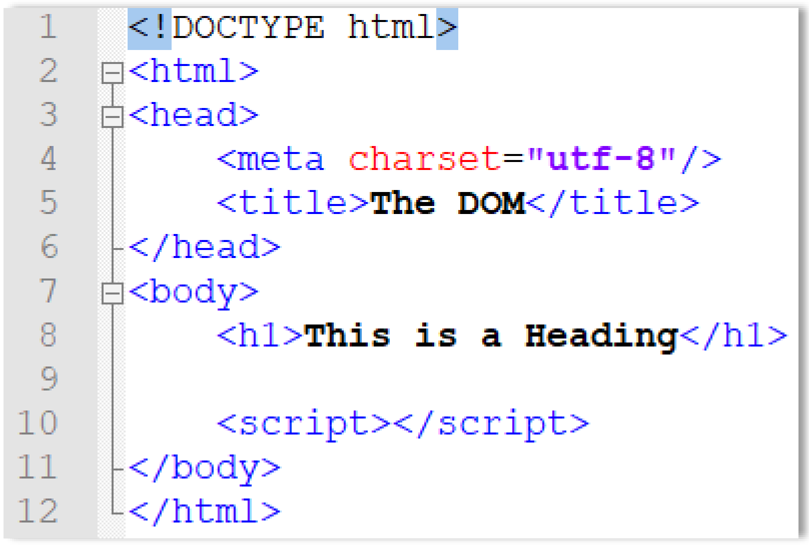
JavaScript: The Document Object Model (DOM)

IMPORTANT! Save all your work to a safe location such as OneDrive.

Create a folder for GUI & Web Development into which you will save all your work for this module, arranged how you wish. Ideally you should create a folder each week for your lab exercises. Note that you should create **a separate html and js file** for each exercise.

Exercise 1:

1. Create the following webpage:



```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8"/>
5      <title>The DOM</title>
6  </head>
7  <body>
8      <h1>This is a Heading</h1>
9
10     <script></script>
11 </body>
12 </html>
```

2. Save and run this page to confirm it works.

3. In this exercise we will change the content of the <h1> element, line 8 above, using the DOM. We will need to assign the <h1> an ID so that we have a way of specifying that this is the HTML element that we want to change:

```
<h1 id="myHeading">This is a Heading</h1>
```

4. To change this using the DOM, we will need to target this element using the following below. Ensure you use the correct ID in the code above! You are targeting the <h1> element on your page, so use the id you added to the <h1> element.

```
document.getElementById("ID_Goes_Here").innerHTML
```

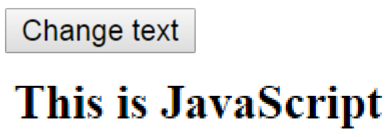
5. This line above is known as a DOM selector. It is selecting the inner content (innerHTML) of the <h1>, so to change the content, we will specify what the content will become:

```
document.getElementById("ID_Goes_Here").innerHTML = "This has changed!";
```

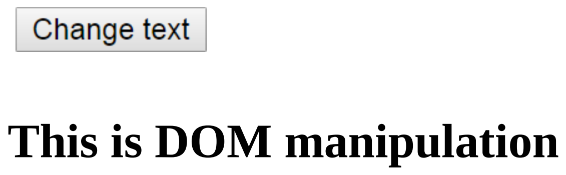
6. Reload your webpage in the browser. The content of the <h1> element should have changed.

Exercise 2:

1. Create a web page that has a heading and a button as shown below.

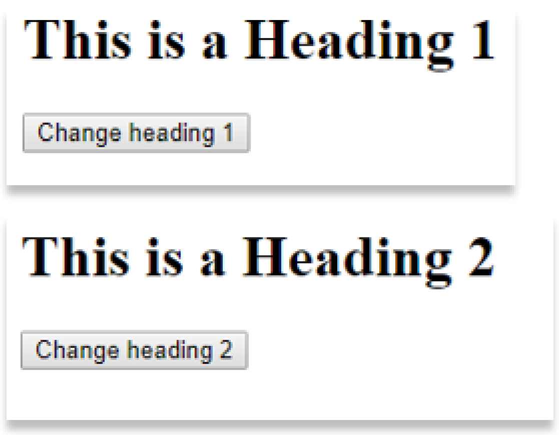


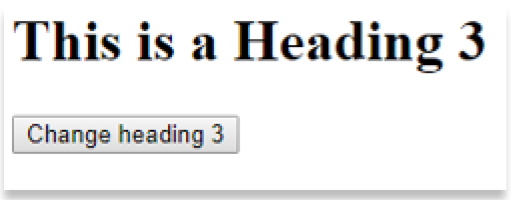
2. When the button is clicked the heading should change text as shown below:



3. Create a function to perform this change using the DOM. Assign this function to the onclick event for the button - In other words, when the button is clicked, the heading should change to text as shown.

