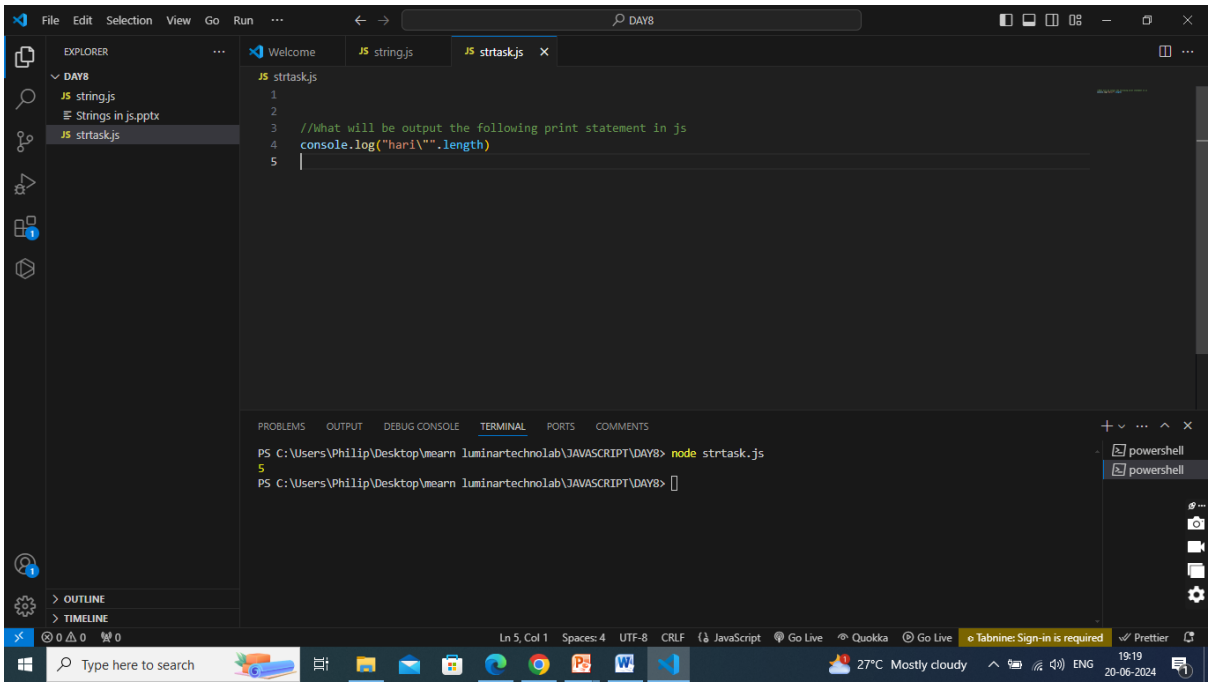
TASK

1. What will be output the following print statement in js console.log("hari\"".length)



2. Explore the includes, startswith and endswith function of a string.

Startswith:

- The startsWith() method returns true if a string starts with a specified string.
- Otherwise it returns false.
- The startsWith() method is case sensitive.

Endswith:

- The endsWith() method returns true if a string ends with a specified string.
- Otherwise it returns false.
- The endsWith() method is case sensitive.

