REGEX ASSIGNMENT

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.

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

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

Example

"7" ⟶ true

"134" ⟶ false

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

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"

Notes

- Return a **string**.
- If you get a number with .0 on the end, return the *integer value* (e.g. return "4" rather than "4.0").
- If the number is 0, 0.0, 000, 00.00, etc... return "0".

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

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

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

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.

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.