

Non-mandatory exercise 5 – HTML and CSS

This exercise is non-mandatory (you are not supposed to deliver any answers), but it is assumed that you complete the exercise as preparation for the mandatory test (in a few weeks).

Part 1 – read and understand HTML, CSS and JavaScript

In this part you are going to read and understand HTML, CSS and JavaScript.

There are no errors in the code (if it is not specified). I.e., you don't have to look for errors 😊.

Task 1 – Common HTML-elements

Explain what is shown on the webpage in the example below. Assume that the HTML is written inside the body-tag in a webpage and also that the image-file exists in the correct folder:

```
<h1>The red hamster</h1>

<p>The red hamster is a sneaky old fellow. Lorem ipsum, dolor sit amet consectetur adipisicing elit. Labore, dicta?</p>
<p>The hamsters main weapon is it's big harpoon-like horn. Lorem Voluptates harum aut et provident vero eos odit quibusdam, consequuntur debitis magnam?</p>
```

What will be shown in the example above if the image-file doesn't exist – or is in the wrong folder?

What will be shown on the webpage in these examples:

```
<p>Dog</p>
<p>Cat</p>
<p>Tiger</p>

<ul>
  <li>Dog</li>
  <li>Cat</li>
  <li>Tiger</li>
</ul>

<div>
  <p>Dog</p>
  <p>Cat</p>
  <p>Tiger</p>
</div>
```

```
<div>
  <span>Dog</span>
  <span>Cat</span>
  <span>Tiger</span>
</div>

<div>
  <span>Dog</span><span>Cat</span><span>Tiger</span>
</div>
```

```
<canvas
  id="cnv"
  width="400"
  height="300"
  style="border: solid black 1px;">
</canvas>
```

Task 2 – UI-elements

Explain what is shown on the webpage in the example below. Assume that the HTML is written inside the body-tag in a webpage:

```
<span>Weight in kg: </span>
<input type="text" id="inpWeight" value="60">
<span>Height in cm: </span>
<input type="text" id="inpHeight" value="170">
<button id="btnCalculate">Calculate BMI</button>
```

Using a <label>-tag instead of a -tag gives the same visual result, but what is the difference in user experience? Tip: check out the *for*-attribute:

```
<label for="inpWeight">Weight in kg: </label>
<input type="text" id="inpWeight" value="60">
<label for="inpHeight">Height in cm: </label>
<input type="text" id="inpHeight" value="170">
<button id="btnCalculate">Calculate BMI</button>
```

What does the *number*-type do?

```
<label for="inpWeight">Weight in kg: </label>
<input type="number" id="inpWeight" value="60">
<label for="inpHeight">Height in cm: </label>
<input type="number" id="inpHeight" value="170">
<button id="btnCalculate">Calculate BMI</button>
```

What about the *range*-type?

```
<label for="inpRadius: "></label>
<input id="inpRadius" type="range" min="0" max="200" value="20" />
```

Tip: You can check out the the different input-types at [w3schools.com](https://www.w3schools.com/html/html_form_elements.asp):

The different input types are as follows:

- `<input type="button">`
- `<input type="checkbox">`
- `<input type="color">`
- `<input type="date">`
- `<input type="datetime-local">`
- `<input type="email">`
- `<input type="file">`
- `<input type="hidden">`
- `<input type="image">`
- `<input type="month">`

- `<input type="number">`
- `<input type="password">`
- `<input type="radio">`
- `<input type="range">`
- `<input type="reset">`
- `<input type="search">`
- `<input type="submit">`
- `<input type="tel">`
- `<input type="text">` (default value)
- `<input type="time">`
- `<input type="url">`
- `<input type="week">`

What does the <fieldset> and <legend>-tag do?

```
<fieldset>
  <legend>Check your BMI</legend>
  <label for="inpWeight">Weight in kg: </label>
  <input type="number" id="inpWeight" value="60">
  <label for="inpHeight">Height in cm: </label>
  <input type="number" id="inpHeight" value="170">
  <button id="btnCalculate">Calculate BMI</button>
</fieldset>
```

What is shown in the webpage here:

```
<select id="slcColors">
  <option value="red">Red</option>
  <option value="orange">Orange</option>
  <option value="purple">Purple</option>
</select>
```

What does the size-attribute do?

```
<select id="slcColors" size="3">
  <option value="red">Red</option>
  <option value="orange">Orange</option>
  <option value="purple">Purple</option>
</select>
```

A <textarea>-tag is an alternative to an <input>-tag. What is the difference?

```
<label for="inpTxt">Write your comment: </label>
<textarea id="inpTxt" cols="30" rows="10"></textarea>
```