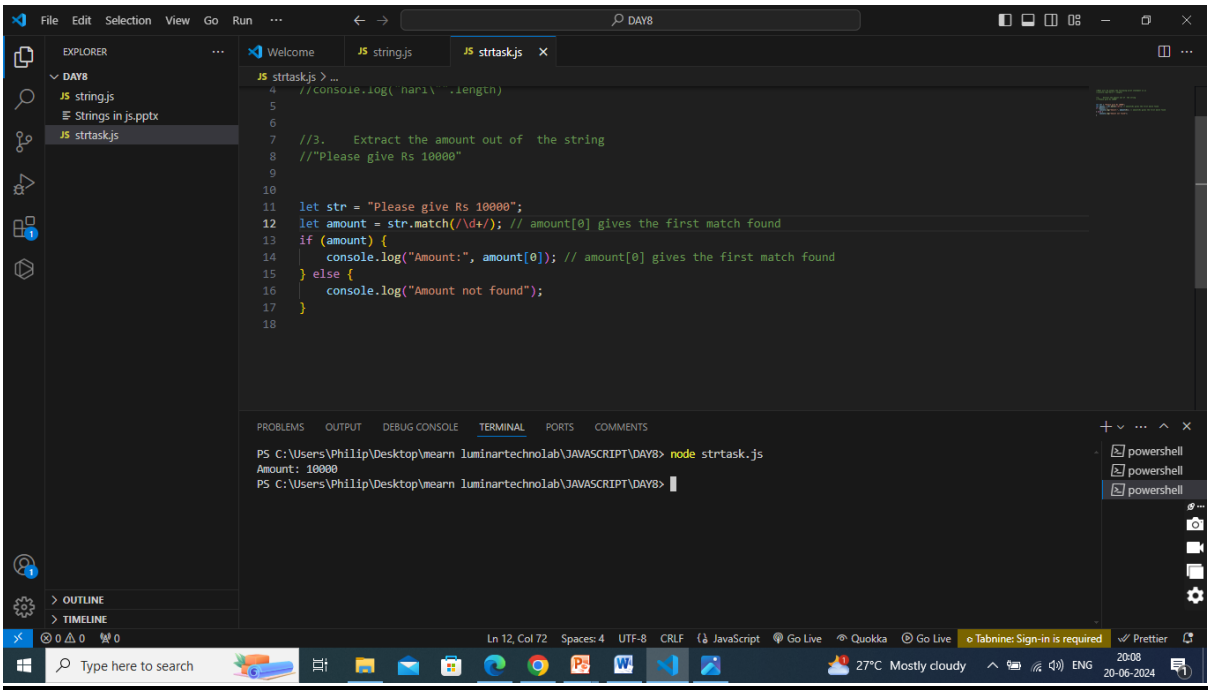
Includes

- The includes() method returns true if a string contains a specified string.
- Otherwise it returns false.
- The includes() method is case sensitive.

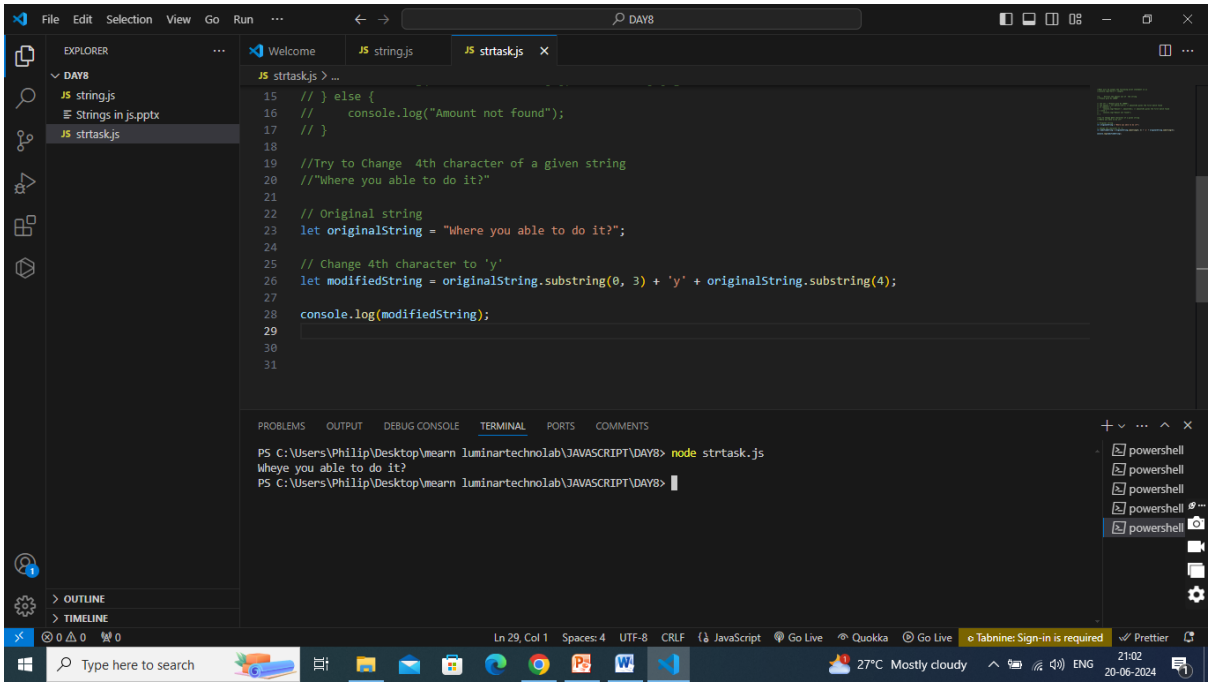
Syntax:

string.includes(*searchvalue*, *start*)

