EECS 1012: LAB 07 – more on JavaScript

A. IMPORTANT REMINDERS

1) You should 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 during your lab hours.
2) You are required to pass the pre-lab mini quiz posted on eClass not later than the first 10 minutes of your lab time. You should study the recent course materials and corresponding links/hints and Section B of this document, as well as working on at least first tasks of this lab before trying the prelab quiz. You have 4 attempts; and you need at least 80% to pass. However, each time you may get some different questions. You should try your first attempt at least one day before your deadline so that, if needed, you have time to (re)study the materials for your next attempts. Failing the pre-lab mini quiz is equal to failing the whole lab, yet you are still highly encouraged to complete the lab and submit your work to eClass.
3) 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.
4) You can submit your lab work in eClass any time before 21:00 on Wednesday of the week the lab is for. In order to pass this lab, your grade in it should be at least 70%.

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

1) Complete your My Learning Kit Project (that you started from Lab03) with 30 problem definitions, flowcharts, and JavaScript Solution. Don't need to do any task for the "another solution" panel or the "Run" button yet. As you will need this project for Lab08 too, failing to complete it by Lab07 may affect your grade in Lab08 too.
2) Download this lab files and read them carefully to the end.
3) You should have a good understanding of
    - JavaScript objects, such as **Math**, **Date**, and DOM **document**
    - **array** and **string** in JavaScript

C. GOALS/OUTCOMES FOR LAB

1) To practice more concepts in programming, including variables, arrays, functions (aka sub-algorithms), and program control statements.
2) To use more JS objects, such as `document`, `Math`, and `Date`.

D. TASKS

Part 1:

1) TASK 1: Simple "HEADS" or "TAILS" button output with an if-statement.
2) TASK 2: Passing variables to functions.
3) TASK 3: Passing variables and for-loop.
4) TASK 4: Random + string concatenation + if-statement.
5) TASK 5: Date object + array + string concatenation.
6) TASK 6: Global variable and if-statement

E. SUBMISSIONS

1) Manual verification by a TA
   You may want to have your TA verifying 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.

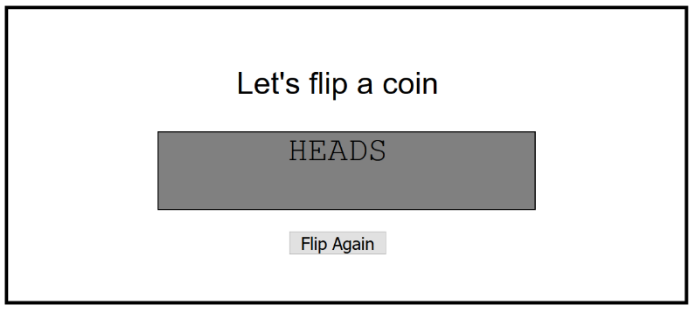
2) eClass submission

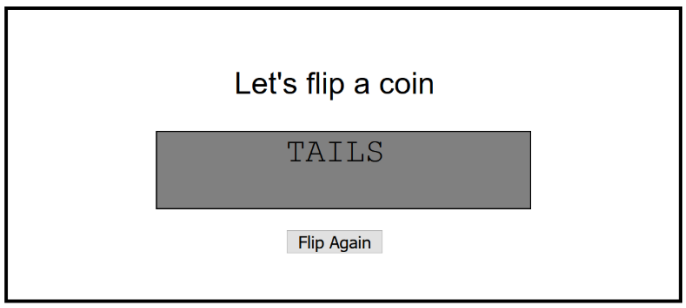
Create a **folder** named "**Lab07**" and copy **all** your HTML and JS files. Once you are done, compress the folder and upload the zip (or tar) file to eClass. Also, create a folder name "**LearningKit**" and copy **all** your Learning Kit materials; then, compress it and upload the zip/tar file to eClass.

F. FURTHER DETAILS

Task 1: Edit task1.js (you do not need to edit the HTML file).

For this task, we have already declared the JavaScript function myFunction() for you. Your function should do the following. Each time the button is clicked, your myFunction() code should call a sub-algorithm that generates a random number between 0 and 2 (including 0 and excluding 2). If the random number is less than 1, then have the innerHTML of the paragraph variable set to "HEADS", otherwise set it to "TAILS". See example outputs below.





