EECS 1012. Lab 3: HTML + CSS + JS (September 28– Oct 2, 2020)

A. IMPORTANT REMINDERS

1) Each lab including the pre-lab mini quiz is about 2.0 % of your overall grade.
2) You must attend your own lab session (the one you are enrolled in). If you need to change your lab enrollment, you should go to the department. Instructors or TAs cannot change your enrollment. TAs are available via Zoom to help you, and you are welcome to attend these online lab sessions (the attendance is optional, but is highly recommended). You can also have your work verified and graded during the lab sessions. Feel free to signal a TA for help if you stuck on any of the steps below. Yet, note that TAs would need to help other students too. In case you run out of time, the submission you make over eClass will be marked by the TAs after the lab ends (possibly not by the same TAs who assisted you during the lab session).
3) You can submit your lab work anytime before the deadline. We do not accept late submissions.
4) You must complete the pre-lab mini quiz posted on eClass no later than the first 15 minutes of your lab time.

B. IMPORTANT PRE-LAB WORKS YOU NEED TO DO BEFORE GOING TO THE LAB

1) Download this lab files and read them completely.
2) You should have a good understanding of
    - Events (such as `onclick`, `ondblclick`)
    - `document.getElementById().innerHTML`
    - `document.getElementById().setAttribute()`
    - `document.getElementById().style`
    - `document.getElementById().style.display`

C. GOALS/OUTCOMES FOR LAB

1) To learn how to change the *behaviour* of an HTML document using JavaScript.

D. TASKS

Part 1: Start developing a learning kit that by end of term could include anywhere between 30 to 50 computational problems and solutions. The details are as follows.

1) You are provided a simple `myLearningKit.html` document and supporting files such as `myLearningKit.css` and `myLearningKit.js` as well some images. Your first task will be to improve the presentation of the HTML file slightly. Then, your major task is to augment the *behaviour* of the HTML file by providing `JavaScript` code.
2) You will generate at least five sets of HTML and JS files in this process. If you decide to demo your work to the TA, please have each HTML file open in a different tab of your browser, so you can show the progression.
3) See the following pages for details on how to modify your HTML and JS files.

Part 2: Validate your HTML code using the https://validator.w3.org (upload your .html file)

- If you have errors, fix them and retry the validation until your HTML document is error free.

Part 3. Verification and submission of your work. See the following section.

E. SUBMISSIONS

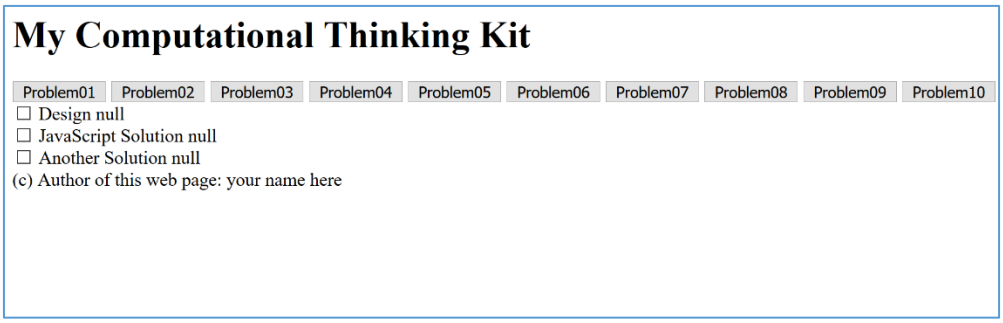
1) Manual verification by a TA (optional)
   You may have one of the TAs verify your lab before submission. The TA will look at your various files in their progression. The TA may also ask you to make minor modifications to the lab to demonstrate your knowledge of the materials. The TA can then record your grade in the system.

2) eClass submission

You will see an assignment submission link on eClass. Create a folder named "Lab03" with all your lab materials inside (image folder, `myLearningKit_Ex{1,2,3,4,5}.html` and `myLearningKit_Ex1.css`, and `myLearningKit_Ex{2,3,4,5}.js`). This folder should be compressed (**zip** or **tar.gz**) and the compressed file submitted.