3. Extract the amount out of the string
"Please give Rs 10000"



4. Try to Change 4th character of a given string
"Where you able to do it?"



5. Note on All string methods with programs

1. length The length property returns the length of a string. let str = "Hello, world!"; console.log(str.length); // Output: 13

2. charAt() The charAt() method returns the character at a specified index. let str = "Hello, world!"; console.log(str.charAt(0)); // Output: H

3. charCodeAt() The charCodeAt() method returns the Unicode value of the character at a specified index. let str = "Hello, world!"; console.log(str.charCodeAt(0)); // Output: 72

4. concat() The concat() method concatenates two or more strings. let str1 = "Hello"; let str2 = "world"; console.log(str1.concat(", ", str2, "!")); // Output: Hello, world!

5. includes() The includes() method checks if a string contains a specified substring. let str = "Hello, world!"; console.log(str.includes("world")); // Output: true

6. indexOf() The indexOf() method returns the index of the first occurrence of a specified substring. let str = "Hello, world!"; console.log(str.indexOf("world")); // Output: 7

7. lastIndexOf() The lastIndexOf() method returns the index of the last occurrence of a specified substring. let str = "Hello, world! Hello again!"; console.log(str.lastIndexOf("Hello")); // Output: 13

8. slice() The slice() method extracts a part of a string and returns it as a new string. let str = "Hello, world!"; console.log(str.slice(7, 12)); // Output: world

9. substring() The substring() method extracts characters from a string between two specified indices. let str = "Hello, world!"; console.log(str.substring(7, 12)); // Output: world

10. substr() The substr() method extracts a part of a string, starting at a specified index and extending for a given number of characters. let str = "Hello, world!"; console.log(str.substr(7, 5)); // Output: world

11. replace() The replace() method replaces a specified value with another value in a string. let str = "Hello, world!"; console.log(str.replace("world", "JavaScript")); // Output: Hello, JavaScript!

12. toLowerCase() The toLowerCase() method converts a string to lowercase. let str = "Hello, WORLD!"; console.log(str.toLowerCase()); // Output: hello, world!

13. toUpperCase() The toUpperCase() method converts a string to uppercase. let str = "Hello, world!"; console.log(str.toUpperCase()); // Output: HELLO, WORLD!

14. trim() The trim() method removes whitespace from both ends of a string. let str = " Hello, world! "; console.log(str.trim()); // Output: Hello, world!

15. split() The split() method splits a string into an array of substrings. let str = "Hello, world!"; let arr = str.split(", "); console.log(arr); // Output: ["Hello", "world!"]

16. startsWith() The startsWith() method checks if a string starts with a specified substring. let str = "Hello, world!"; console.log(str.startsWith("Hello")); // Output: true

17. endsWith() The endsWith() method checks if a string ends with a specified substring. let str = "Hello, world!"; console.log(str.endsWith("world!")); // Output: true

18. repeat() The repeat() method returns a new string with a specified number of copies of the original string. let str = "Hello!"; console.log(str.repeat(3)); // Output: Hello!Hello!Hello

19. match() The match() method retrieves the matches of a string against a regular expression. let str = "The rain in SPAIN stays mainly in the plain"; let result = str.match(/ain/g); console.log(result); // Output: ["ain", "ain", "ain"]

20. search() The search() method searches a string for a specified value and returns the position of the match. let str = "The rain in SPAIN stays mainly in the plain"; let position = str.search("ain"); console.log(position); // Output: 5

21. padStart() The padStart() method pads the current string with another string until the resulting string reaches the given length. let str = "5"; console.log(str.padStart(4, '0')); // Output: 0005

