

## Exercise 4

Create a random number using the expression

```
Math.trunc( Math.random() * 10 ) + 1
```



The above expression creates a random number between 1 and 10.

Create a function that asks the user for a number between 1 and 10 and repeat the question until a correct number is entered.

Help: `prompt( 'Message' )` gives you a popup displaying 'Message' and asks for an input. The return value of the `prompt` function is the input (string).

The solution is given below but don't look at it until you put enough effort to solve the challenge.

Output
JavaScript
HTML
CSS (SCSS)
<div>Run Code</div>



## Solution

Here is the solution.

```
let value = Math.trunc( Math.random() * 10 ) + 1

function guessValue( value ) {
  let guess;
  do {
    guess = prompt( 'Enter your guess (1 - 10):' );
  } while ( guess != value );
}

guessValue(value); //calling the function
```

You can copy paste it in the Javascript tab in the code editor above and run it. The program won't stop until you enter the correct number so keep trying by entering numbers from 1-10.

Using a do-while loop makes sense, because we have to get a guess first to be able to evaluate the condition `guess != value`. Note that the main use case for do-while loops is to validate user input.

## Let's add some more functionality

Your program should now also print one of the following messages once a value is entered:

- “The value you entered is too high! Try entering a lower number!”
- “The value you entered is too low! Try entering a higher number!”
- “Correct! 36 is the right answer.” - here, assume 36 was the number you had to guess
- “Invalid guess! You have to enter a number between 1 and 50.”
- “Invalid format! Only numbers are allowed!”

Hint: You can use the `alert` function to print messages. The `alert(“message here”)` method displays a pop-up box with a specified message and an OK button.

Output

JavaScript

HTML

CSS (SCSS)

Run Code



## Solution

```
function logHintOnGuess( guess, value ) {
  guess = Number.parseInt( guess, 10 );
  if ( Number.isNaN( guess ) ) {
    alert( "Invalid format! Only numbers are allowed!" );
  } else if ( guess < 1 || guess > 10 ) {
    alert( "Invalid guess! You have to enter a number between 1 and 10." );
  } else if ( guess > value ) {
    alert( "The value you entered is too high! Try entering a lower number!" );
  } else if ( guess < value ) {
    alert( "The value you entered is too low! Try entering a higher number!" );
  } else if ( guess == value ) {
    alert( "Correct! " + value + " is the right answer." );
  }
}

let value = Math.trunc( Math.random() * 10 ) + 1

function guessValue( value ) {
  let guess;
  do {
    guess = prompt( 'Enter your guess (1 - 10):' );
    logHintOnGuess( guess, value ); //calling the Hint function here
  } while ( guess != value );
}

guessValue(value); //calling the function
```

You can copy paste it in the Javascript tab in the code editor above and run it.

You already know everything to read this function. I will still explain the thought process behind writing it.

- First of all, we have to convert the guess to an integer to see if the format is correct.
- We have to swap the order of the messages when checking them in the if-else if-else structure. Because when the guess is not a number, we have to display an invalid format message.
- Then we check if our guess is outside the specified boundaries. In general, checking the error cases first ensures that after all the errors are covered, we can assume that the input is correct
- The order of the low guess, high guess, and exact guess message does not matter. However, notice that in the very last else branch, the presence of the condition is optional. We could comment out the `guess == value` condition.