F. WEBPAGE CONTENT

STARTING POINT: **myLearningKit.html**, **myLearningKit.css**, **myLearningKit.js**, and images in the **image folder**. You are given the following HTML file.



This file is not connected to a CSS file. In Exercise 1, you improve the *presentation* of this HTML document by modifying the HTML and CSS files slightly.

Before proceeding further, open **myLearningKit.HTML** in VS Code and carefully learn about its content and structure.

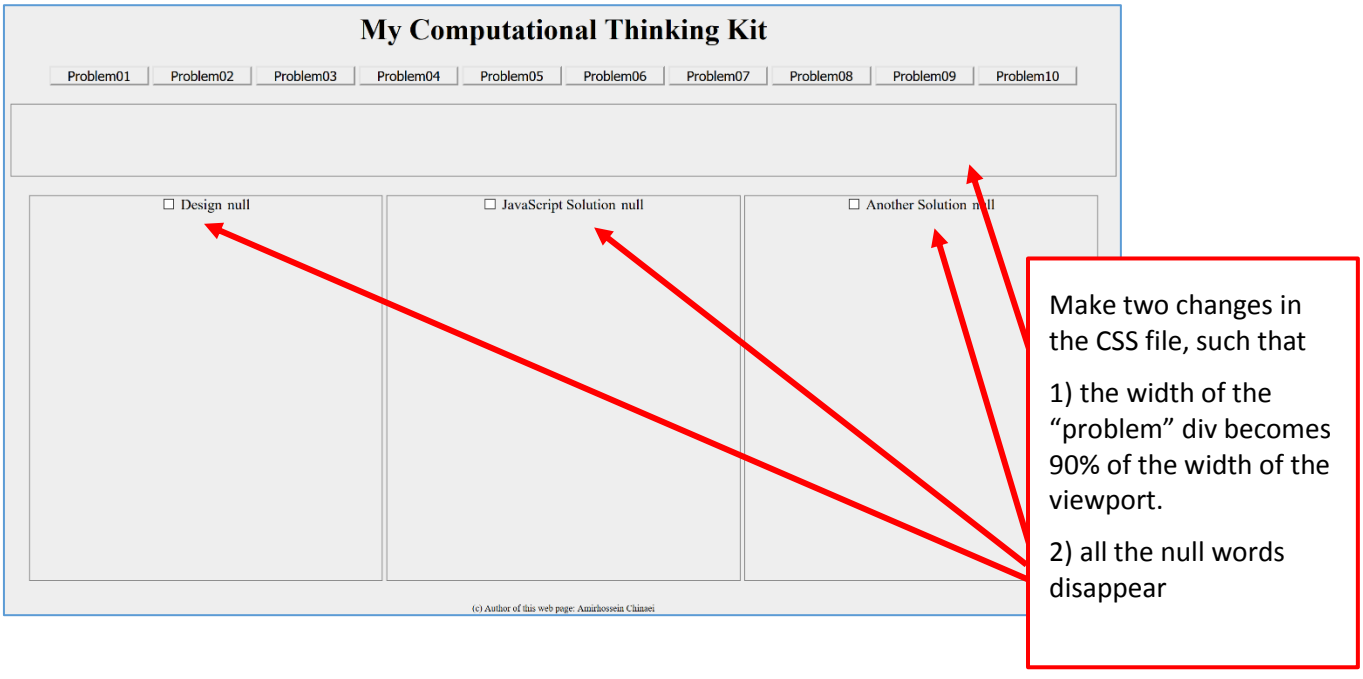
Note that this document eventually is going to be an interface (in particular, a webpage) that contains some basic computational problems together with your flowcharts and implementations. However, it takes a few weeks until we get there. So, for now we (ab)use the document towards demonstrating food items.

Exercise 1: (CREATE: **myLearningKit_Ex1.HTML** and **myLearningKit_Ex1.css)**
Copy **myLearningKit.HTML** to a new file named **myLearningKit_Ex1.html**. Copy **myLearningKit.CSS** to a new file named **myLearningKit_Ex1.css**.