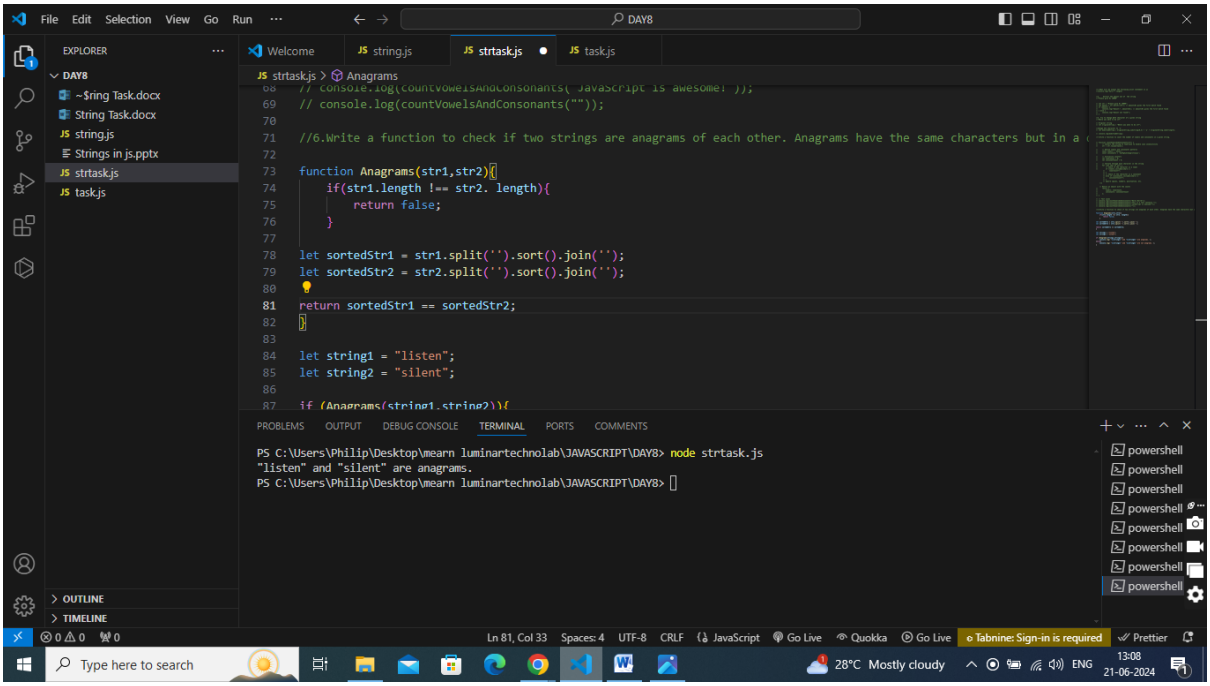
22. padEnd() The padEnd() method pads the current string with another string until the resulting string reaches the given length. let str = "5"; console.log(str.padEnd(4, '0')); // Output: 5000

23. localeCompare() The localeCompare() method compares two strings in the current locale. let str1 = "a"; let str2 = "b"; console.log(str1.localeCompare(str2)); // Output: -1 (means str1 is less than str2)