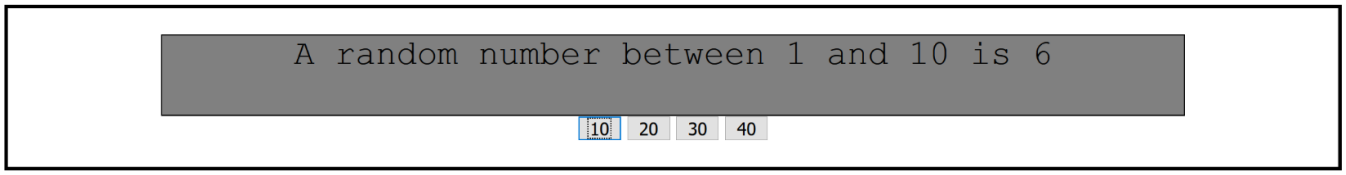
Task 2. Edit task2.html and task2.js

(1) Link your task2.js to your HTML code.

(2) Have the text in the paragraph "mydata" start with **Result** (see below).

(2) Add four buttons to your `Task2.html` as shown below.

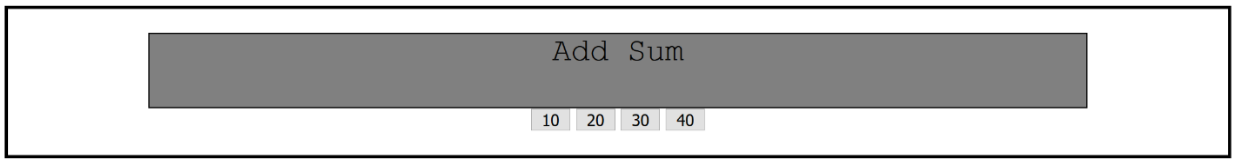
(3) Write a function in JavaScript that has one parameter. When a button is pressed, it should pass the value shown in the button (e.g., 10, 20, 30, or 40) to a function named `passNum`. In your JavaScript code, your function should call a sub-algorithm that generates a random whole number between 1 and the passed value, inclusively. See example below for when the button that passes value 10 is pressed.

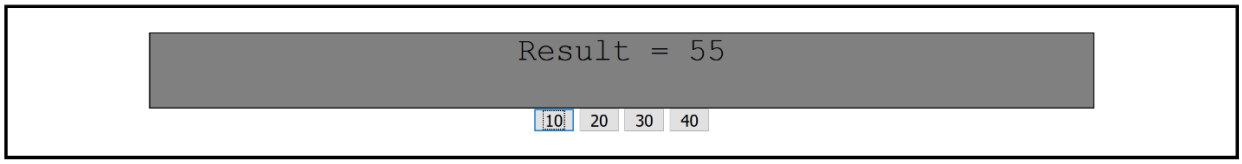


Task 3. Edit task3.js (you do not need to edit the HTML file).

Write a function in JavaScript that has one parameter. When each button is pressed, it should pass the *value* shown in the button (e.g., 10, 20, 30, or 40). Use a for-loop to compute the sum of 0 to the passed *value*.

For example, if the value passed is 10, then compute 0+1+2+3+4+5+6+7+8+9+10=55. See below.





Task 4. Modify task4.html and task4.js

(1) Link your JavaScrpit file to your HTML file.