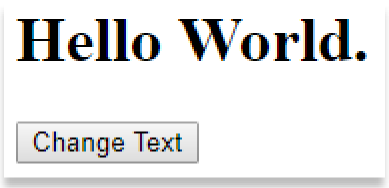
4. Update the webpage as shown below. When button “Change heading 1” is clicked, the heading text should change to read “Heading 1 changed!”. Code your page so that clicking the other buttons causes a similar change on the appropriate heading.



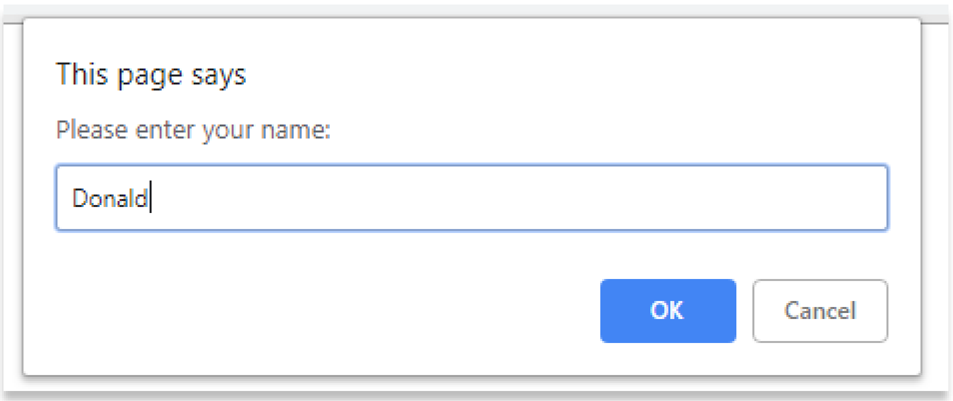

This is a Heading 3

Exercise 3:

1. Create a webpage as shown below:


Hello World.

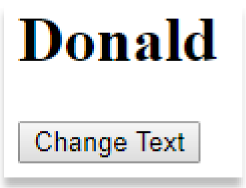
2. When the user clicks on the button a prompt should appear:


This page says

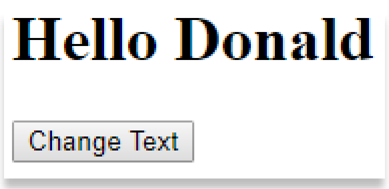
Please enter your name:

Donald

3. After the user enters their name and presses OK, the page will update so that whatever the user entered is now displayed on the page instead of "Hello World." Text.

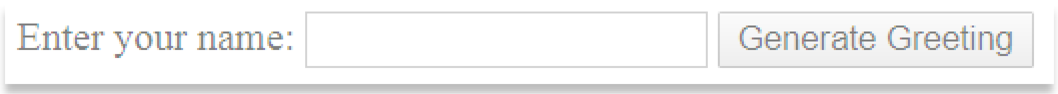

Donald

4. Amend your page so that the output is similar to:

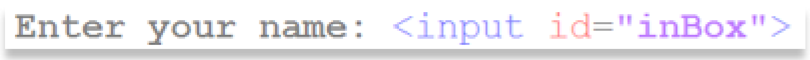

Hello Donald

Exercise 4:

1. Create a webpage as shown:



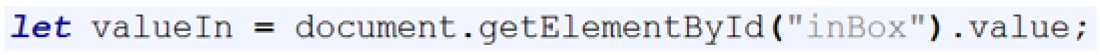
2. The input box can be created with the code provided here:



```
Enter your name: <input id="inBox">
```

3. Add an onclick attribute to the Generate Greeting button and assign a function named getInput to it.

4. The getInput function should read the value typed into the input box, and generate an alert message which contains a greeting. For example, if you type in John the alert message should contain the message “Hello John”. To read a value from the input box you can use the following code:



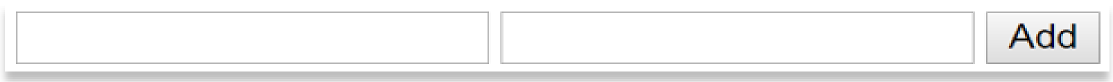
```
let valueIn = document.getElementById("inBox").value;
```

5. Amend the webpage so it now contains a <h1> element.

6. Update the getInput function so that the greeting is written to this <h1> element instead of using the alert.

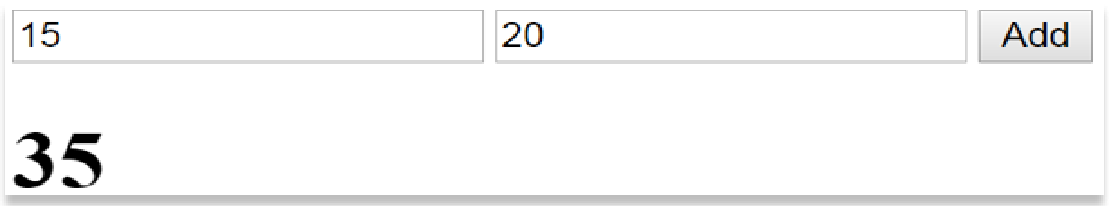
Exercise 5:

1. Create a webpage as shown below:



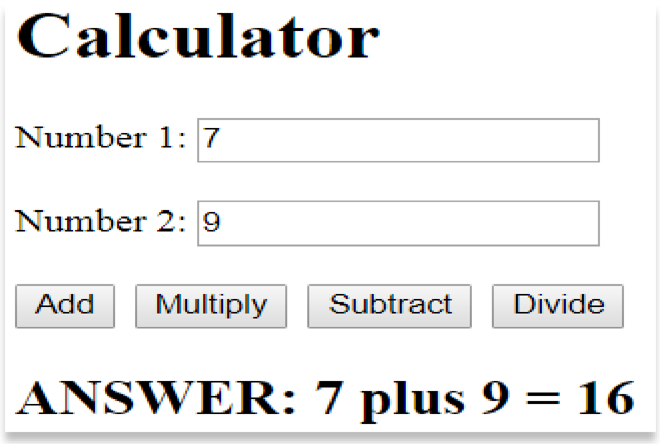
2. Write JavaScript code which will read the two values typed into the input fields when the Add button is clicked and display the result of adding those two values together in an alert. Remember to convert your values to numbers to avoid String concatenation.

3. Amend your webpage so that the result is written to a <h1> element instead of using an alert. See below for an example:



Exercise 6:

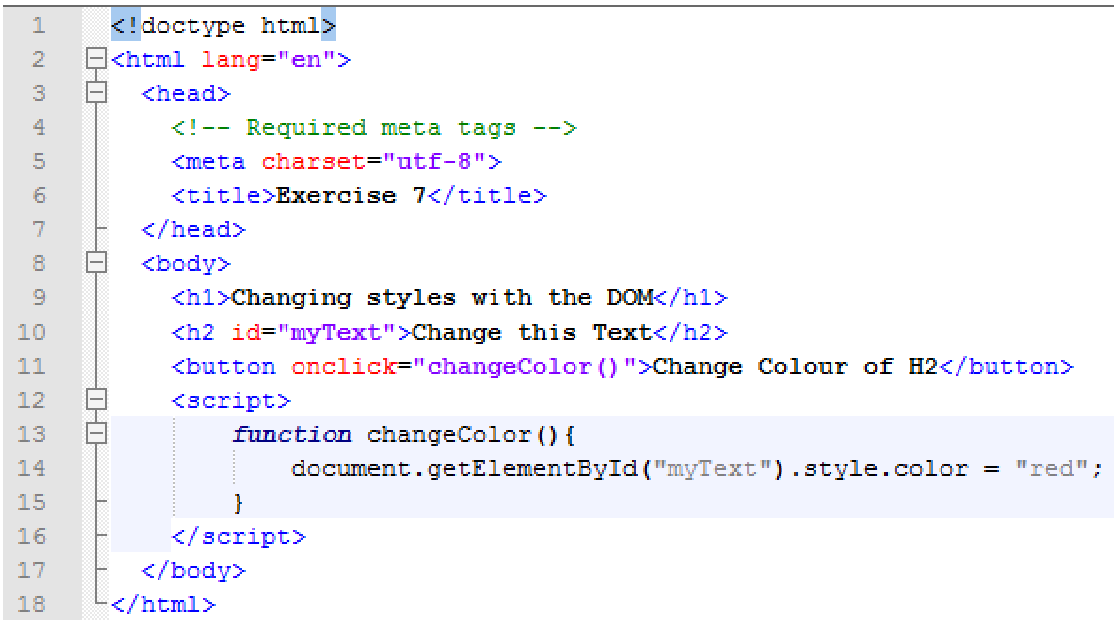
1. Recreate the page shown below:



2. Using the DOM and functions, implement a calculator which will read the numbers typed into the input fields and perform an appropriate calculation based on the button clicked. The result should be written below the buttons using an appropriate format, similar to as shown in the screen shot.