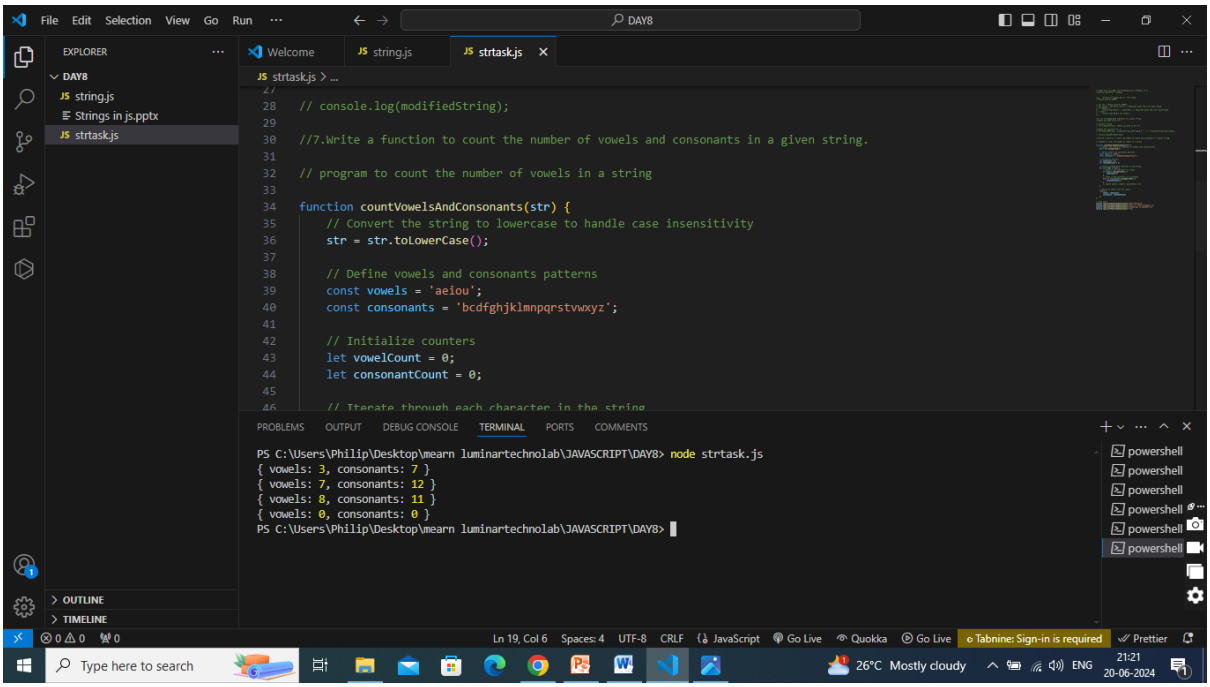
24. fromCharCode() The String.fromCharCode() method returns a string created from the specified sequence of UTF-16 code units. console.log(String.fromCharCode(72, 101, 108, 108, 111)); // Output: Hello

25. raw() The String.raw() method is a tag function of template literals. It returns a string where escape sequences are replaced with their literal counterparts. console.log(String.raw`Hello\nWorld`); // Output: Hello\nWorld (no newline character)

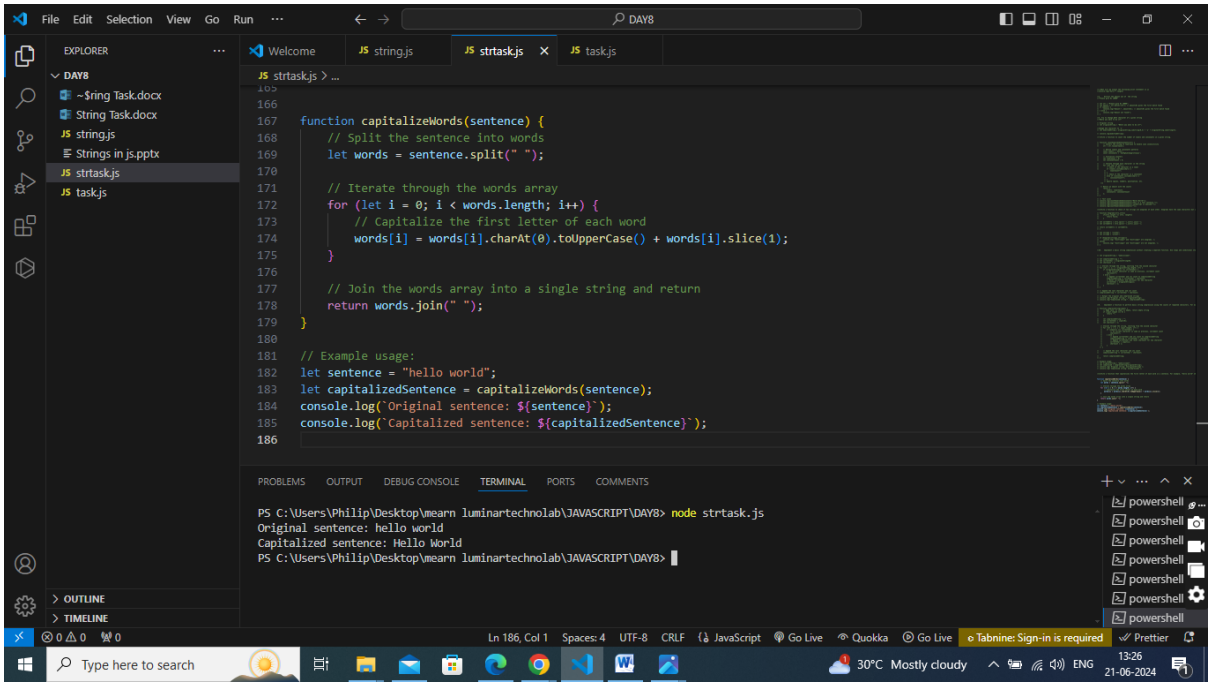
6. Write a function to check if two strings are anagrams of each other. Anagrams have the same characters but in a different order. For example, "listen" and "silent" are anagrams.