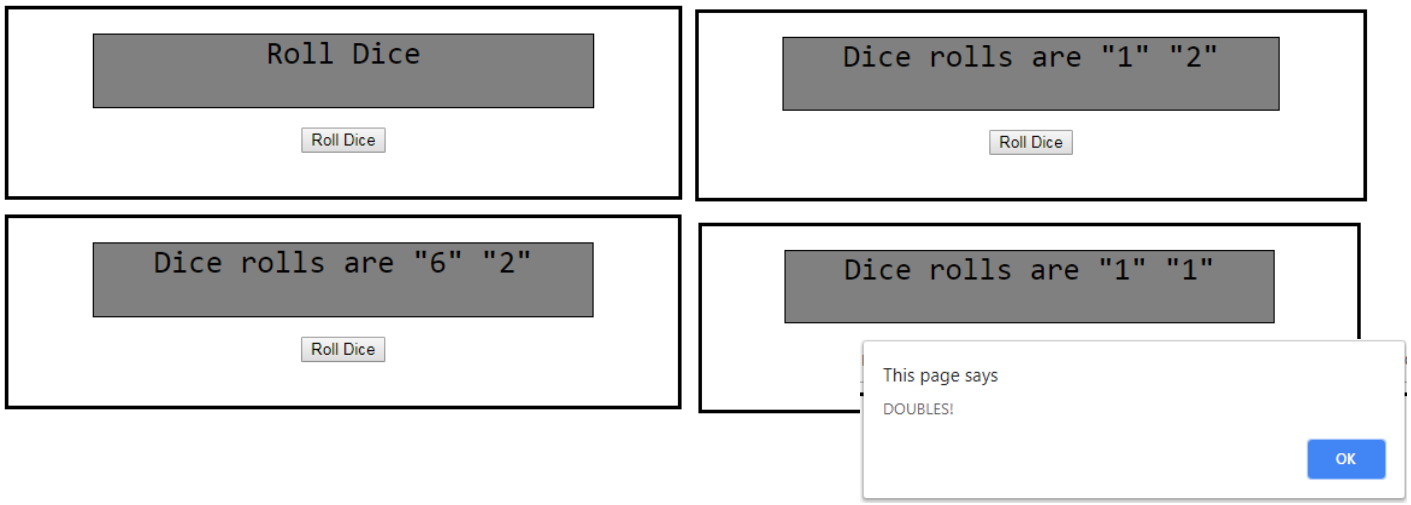
(2) Have the text in the paragraph "mydata" start with **Roll Dice**. Add a button "Roll Dice". Have this button respond the click event.

(3) Have the `onlick` for your button link to your JavaScript function. The function does not have parameters.

(4) Each time you click, have your JavaScript function compute two random numbers from 1 to 6. These represent dice. Change the `innerHTML` to say Dice rolls are "value1" and "value2", where value1 and value2 are the results of your random numbers.

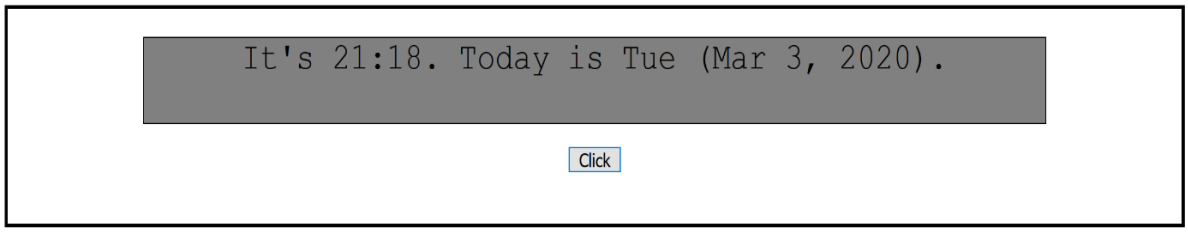
(5) If the two numbers are the equal, create an alert that says "DOUBLES!".

See examples below.



Task 5. Modify task5.js (you do not need to edit the HTML file).

In `task5.js`, complete line 27 such that when `task5.html` runs, the current time and date are shown with a format similar to the figures below.



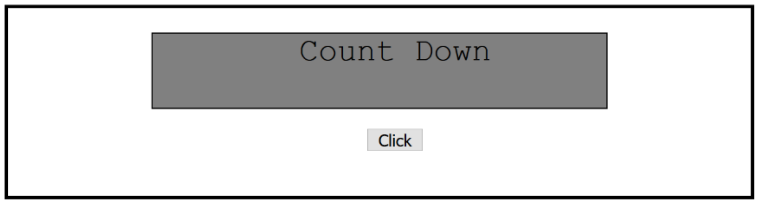
Task 6. Modify task6.js (you do not need to edit the HTML file).

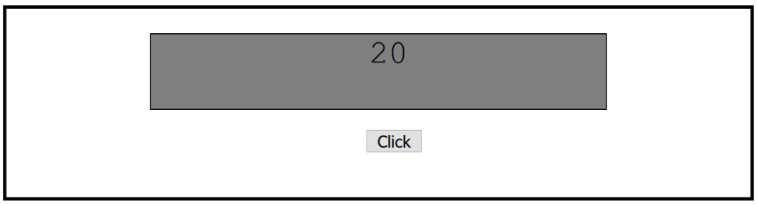
In task6.js, declare a global variable. This is a variable that is created outside your function. Inside your function, you do not need to declare it again. If you modify the variable, the modification will be remembered next time you access the function. See example code here.