Make two changes to **myLearningKit_Ex1.html**, as follows:

1) Connect it to **myLearningKit_Ex1.CSS** by adding a link tag in the head element.
2) In the footer element, replace "your name here" with your name.

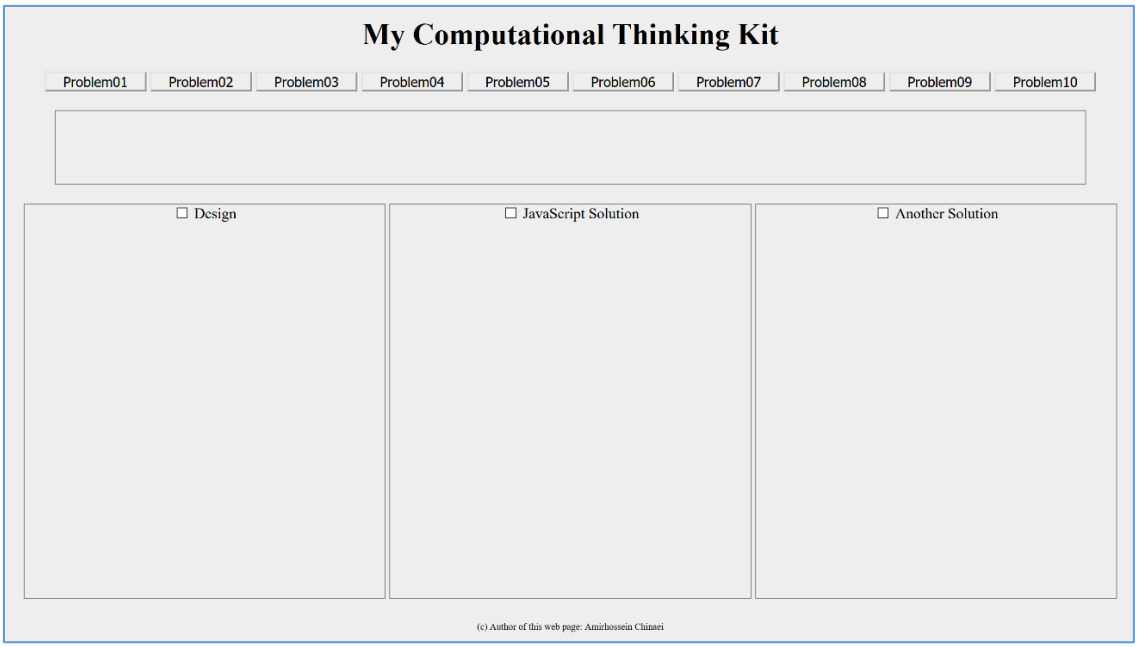
When you open your HTML file with browser, you should see the following result:



Now, as stated in the red box above, make two changes to **myLearningKit_Ex1.css** as follows. Currently, the width of the "problem" division is 100% of its parent (which is the body element). We want to have the width of the "problem" division set to 90% of the width of the viewport. Find it in w3schools how we can use width of viewport.

Side note. Make sure you get used to find help directly in w3schools (and not from other websites or search engines). Why? Because in the lab-tests, you may have access to w3schools but not to anything else.

Also, as you can see in the HTML file, the source file of the images as well as the alt attributes are all set to *null*. In other words, we do not want yet these img tags to display anything. But, in the picture above, the word "null" is displayed. If, in the HTML file, we remove the attributes src and alt from the images, our HTML code is no longer valid. Hence, we need to make a change in the CSS file such that these images do not display anything for now. After making these two changes in the CSS file, you should have the following result:



Recall: you should figure out how to make such changes by yourself for instance via w3schools (or, by some quick searches in the web). Asking for help from your friends or from TAs should be the last resort.

Optional: you may want to change the colouring and borders of some of the elements above to make your kit page more appealing. We do not grade it though.