Exercise 7:

1. In this exercise we will change the styles of some HTML elements using the DOM. Create a page like the one shown below and add buttons for each change:

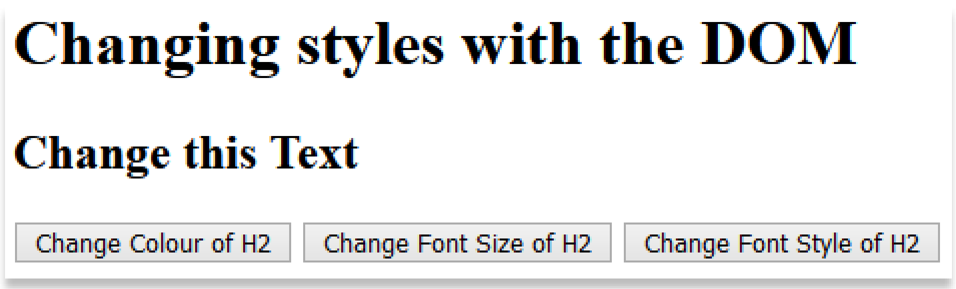


```html
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <title>Exercise 7</title>
  </head>
  <body>
    <h1>Changing styles with the DOM</h1>
    <h2 id="myText">Change this Text</h2>
    <button onclick="changeColor()">Change Colour of H2</button>
    <script>
        function changeColor(){
            document.getElementById("myText").style.color = "red";
        }
    </script>
  </body>
</html>
```

Note: No need for an external JavaScript file in this exercise. You can embed your code in the webpage as shown above.

2. Save your changes and load the page into a browser.

3. Test that when you click the button the colour of the level heading 2 text is changed to red.
4. Amend the page so that it includes two more buttons – One for “Change Font Size of H2” and another for “Change Font Style of H2” as shown below:



Changing styles with the DOM

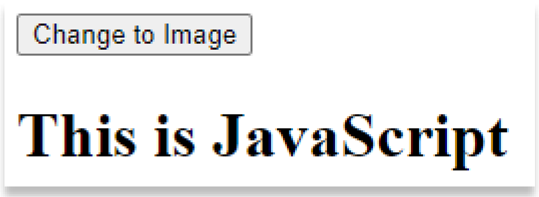
Change this Text

5. Create appropriate functions which run when these buttons are clicked to update the style as suggested by the button labels. You can refer to the following page to help with this:

   https://www.w3schools.com/jsref/dom_obj_style.asp

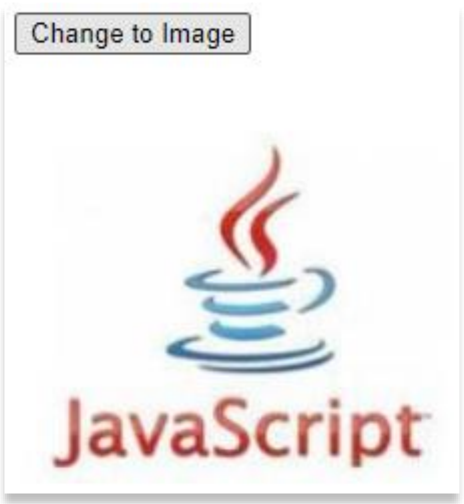
Exercise 8:

Create a web page that has text and a button as shown below:



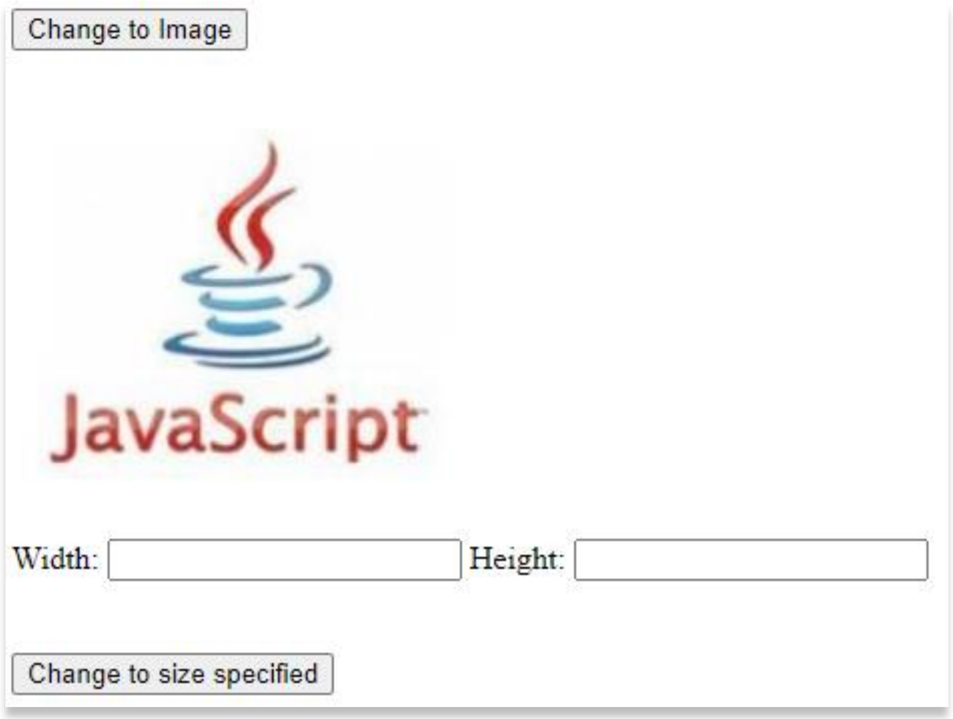
This is JavaScript

When the button is clicked, it should change to an image, as shown below:



Note: A copy of this image is available on Moodle along with this lab – see images.zip. Download this file and unzip it to access the image(js.jpg).