In the example below we use a <form>-tag to retrieve different values. Why must we use `evt.preventDefault()` inside the event-handler for the *submit*-event?

```

<form id="frmBMI">
  <label for="inpWeight">Weight in kg: </label>
  <input type="number" name="inpWeight" value="60">
  <label for="inpHeight">Height in cm: </label>
  <input type="number" name="inpHeight" value="170">
  <input type="submit" value="Calculate BMI">
</form>

<script>

  let frmBMI = document.getElementById('frmBMI');

  frmBMI.addEventListener('submit', function(evt) {
    evt.preventDefault();
    let bmiData = new FormData(frmBMI);
    let weight = bmiData.get("inpWeight");
    let height = bmiData.get("inpHeight");

    console.log(weight, height);
  });

</script>

```

Task 3 – create HTML using JavaScript

In the examples below we are using two different methods for creating new HTML-elements. In the first examples we are using *document.createElement* to create all the elements:

```

<div id="container"></div>

<script>

  let container = document.getElementById('container');

  let personData = [
    {name: "Jenny", phone: 55544345},
    {name: "Sue", phone: 5558613},
    {name: "Peter", phone: 5559399},
    {name: "James", phone: 5551111}
  ]

  for (let person of personData) {
    let personDiv = document.createElement('div');
    let h3 = document.createElement('h3');
    let p = document.createElement('p');
    let hr = document.createElement('hr');

    h3.innerHTML = person.name;
    p.innerHTML = "Phone: " + person.phone;
    personDiv.appendChild(h3);
    personDiv.appendChild(p);
    personDiv.appendChild(hr);
    container.appendChild(personDiv);
  }

</script>

```

Method 1

In the example below, we are doing the same thing, but are filling in the div's by setting the innerHTML of the div's using HTML-code. This will often be faster and doesn't require us to write so much code:

```
<div id="container"></div>

<script>

  let container = document.getElementById('container');

  let personData = [
    {name: "Jenny", phone: 55544345},
    {name: "Sue", phone: 5558613},
    {name: "Peter", phone: 5559399},
    {name: "James", phone: 5551111}
  ]

  for (let person of personData) {
    let personDiv = document.createElement('div');
    let html = "<h3>" + person.name + "</h3>";
    html += "<p>Phone: " + person.phone + "</p>";
    html += "<hr>";
    personDiv.innerHTML = html;
    container.appendChild(personDiv);
  }

</script>
```

Method 2

What is shown on the webpages above? (It is the same result in both examples)

Task 4 – CSS selectors

Explain what the following CSS-selectors select on the webpage:

- *
- #fox
- .nice
- article,div
- article.nice
- .nice div
- article .nice
- #fox + p
- p ~ .nice
- *[href]
- img[src="dog.png"]
- img[src\$=".png"]
- button:hover
- input[type=radio]:checked
- .nice:not(p)
- #fox:target

Task 5 – functions (methods) for selecting and manipulating the webpage

Explain the difference between these (DOM)-functions for selecting elements on the webpage:

- `document.getElementById()`
- `document.getElementsByName()`
- `document.querySelector()`
- `document.querySelectorAll()`

Explain what these code-examples do:

```
myDiv.appendChild(myButton);
```

```
myDiv.removeChild(btn3);
```

```
myDiv.insertBefore(btn1, btn2);
```

```
myDiv.replaceChild(fancyButton, btn1);
```

```
let buttons = myDiv.children;
```

```
myDiv.innerHTML = "";
```

```
let containerElm = myButton.parentNode;
```

```
myDiv.style.backgroundColor = "lightgrey";
```

```
fancyButton.style.padding = "5px";
```

```
fancyButton.classList.add("fancy");
```

```
myImg.setAttribute("src", "/images/dog.jpg");
```

```
myImg.src = "/images/dog.jpg";
```

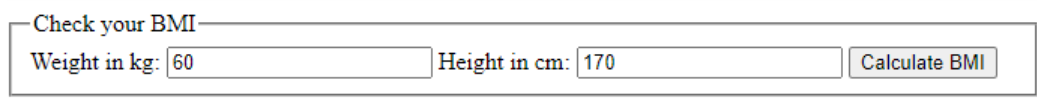
```
window.location.href = "https://uia.no";
```

Part 2 – write code

In this part you are going to write your own HTML, CSS and JavaScript-code

Task 5

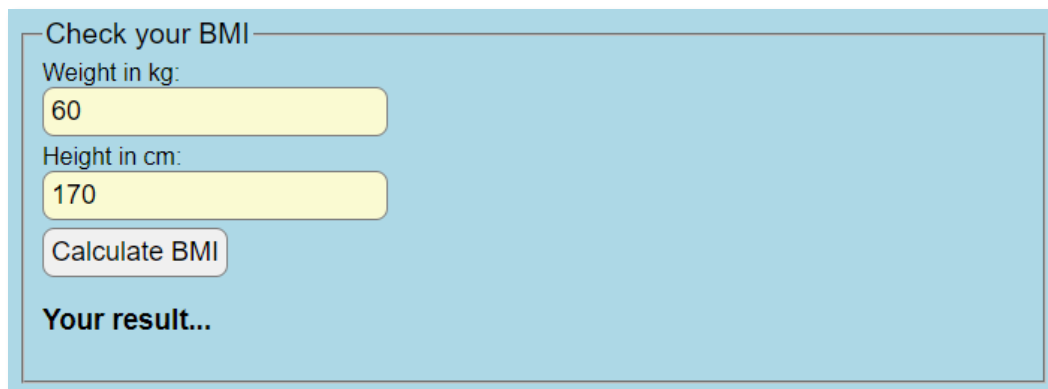
The default visual appearance for the HTML-elements below is this:



A screenshot of a web form titled "Check your BMI". It contains two input fields: "Weight in kg:" with the value "60" and "Height in cm:" with the value "170". To the right of these fields is a button labeled "Calculate BMI".

```
<fieldset>
  <legend>Check your BMI</legend>
  <label for="inpWeight">Weight in kg: </label>
  <input type="number" id="inpWeight" value="60">
  <label for="inpHeight">Height in cm: </label>
  <input type="number" id="inpHeight" value="170">
  <button id="btnCalculate">Calculate BMI</button>
  <h3 id="txtResult">Your result...</h3>
</fieldset>
```

Add CSS-code so that the elements are styled similar to this:

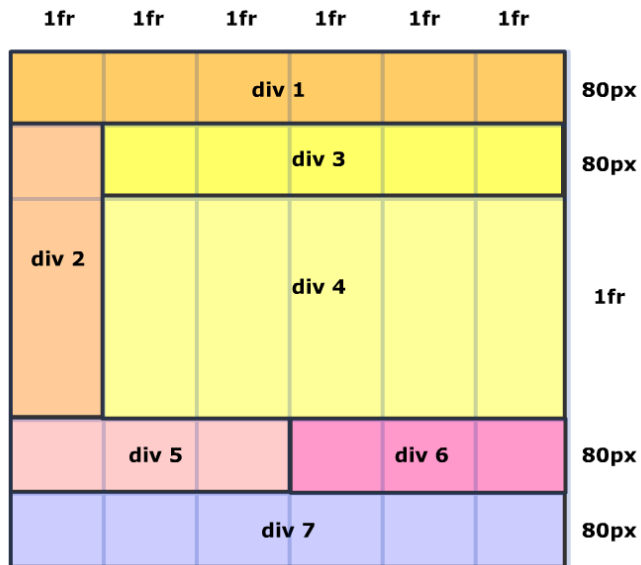


A screenshot of the same BMI calculator form, but with CSS styling. The form is enclosed in a light blue box with a thin border. The title "Check your BMI" is at the top left. Below it, the labels "Weight in kg:" and "Height in cm:" are followed by yellow input fields containing "60" and "170" respectively. A button labeled "Calculate BMI" is below the height field. At the bottom, the text "Your result..." is displayed in a bold font.

NB! Don't change the HTML-code – use only CSS to style the elements.

Task 6

- Use CSS *grid* and *grid-areas* to create a layout like this using <div> elements. Let the divs have different colors and a border so that it easy to see the layout on the screen:

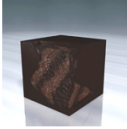


- b) Explain how you would make this layout responsive and adapted for small screens (e.g., smartphones).

Task 7

- a) Start with *chocolist.html*. The program contains an array with information about different chocolates. Write JavaScript-code so that you display a list with the different chocolates on the webpage (use a for-of loop). Let the information for each chocolate be inside a div-element. Tip: use *method 2* in *Task 3* above:


Tiger



Lys sjokolademousse med striper av mocca-mousse og overtrekk av mørk sjokolade

Price kr 12,-


Green ferret



Eple-fondant i mørk melkesjokolade

Price kr 15,-


Afterglow



Appelsinfondant blandet med appelsinlikörgelé og overtrekk av mørk sjokolade

Price kr 21,-

Black mist



- b) Style the chocolate-divs so the list looks similar to the illustration below. Tip: let each chocolate-div be a grid container and place the different elements in the grid:

	Tiger Lys sjokolademousse med striper av mocca-mousse og overtrekk av mørk sjokolade Price kr 12,-
	Green ferret Eple-fondant i mørk melkesjokolade Price kr 15,-
	Afterglow Appelsinfondant blandet med appelsinlikörgelé og overtrekk av mørk sjokolade Price kr 21,-
	Black mist Mocca ganache i mørk melkesjokolade Price kr 15,-
	Black & white Hvit sjokolade med overtrekk av mørk melkesjokolade Price kr 10,-
	Rum & cherry Kirsebær i rom-likør Price kr 16,-

Task 8

Style a button similar to this:



Task 9 - a little challenge

Create two radio-buttons and style them so they look similar to this:



Tip: The visual appearance of radio-buttons and checkboxes is controlled by the browser – not CSS. Even so, we can style them using a *label*-tag. Clicking on a label-tag will be the same as if you clicked on the input it is referring to (using the *for* attribute). Because of this, you can hide the input (the radiobutton) with *display: none* and style the label so that it is “checked” when the input is checked.