Exercise 2: (CREATE: **myLearningKit_ex2.html** and **myLearningKit_Ex2.js)**
Copy **myLearningKit_Ex1.html** to a new file named **myLearningKit_Ex2.html**. Copy **myLearningKit.js** to a new file named **myLearningKit_Ex2.js**.

When you open the **myLearningKit_Ex2.html** in your browser, if you click on any of the buttons, nothing happens. In other words, we have not defined any behaviour for our page yet. Let's do it step by step.

1) Add an attribute `onclick` to the button designated for Problem01 such that when clicked function `p01Func` is called.

```
onclick="p01Func()"
```

2) The body of `p01Func()` can be provided in the HTML file internally. In particular, we can add a script element and write our `JavaScript` code inside it. But, similar to what we do for CSS, we encourage you to have your `JavaScript` code in an external file. To do so, add the following line to your HTML file:

```
<script src="myLearningKit_Ex2.js"></script>
```

3) Open **myLearningKit_Ex2.js**. We already provided you with the skeleton of `p01Func()`.

```
Function p01Func(){
…
}
```

The body of the function is what we write in the pair of curly braces. We want to update the content of the "problem" division with the following paragraph:

I'm looking for a type of pancake originating from the Indian subcontinent, made from a fermented batter. It is somewhat similar to a crepe in appearance.

To do so, we need to get from our HTML file the element that its ID is "problem", by the following `JavaScript` code:

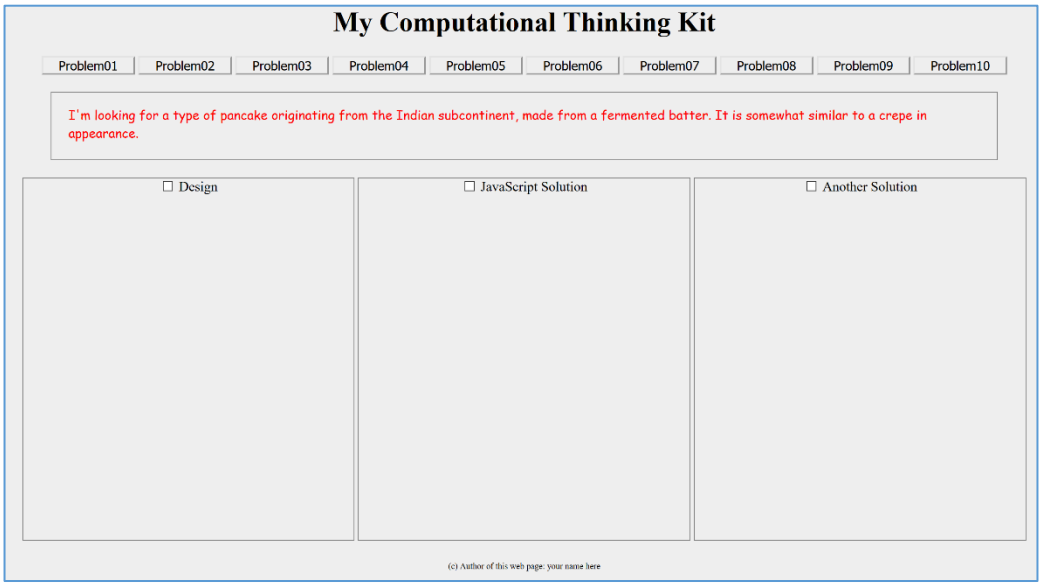
```
document.getElementById("problem")
```

This would get us a construct, that we know it as an **object**.

Then, we need to assign the paragraph above to the `innerHTML` of this object. Here, it's the `JavaScript` code:

```
document.getElementById("problem").innerHTML="<p>…</p>";
```

where **…** is replaced with the red paragraph above. Now, when you click on the button representing `Problem01`, you should see the following result.