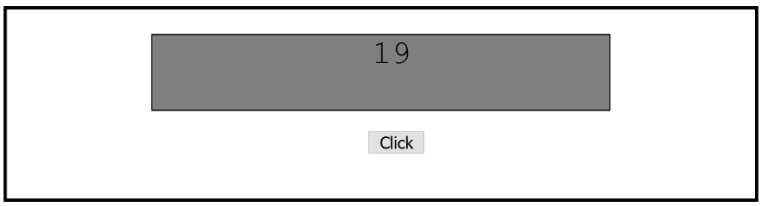
```
var i = 21;

function myFunction()
{
 i = i--; // the value of i will be remembered next function call
}
```

Each time your button is clicked, you should reduce the global variable by 1 and show the result. Your innerHTML of the paragraph with id “mydata” should show the current value of the global variable. When the variable gets to 0 or less, have the your innerHTML change to BOOM!

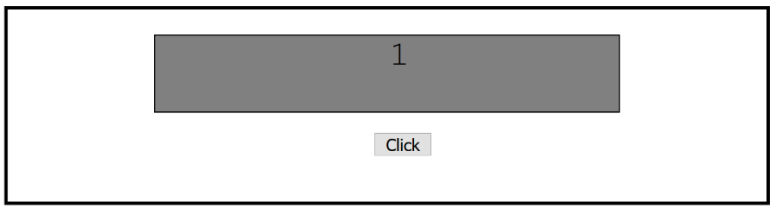
Your code should produce the exact results as the following figures. (i.e. it should not show 21, and it should show count down all the way to 0, and then show BOOM! for the following clicks on the button.)

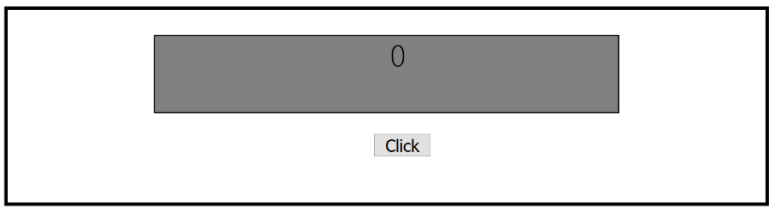


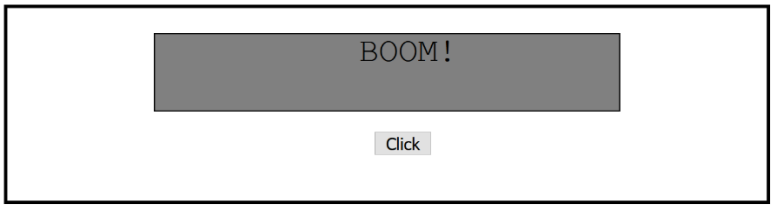




…







G. AFTER-LAB WORKS (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 practices:

1) Revisit objects `Math`, `Date`, and `document` in w3schools and explore more methods or properties of them by writing simple JavaScript codes.

a. For object Math, learn more about `PI`, `round`, `pow`, `sqrt`, `abs`, `ceil`, `floor`, `sin`, `min`, `max`, etc. (https://www.w3schools.com/js/js_math.asp)
b. For object `Date`, see 4 different constructors. (https://www.w3schools.com/js/js_dates.asp)
c. For object `document`, see different methods such as `getElementsByTagName`, etc. (https://www.w3schools.com/js/js_htmldom_document.asp)
d. Also see this fun animation made by html, css, JavaScript (https://www.w3schools.com/js/js_htmldom_animate.asp), and use your creativity to create some similar animations. As an example, make the red square go from the bottom-left corner to the top-right corner.

2) Revisit **arrays** and **strings** of JavaScript in w3schools. These two data structures have many applications in computer science and in your follow up courses. Interesting methods of strings include `indexOf()`, `lastIndexOf()`, `search()`, `slice()`, `substring()`, `substr()`, `replace()`, `concat()`, `trim()`, `charAT()`, `split()`, etc. More on strings here: https://www.w3schools.com/js/js_strings.asp More on arrays here: https://www.w3schools.com/js/js_arrays.asp
3) Add more algorithms and programs to your **Learning Kit** Project. You would need to have 35 buttons for 35 problems, algorithms, and JavaScript solutions in your Learning kit and submit it with your next lab (Lab 08). For your problem 31 to 35, you are required to choose problems that need some nested loops or sub-algorithms. Also, you should start thinking of implementing the "Run" button so that when for instance Problem 4 is active and the Run button is clicked, the corresponding JavaScript code runs. You would need to make the Run button working for all your problems by Lab 09.

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