Amend the current webpage, so that the user can specify the height and width of the image. See here for info: https://www.w3schools.com/jsref/prop_style_width.asp. Example of web page is shown below:



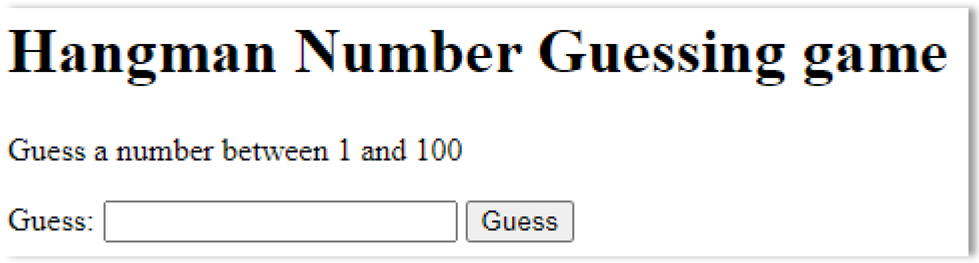
Note: Values should be specified as pixels(px).

Exercise 9:

1. Create a HTML page with a <ul> that lists the 12 months of the year. Add a button with the text "Seasons". This button should trigger a JavaScript function that highlight the months of each season in a different colour (e.g. all spring months written in blue, all summer months written in yellow, etc.)
2. Add a button that will reset the styling of the elements in the list to default.

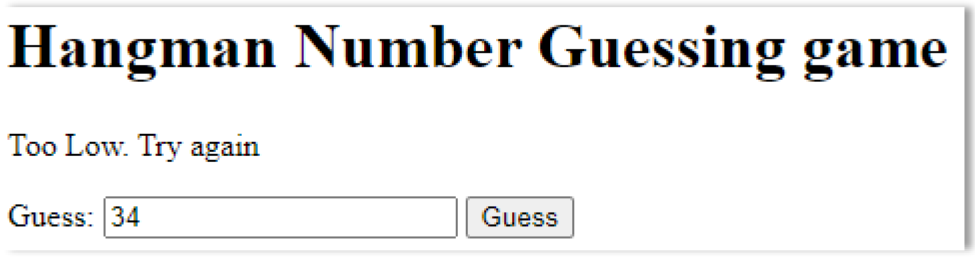
Exercise 10:

1. Create a web page similar to as shown below:


Hangman Number Guessing game

Guess a number between 1 and 100

The game: The program will generate a random number between 1 and 100. The user has 6 guesses to guess the number. After each incorrect guess, the user is notified whether the guess is too high or too low. For example:

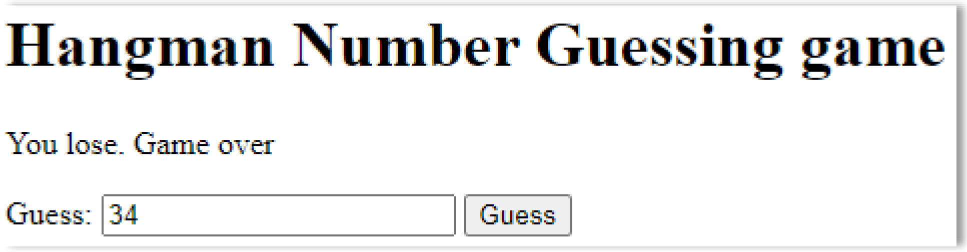


Hangman Number Guessing game

Too Low. Try again

Guess: 34

This continues until either the user guesses correctly, or they run out of goes:



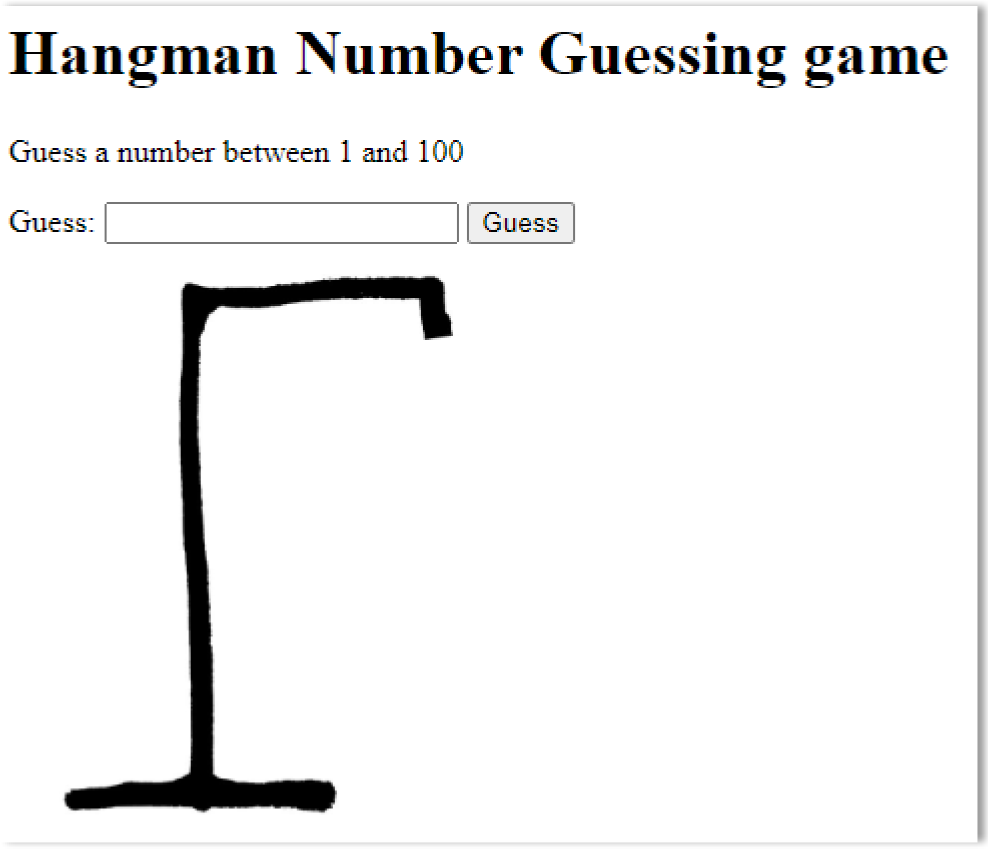
Hangman Number Guessing game

You lose. Game over

Guess: 34

2. Amend the code so that an image is displayed below the input field and button. After every incorrect guess, update the image. The images are available in the images.zip file on Moodle along with this lab. See examples as follows:

<u>Start:</u>

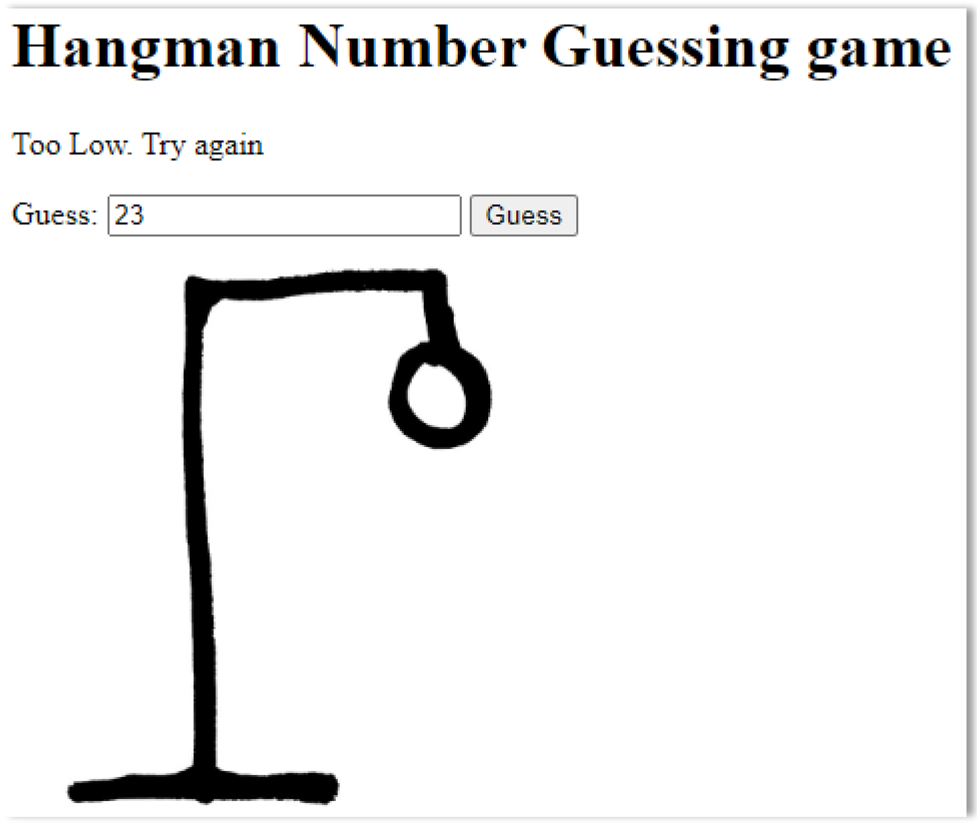


Hangman Number Guessing game

Guess a number between 1 and 100

Guess:

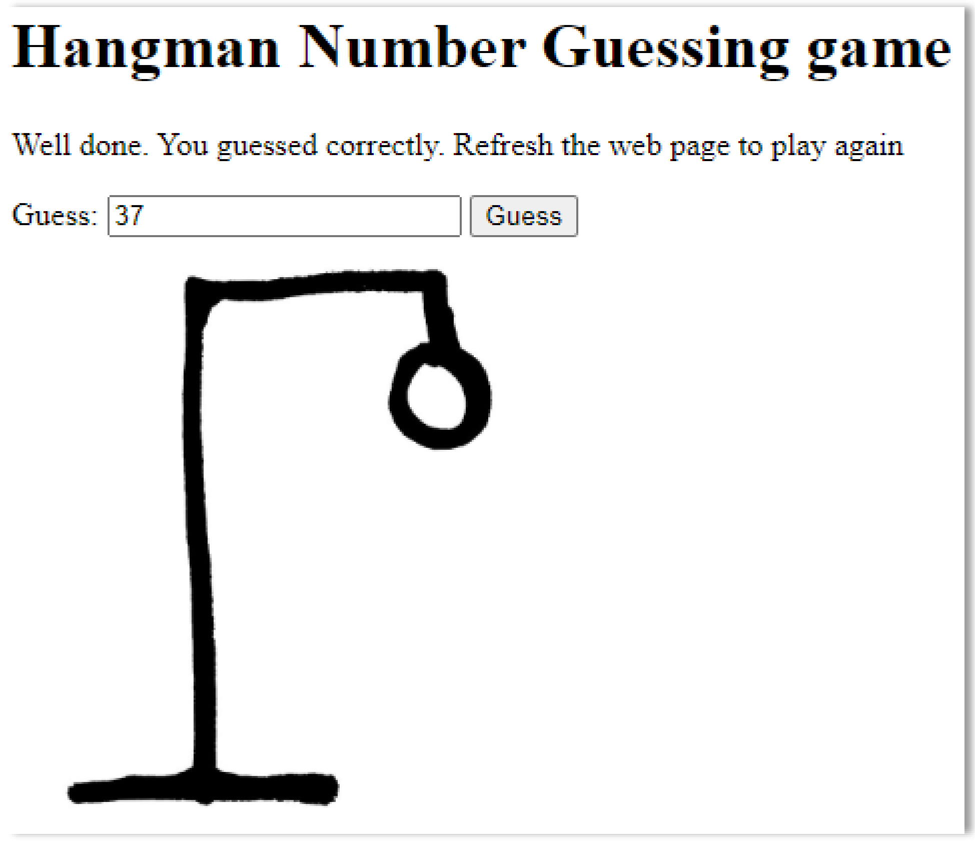
First Guess (incorrect):


Hangman Number Guessing game

Too Low. Try again

Guess: 23 Guess

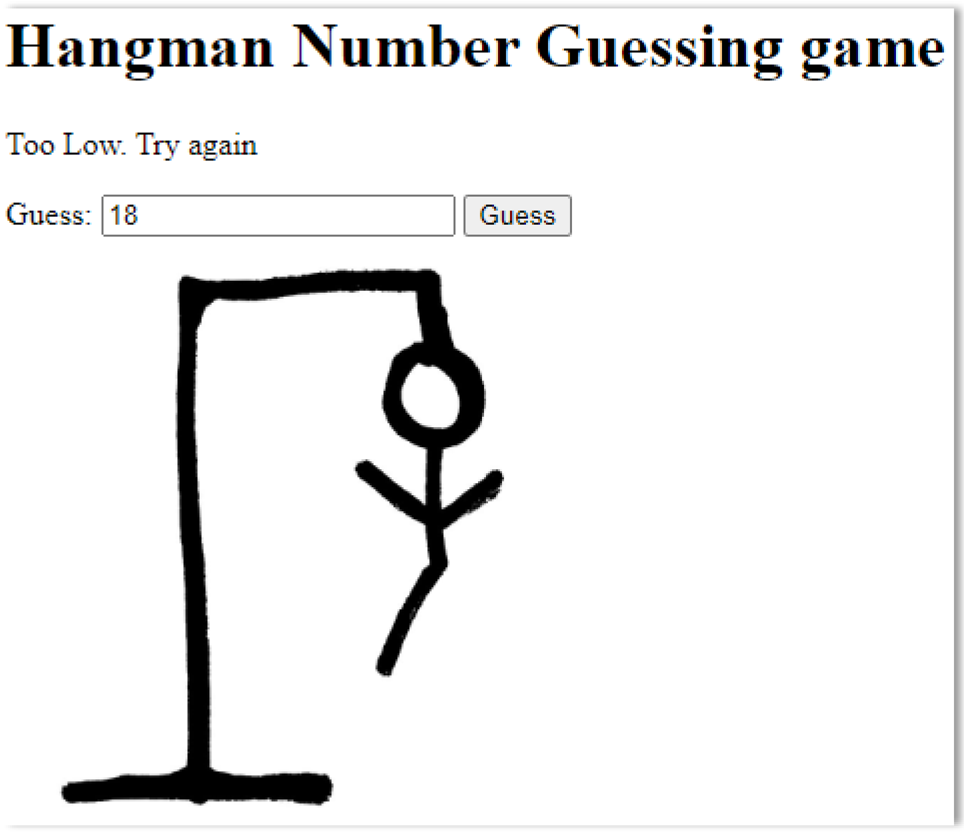
Second Guess (correct):