4) At this point, you may want to revisit the **myLearningKit_Ex2.html** file and look up the `img` elements and their IDs. The idea is that for `Problem01`, `dosaDesign.jpg` is assigned to the image that its ID is "flowchart" and `dosa1.jpg` is assigned to the image that its ID is "js".
So, you need to add more `JavaScript` code to what you did in Step 3. When you get an element by its ID, you can set any of its attributes by using `setAttribute()`. Since we want to set the `src` attribute of the element with ID "flowchart" to `dosaDesign.jpg`, we need to add the following code:

```
document.getElementById("flowchart").setAttribute("src","images/dosa/dosaDesign.jpg");
```

Because we do not want to display the image yet, we need to set the display property of its style to *none*.

Similar to above, assign `dosa1.jpg` to `src` of the element that its ID is "js" and set its display to *none*. (You may want to download another image for dosa from internet (save it as `dosa2.jpg`) and assign it to the element that its ID is "another" and set its display to none.)

Note that the images are still not displayed. We will display them in the next exercise.

Exercise 3: (CREATE: **myLearningKit_Ex3.html** and **myLearningKit_Ex3.js)**
Copy **myLearningKit_Ex2.html** to a new file named **myLearningKit_Ex3.html**. Copy **myLearningKit_Ex2.js** to a new file named **myLearningKit_Ex3.js**.

The idea is to modify the new files such that when we check/uncheck the checkboxes by clicking on them, the images display/disappear, respectively.

1) First, we need to add the `onclick` event to checkbox "check1" in **myLearningKit_Ex3.html** . This is similar to what you did in Step 1 of Exercise 2 for buttons. When this checkbox is clicked, we want to call a function that we name it `checkUncheck1()`.

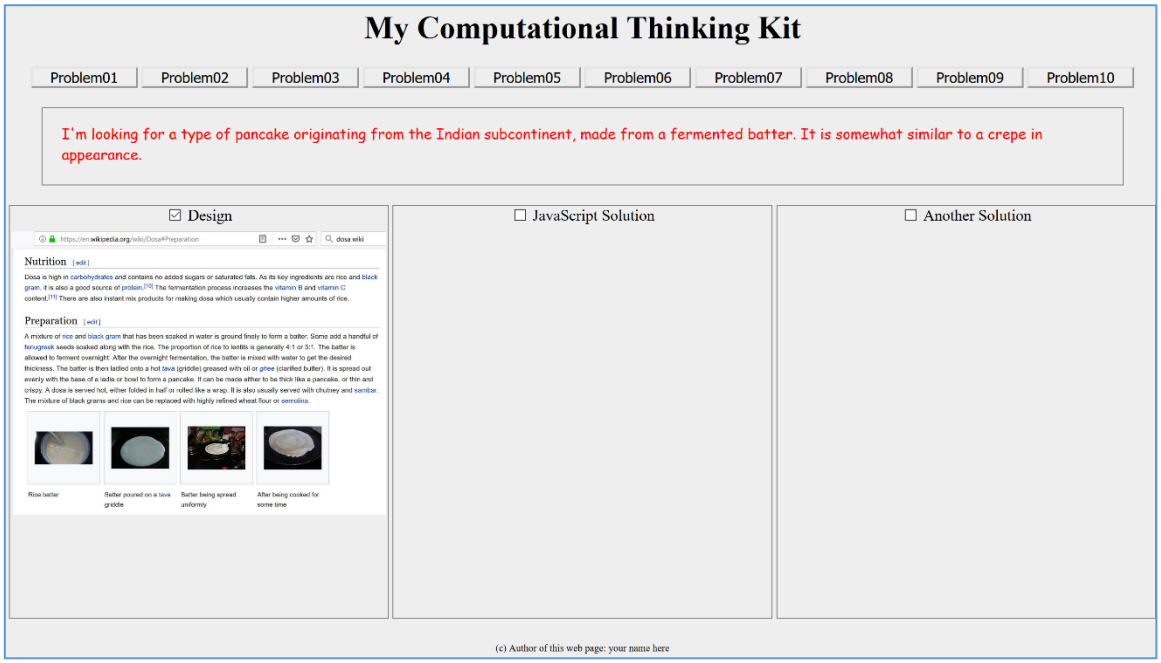
```
onclick="checkUncheck1()"
```

2) Then, we want to provide the function in **myLearningKit_Ex3.js**. Remove the two lines that we ask you to remove in the `js` file to uncomment `checkUncheck1()`. Then add each of the following two commands where appropriate.

```
document.getElementById("flowchart").style.display="inline";
document.getElementById("flowchart").style.display="none";
```

