Mohan Sai Potla

27 Oct 2022

JavaScript Assignment 16

1). The time has a format: **hours:minutes**. Both hours and minutes have two digits, like 09:00.
Make a regex to find time in the string: **Lunch at 10:10 in the room 123:456.** In this task there's no need to check time correctness yet, so 25:99 can also be a valid result. The regex **should not** match 333:333.

```
//function to check Time format

function checkTime(str) {

    let regex = /\D\d{2}:\d{2}\D/g;

    return regex.test(str);

}


//driver code

let str = "Lunch at 10:10 in03:20hi the room 123:456.";

console.log(checkTime(str));
```

2.) Create a function that finds the word "happiness" in the given string (not case sensitive). If found, return "Hurray!", otherwise return "There is no happiness.".
Example
findHappiness("Work makes me happy") -> There is no happiness.
findHappiness("You give me the feeling of happiness") -> Hurray

```
//function that finds the word "happiness" in the given string (not
case sensitive)

function findHappiness(str) {

    let regex = /happiness/gi;
    let found = regex.test(str);
    if(found){
        return ("Hurray!");
    }else{
        return ("There is no happiness.");
    }
}
```

//driver code

```
let result = findHappiness("You give me the feeling of happiness");
console.log(result);
```

3). Write a regular expression that matches only a prime **number.**
Numbers will be presented as strings.

Example

"7" ➞ true

"134" ➞ false

//function that matches only a prime number

```
function checkPrime(str){
    let num = "1".repeat(str);
    let regex = /^1?$|^(11+?)\1+$/

    let found = regex.test(num);
    if(found){
        return (false);
    }else{
        return (true);
    };
}
```

```
//driver code

let result = checkPrime("3");

console.log(result);
```

4). Create a function that will return an integer number corresponding to the amount of digits in the given integer num

Examples

num_of_digits(1000) ➞ 4

num_of_digits(12) ➞ 2

num_of_digits(1305981031) ➞ 10

```
//function that will return an integer number corresponding to the number of digits in the given integer

function num_of_digits(num) {

    num = num.toString();

    let regex = /[0-9]/g;

    let found = num.match(regex);

    return found.length;

}


//driver code

let result = num_of_digits(1305981031);

console.log(result);
```

5). Create a function that takes in a ***number as a string*** n and returns the number **without trailing and leading zeros**.

- **Trailing Zeros** are the zeros ***after*** a decimal point which ***don't affect the value*** (e.g. the ***last three*** zeros in 3.4000 and 3.04000).
- **Leading Zeros** are the zeros ***before*** a whole number which ***don't affect the value*** (e.g. the ***first three*** zeros in 000234 and 000230).

removeLeadingTrailing("230.000") ➞ "230"

removeLeadingTrailing("00402") ➞ "402"

```
//function that takes in a number as a string n and returns the
number without trailing and leading zeros


function removeLeadingTrailing(str) {
    let regex = /([1-9][0-9]+[.][0-9]+[1-9]+)|([1-9][0-9]+[.][1-
9]+)|([1-9][0-9]+)/g
    let ans = str.match(regex);
    if(ans==null){
        return 0;
    };
    return ans;
}


//driver code
let result = removeLeadingTrailing("00402");
console.log(result);
```

6). Create a function that takes a word and returns true if the word
has two consecutive identical letters.

Examples

doubleLetters("loop") ➞ true

doubleLetters("yummy") ➞ true

```
//function that takes a word and returns true if the word has two
consecutive identical letters
function doubleLetters(str) {
    let regex = /([a-z\d])\1/gi;
    let found = regex.test(str);


    if(found){
        return true;
    }
    else{
        return false;
    }
}
```

```
//driver code
let result = doubleLetters("yummy");
console.log(result);
```

7). ATM machines allow 4 or 6 digit PIN codes and PIN codes cannot contain anything but exactly 4 digits or exactly 6 digits. Your task is to create a function that takes a string and returns true if the PIN is valid and false if it's not.

Examples

validatePIN("1234") ➞ true

validatePIN("12345") ➞ false

```
//function to Validate Pin
function validatePIN(str) {
    let regex = /^\d{4}$|^\d{6}$/
    let found = regex.test(str);

    if(found) {
        return true;
    }else{
        return false;
    }
}
```

```
//driver code
let result = validatePIN("12345");
console.log(result);
```

8). Create a function that determines whether a string is a valid hex code. A hex code must begin with a pound key # and is exactly 6 characters in length. Each character must be a digit from 0-9 or an alphabetic character from A-F. All alphabetic characters may be uppercase or lowercase.

Examples

isValidHexCode("#CD5C5C") ➞ true

isValidHexCode("#EAECEE") ➞ true

isValidHexCode("#CD5C&C") ➞ false

```
//function that determines whether a string is a valid hex code

function isValidHexCode(hexcode) {

    let regex = /^#([0-9A-F]){6}$/gi;

    let found = regex.test(hexcode);


    if(found) return true;

    else return false;

}


//driver code

let result = isValidHexCode("#CD5C5F");

console.log(result);
```

9). Create a function that takes an array of numbers and returns "Boom!" if the digit 7 appears in the array. Otherwise, return "there is no 7 in the array".

Examples

sevenBoom([1, 2, 3, 4, 5, 6, 7]) ➞ "Boom!"
// 7 contains the number seven.

sevenBoom([8, 6, 33, 100]) ➞ "there is no 7 in the array"
// None of the items contain 7 within them.

```
//function that takes an array of numbers and returns "Boom!" if
the digit 7 appears in the array

function sevenBoom(arr) {

    let str = arr.join('');

    let regex = /7/;

    let found = regex.test(str);

    if(found) return "Boom!";

    else return "there is no 7 in the array";

}


//driver code

let result = sevenBoom([1, 2, 3, 4, 5, 6,7]);

console.log(result);
```

10). Create a function that takes a string, checks if it has the same number of x's and o's and returns either true or false.

- Return a boolean value (true or false).
- Return true if the amount of x's and o's are the same.
- Return false if they aren't the same amount.
- The string can contain any character.
- When "x" and "o" are not in the string, return true.

Examples

XO("ooxx") ➞ true

XO("xooxx") ➞ false

XO("ooxXm") ➞ true

// Case insensitive.

Notes

- Remember to return true if there aren't any x's or o's.
- Must be case insensitive.

```
//function that takes a string, checks if it has the same number of
x's and o's and returns either true or false

function XO(str) {

    let regForX = /x/gi;

    let regForO = /o/gi;
    let arrOfX = str.match(regForX);
    let arrOfO = str.match(regForO);

    if(arrOfO == null && arrOfX == null){
        return true;
    }else if (arrOfO.length == arrOfX.length){
        return true;
    }else{
        return false;
    }
}
```

//driver code

```
let result = XO("ooOxxxm");
console.log(result);
```