Hangman Number Guessing game

Well done. You guessed correctly. Refresh the web page to play again

Guess: 37 Guess

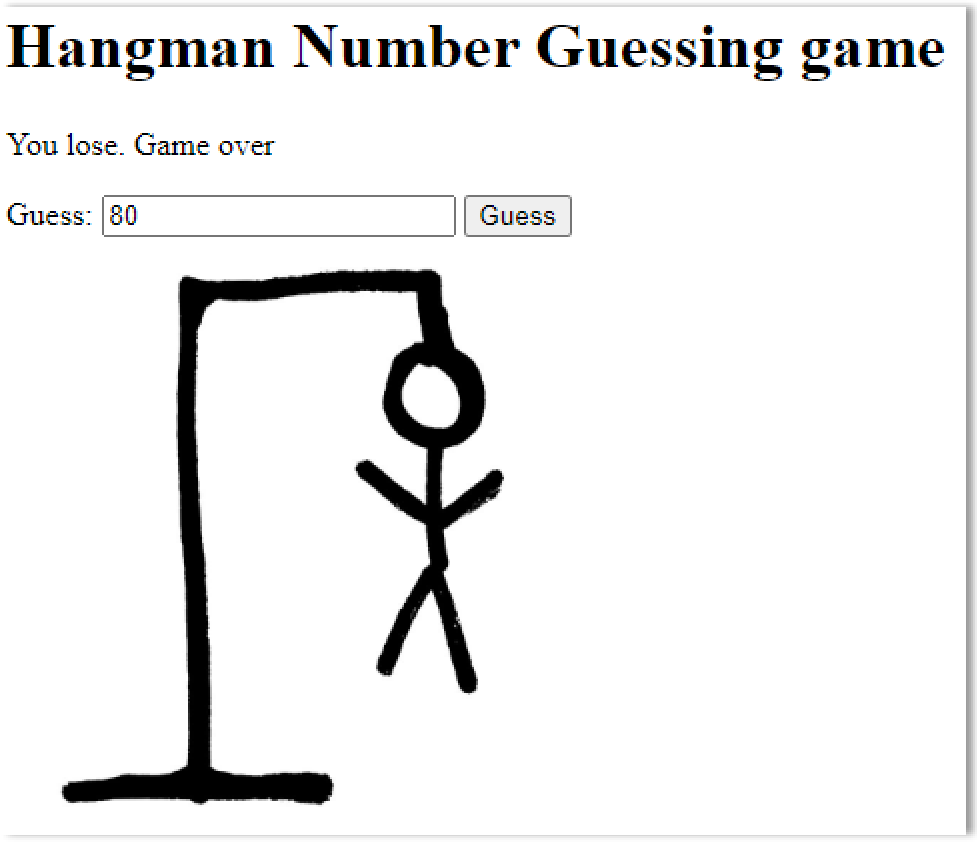
<u>Play another game, at 5th guess (incorrect):</u>



Hangman Number Guessing game

Too Low. Try again

<u>Now at sixth guess (incorrect):</u>

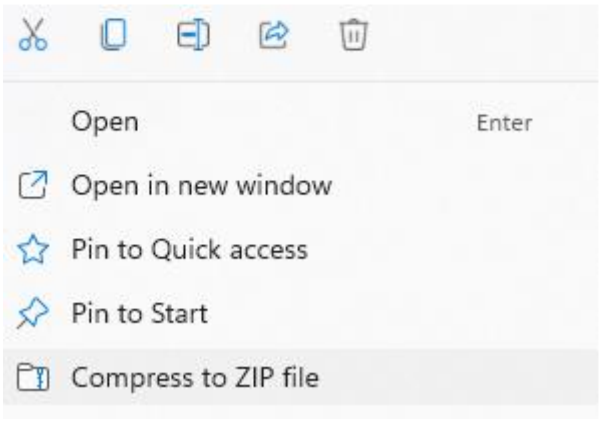


Hangman Number Guessing game

You lose. Game over

Upload your work to Moodle

1. Navigate to the location of the folder where you saved all your work for today's lab.
2. Right-click on the folder and select "Compress to ZIP file". This will create a compressed version of any files you have worked on for the lab.



3. There should be a new compressed file created. This is the file that you will need to upload to Moodle.
4. In Moodle, navigate to the "Submissions" section, and click the ***upload*** link for current lab.
5. Click "add submission" and add the ZIP file you created here. Make sure you complete the submission process. Your lab work has been submitted.