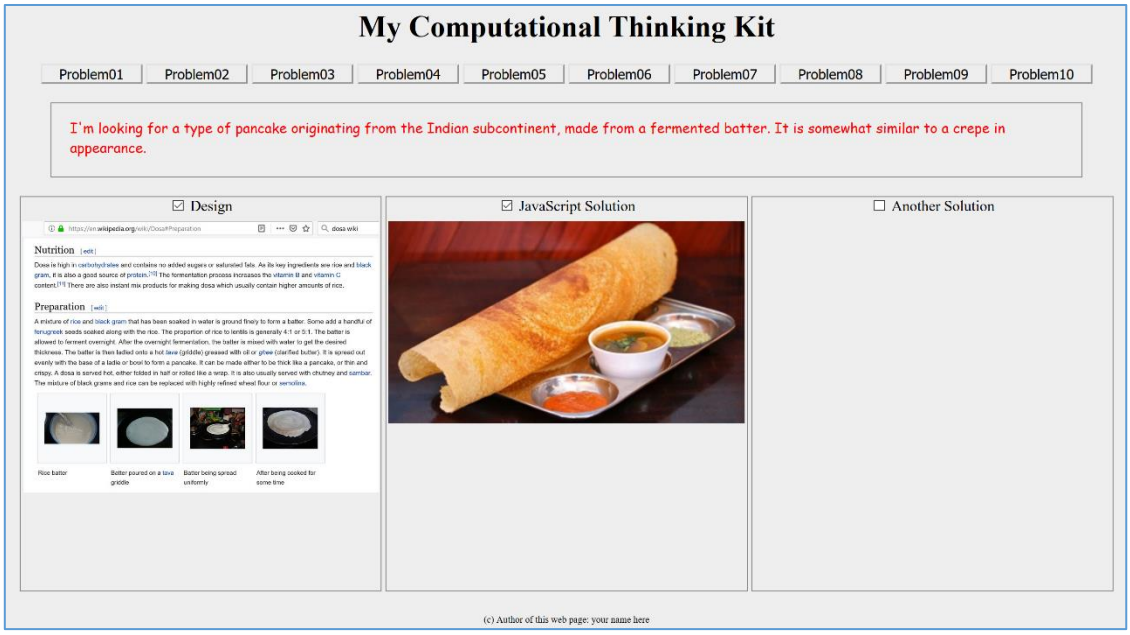
The `if` statement there determines if the image is displayed or not; if not, we display it. If yes, we disappear it. In other words, by clicking on the Design checkbox, the image's display toggles.

The following picture shows when the Design checkbox is checked.



3) Mimic steps 2 and 3 above for toggling the image that is displayed in the next panel.

The following picture shows when the next checkbox is checked too.