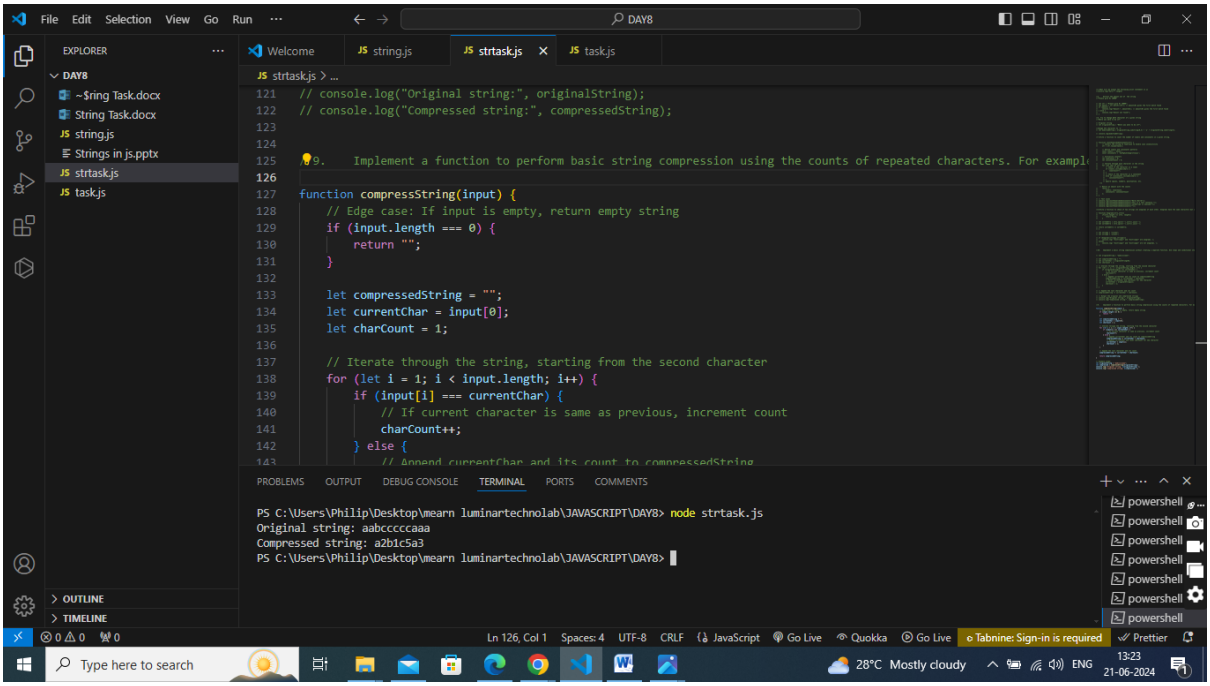
7. Write a function to count the number of vowels and consonants in a given string.



8. Write a function that capitalizes the first letter of each word in a sentence. For example, "hello world" should become "Hello World."



9. Implement a function to perform basic string compression using the counts of repeated characters. For example, the string "aabcccccaaa" would become "a2b1c5a3."



10. Implement a basic string compression without creating a separate function. Use loops and conditional statements to create the compressed string.