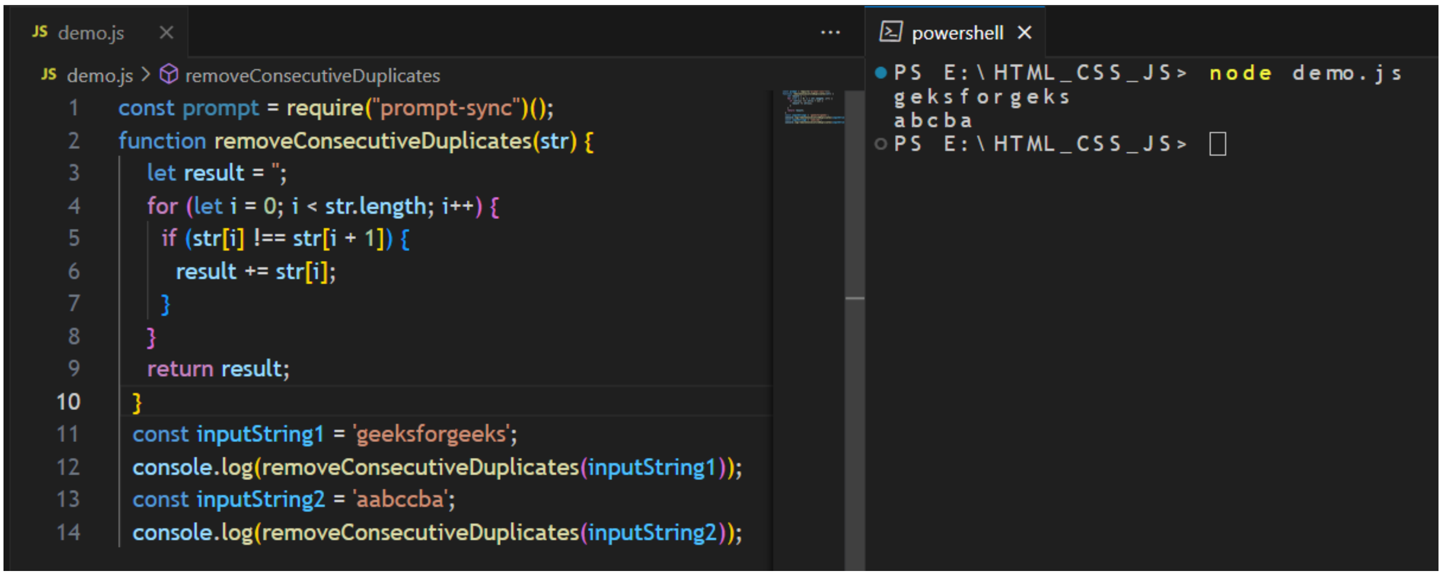
```javascript
const prompt = require("prompt-sync")();
function sentenceToNumericSequence(sentence) {
  const charToDigitMap = {
   'A': '2', 'B': '22', 'C': '222',
   'D': '3', 'E': '33', 'F': '333',
   'G': '4', 'H': '44', 'I': '444',
   'J': '5', 'K': '55', 'L': '555',
   'M': '6', 'N': '66', 'O': '666',
   'P': '7', 'Q': '77', 'R': '777', 'S': '7777',
   'T': '8', 'U': '88', 'V': '888',
   'W': '9', 'X': '99', 'Y': '999', 'Z': '9999',
   ' ': '0'
  };
  let numericSequence = '';
  for (let i = 0; i < sentence.length; i++) {
   const char = sentence[i].toUpperCase();
   const digitSequence = charToDigitMap[char];
   if (digitSequence !== undefined) {
    if (numericSequence.length > 0 && numericSequence[numericSequence.
    length - 1] === digitSequence[0]) {
     numericSequence += '_';
    }
    numericSequence += digitSequence;
   }
  }
  return numericSequence;
}
const inputSentence1 = 'GEEKSFORGEEKS';
console.log(sentenceToNumericSequence(inputSentence1));
const inputSentence2 = 'HELLO WORLD';
console.log(sentenceToNumericSequence(inputSentence2));
```

```
PS E:\HTML_CSS_JS> node demo.js
433_3355777733366677743_33557777
4433555_555666096667775553
PS E:\HTML_CSS_JS>
```

String Question 31:

Input: "geeksforgeeks"   output: geksforgeks

Input: "aabccba"   output: abcba



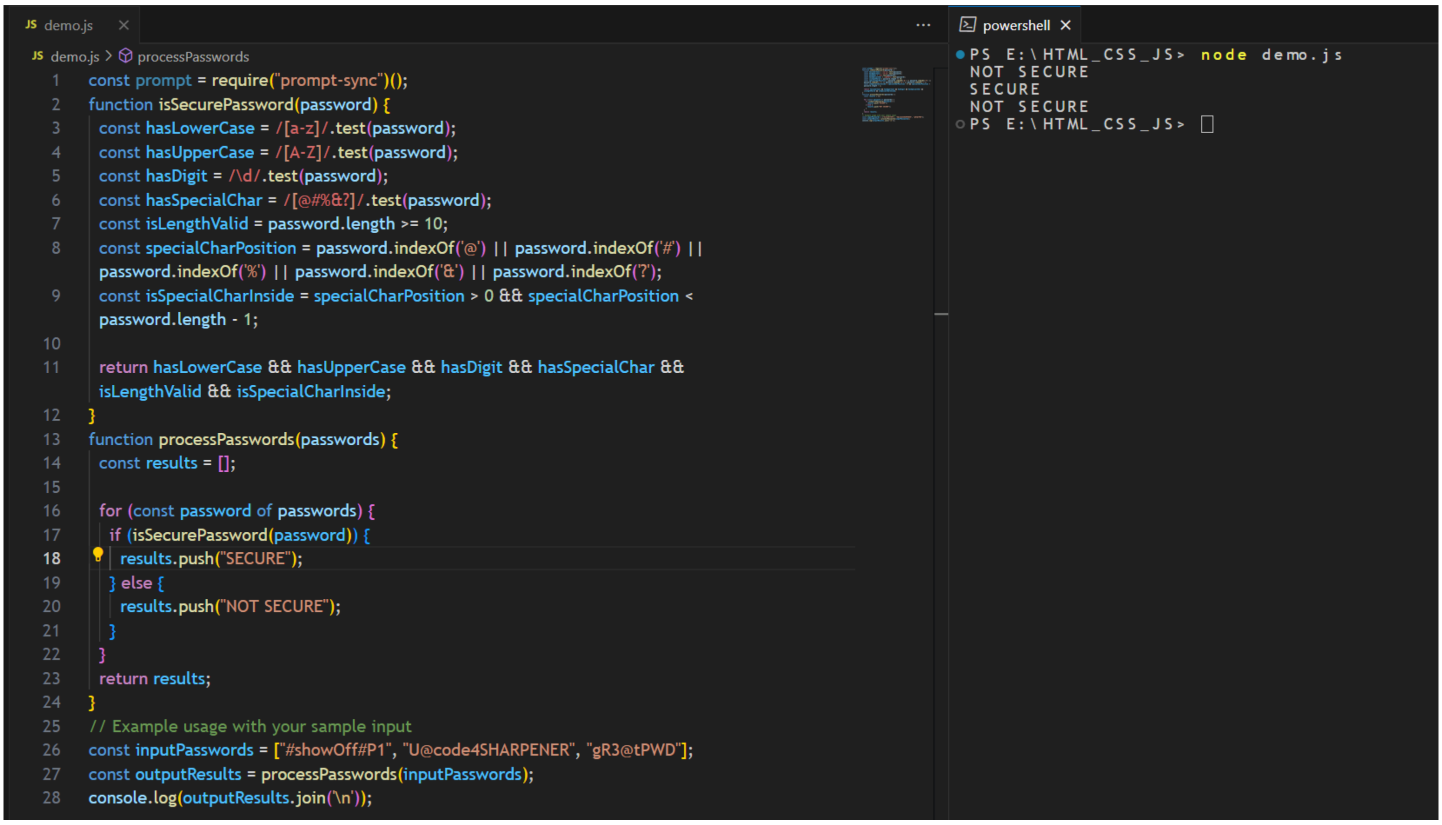
```javascript
const prompt = require("prompt-sync")();
function removeConsecutiveDuplicates(str) {
  let result = '';
  for (let i = 0; i < str.length; i++) {
    if (str[i] !== str[i + 1]) {
      result += str[i];
    }
  }
  return result;
}
const inputString1 = 'geeksforgeeks';
console.log(removeConsecutiveDuplicates(inputString1));
const inputString2 = 'aabccba';
console.log(removeConsecutiveDuplicates(inputString2));
```

```
PS E:\HTML_CSS_JS> node demo.js
geksforgeks
abcba
PS E:\HTML_CSS_JS>
```

String Question 32:

Input: 3   #showOff#P1   U@code4SHARPENER   gR3@tPWD

Sample Output: NOT SECURE   SECURE   NOT SECURE



```javascript
const prompt = require("prompt-sync")();
function isSecurePassword(password) {
  const hasLowerCase = /[a-z]/.test(password);
  const hasUpperCase = /[A-Z]/.test(password);
  const hasDigit = /\d/.test(password);
  const hasSpecialChar = /[@#%&?]/.test(password);
  const isLengthValid = password.length >= 10;
  const specialCharPosition = password.indexOf('@') || password.indexOf('#') || password.indexOf('%') || password.indexOf('&') || password.indexOf('?');
  const isSpecialCharInside = specialCharPosition > 0 && specialCharPosition < password.length - 1;

  return hasLowerCase && hasUpperCase && hasDigit && hasSpecialChar && isLengthValid && isSpecialCharInside;
}
function processPasswords(passwords) {
  const results = [];

  for (const password of passwords) {
    if (isSecurePassword(password)) {
      results.push("SECURE");
    } else {
      results.push("NOT SECURE");
    }
  }
  return results;
}
// Example usage with your sample input
const inputPasswords = ["#showOff#P1", "U@code4SHARPENER", "gR3@tPWD"];
const outputResults = processPasswords(inputPasswords);
console.log(outputResults.join('\n'));
```

```
PS E:\HTML_CSS_JS> node demo.js
NOT SECURE
SECURE
NOT SECURE
PS E:\HTML_CSS_JS>
```