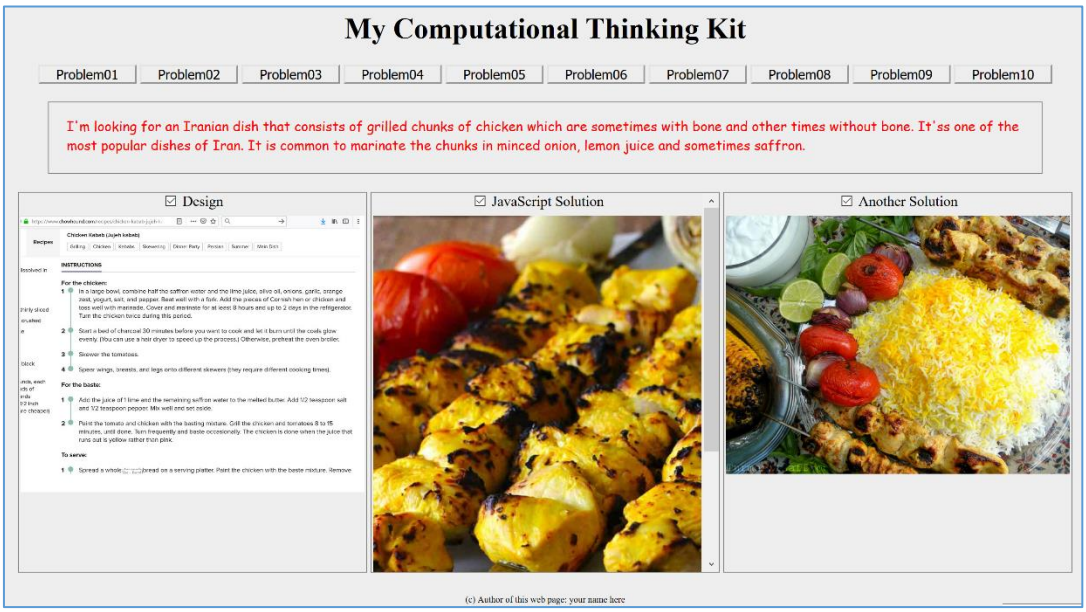


Exercise 4: (CREATE: **myLearningKit_Ex4.html** and **myLearningKit_Ex4.js)**
Copy **myLearningKit_Ex3.html** to a new file named **myLearningKit_Ex4.html**. Copy **myLearningKit_Ex3.js** to a new file named **myLearningKit_Ex4.js**.

Mimic what you did in Exercise 2, to assign problem description and corresponding images when button `Problem02` is clicked. Add function `p02Func()` to the `js` file. Its body is similar to that of `p01Func()`. The `innerHTML`, of the object we get, will have the following paragraph.

I'm looking for an Iranian dish that consists of grilled chunks of chicken which are sometimes with bone and other times without bone. It's one of the most popular dishes of Iran. It is common to marinate the chunks in minced onion, lemon juice and sometimes saffron.

The images that you assign are `jujehDesign.jpg`, `jujeh1.jpg`, and `jujeh2.jpg`. Note that during this exercise, you are required to work on `check3` as well (no matter if you did it during Step 4 of Exercise 2 or not). The following picture illustrates when `Problem02` is clicked and all checkboxes are checked.



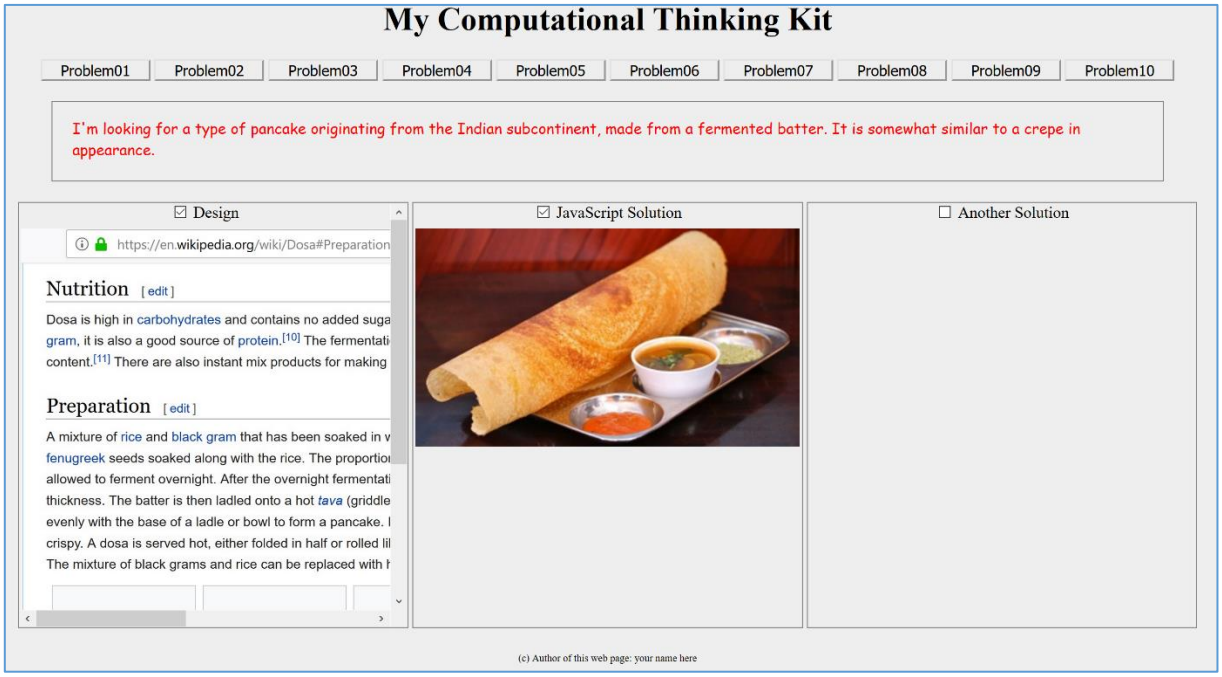
Exercise 5: (CREATE: **myLearningKit_Ex5.html** and **myLearningKit_Ex5.js)**
Copy **myLearningKit_Ex4.html** to a new file named **myLearningKit_Ex5.html**. Copy **myLearningKit_Ex4.js** to a new file named **myLearningKit_Ex5.js**.

In this exercise, we want to add two more functions to our `JavaScript` code. One we call it `zoomIn()` and the other `zoomOut()`. The objective is that when the image of the first checkbox is double clicked, we want to make its width 200% (do this in the body of `zoomIn()`) and when is single clicked, we want to have its original width of 100% (we do this in the body of `zoomOut()`).

Here, it's list of what you need to do:

1) In your HTML file, add `onclick` and `ondblclick` attributes to the image that its ID is "flowchart"
   a. on an `ondblclick` event, function `zoomIn()` is called
   b. on an `onclick` event, finction `zoomOut()` is called
2) In your `js` file, add `zoomIn()` and `zoomOut()` functions
   a. write `document.getElementById("flowchart").style.width="200%";` in the body of `zoomIn()`
   b. write a similar line in the body of `zoomOut()` to change the width to its original size

The following figure illustrates when `Problem01` is clicked and the Design image is double clicked.



Exercise 6 optional further practice:

You may want to continue this project and add at least two more problems (or food definition) and corresponding design (or recipe) and implementations (or result pictures) for further practice.

Note. In order for the project to work properly all problems should have 3 images, one for design, one for `js` solution, and the 3rd for "another solution".

G. AFTER-LAB TASKS (THIS PART WILL NOT BE GRADED)

In order to review what you have learned in this lab as well as expanding your skills further, we recommend the following questions and extra exercises:

Short-answer questions:

1) How many functions are defined in your **myLearningKit_Ex5.js**?
2) How many function calls are in your **myLearningKit_Ex5.html**?
3) How many different types of events you handled in your **myLearningKit_Ex5.html**?
4) For which element(s) in your **myLearningKit_Ex5.html** you defined two events?
5) How can you see your JavaScript errors in Firefox?

To master your skills further,

6) Create a web page with text of your choosing (a single paragraph will do), which upon the mouse being positioned over it causes the same text to be repeated below the paragraph. When the mouse is no longer positioned over the text the copy of the text remains.
7) Same as 6 but when the mouse is no longer over the text the copy of the text disappears.
8) Create a web page with two paragraphs of text of your choosing, and with a button below the text. Clicking on the button should cause a solid border to appear around the first paragraph and a dashed border to appear around the second paragraph
9) Create a web page with two paragraphs of text of your choosing, and with a button below the text. Clicking on the button should cause a new paragraph to appear below the button containing the combined text of both paragraphs above the button.

Please feel free to discuss any of these questions in the course forum or see the TAs and/or Instructors for help.