EECS 1012 Lab 4: Flowcharts and LOOPS

Due: October 11, 9 p.m.

1 Introduction

You can submit your lab work in eClass any time before October 11, 9 p.m.

2 Learning Outcomes

The purpose of this lab is:

1. To practice computational thinking by first drawing flowcharts for basic computational problems
2. To practice translating flowcharts to functional JavaScript.
3. To gain some initial experience linking Javascript to interactive elements in an HTML.

Before you begin, watch the overview of this lab and complete the pre-lab quiz at this URL:

https://eclass.yorku.ca/mod/h5pactivity/view.php?id=3027529

3 Flowchart Task

Your first and major task in this lab is to draw six flowcharts that include either loops or selection statements. This task can be done in teams of two (but not more). You may work solo in case you are unable to find a partner for this part. Your drawings will need to be submitted as image files. Only png or jpg formats are acceptable for images.

More specifically, draw the following flowcharts and write your name on each. Use the symbols introduced in the slides for all exercises. Take a screenshot (or a picture) of each flowchart you draw and submit these to eClass with the names img_01,02,03,04,05,06.jpg—png. Each img_x should be the flowchart of exercise x below.

In case you submit images produced by a phone camera, make sure you change the file format in the camera settings to JPG (as opposed to HEIC, DNG, etc.) Feel free to reduce the image dimensions to reduce the file size – either in the camera settings or in an image editor. Try to keep the size of each image below

500 KB. Also, using exposure correction of +0.6 or +1 in the camera makes the images look better (you can also brighten them afterwards in an editor).

- draw a flowchart for the computer program called $guessAnimal(hasFourLegs, climbsTrees)$, which you coded last week. This function should guess the kind of animal based on whether the animal has four legs and/or can climb a tree. The inputs should be boolean values (i.e. true or false). If the animal has four legs and can climb, the function should return "It's a cat". If it can climb but does not have four legs, the function should return "It's a snake". If it has four legs but can't climb, the function should return "It's a dog". Otherwise, return "It's a fish". Your solution must include a SELECTION STATEMENT.

- draw a flowchart for the computer program called $isLeapYear(year)$, which you coded last week. This function should accept a year represented as a positive integer as input and return true if the input year is a leap year, else false. If you do not remember the definition of a leap year, see https://www.mathsisfun.com/leap-years.html. Your solution must include a SELECTION STATEMENT.

- draw a flowchart for a computer program called $isPositiveMultipleOf4Or7(number)$. This should accept, as input, a positive integer value and should return true if the input is a multiple of 4 or 7. If it is not, the program should return false. Your flowchart solution MUST include a LOOP (meaning, do NOT simply divide by 4 or 7 and check for a remainder; use a loop instead).

- draw a flowchart for a computer program called $bits(number)$. This should accept, as input, a positive integer value and return the number of bits that would be required to represent that number in binary. As an example, consider the number 8. In binary, this number is represented as 1000. As there are four places in the binary number, it requires four bits to represent. Your program should therefore return 4 if the input is 8. Your flowchart must include a LOOP.

- draw a flowchart for a computer program called $currHour(elapsedMinutes)$. This should accept, as input, a positive number of minutes and return a string indicating the hour of the day and assuming that $elapsedMinutes$ have passed since noon. For example, if the input is 127, the output should be "2 pm" as two hours will have elapsed since noon making the current hour 2 p.m. if the input is 727, the output should be "12 am" as 12 hours will have elapsed since noon, making the current hour 12 a.m. Note that the hour of the day should be an integer between 1 and 12, and it should be followed by "am" or "pm" in your return string. Your flowchart solution must include a LOOP.

- draw a flowchart for a computer program called $calcCoins(denomination, cents)$. This should accept, as input, a string denomination **and** a positive number of cents. The denomination will be either "quarters", "nickels" or "dimes". The function should return **two values**. The first should be the number of quarters, nickels or dimes that can be formed out of the cents in the input, and the second should be the number of remaining cents. For example, if the input denomination is "nickels" and the input number of cents is 33, the outputs should be 6 and 3. This is because it will take 6 nickels to represent 30 cents of the total, and 3 cents will remain. Your flowchart solution must include a SELECTION STATEMENT and a LOOP.

4 Programming Task

Your second task will be to translate some of the methods above into functional Javascript code. We will, moreover, link our solutions to HTML so that you can run them both at the command line and within a browser to produce a result.

Toward this end you are given the files $lab4.html$, $lab4.css$, $lab4.js$ and $lab4.test.js$ files. Review these files carefully, and try to understand each line. In $lab4.js$ you must encode the following methods, as described above. Your implementations should be consistent with your flowcharts, meaning that variable names and control flow in your flowcharts should be identical:

- $isPositiveMultipleOf4Or7(number)$.
- $bits(number)$.
- $currHour(elapsedMinutes)$.
- $calcCoins(denomination, cents)$.

In addition you will encode:

- $calcChange(cents)$. This function will take, as input, a number of cents and returns an array containing: the maximum number of Quarters that can exist in this change, the maximum number of Dimes that can exist (after the Quarters have been counted), the maximum number of Nickels (after both Quarters and Dimes have been counted), and the value of any remaining change. For example, if the input is 65, the output should be the following array: [2,1,1,0]. This is because 65 cents can be formed with 2 quarters, 1 dime and 1 nickel with 0 cents remaining.

Once you have encoded this last function, you should be able to execute every algorithm referenced within the HTML in $lab4.html$. Note however that to run your code within the browser the last line of $lab4.js$ must be COMMENTED OUT (i.e. the line that reads *"export { calcChange, calcCoins, bits, currHour, isPositiveMultipleOf4Or7 }"*. You will however need this line in order to run

the tests at the command line with vitest. Some screenshots that illustrate they way your HTML should look and behave have been included at the end of this specification.

For this lab, we have done the work to tie the HTML to the JS for you. For the next lab, we will be asking you to do this on your own! So make sure you understand how the JS is linked to the HTML and how the JS event listeners are tied to HTML elements.

5 Submission

In order to submit, compress the 'Lab04' folder with 'lab4.js' within it and submit your compressed file via eClass. The file you submit should have a .zip extension. You do not have to include the tests in this folder, but you can if you like. On lab machines, you can compress a folder by right clicking on it and selecting the "compress" option. The links in the getting started section explain how to compress files on other operating systems.

6 Marking Criteria

To receive full marks we will expect that:

- Your flowcharts correctly use symbols covered in class;
- Your JS should reflect your flowcharts;
- Your code should pass every test released with the lab, as well as a handful of additional tests that we have withheld. You should be testing your own code as well!
- Your code should be well structured and styled.
- Every variable you use must be declared! This means any variable you assign should be declared at least once as a "let", "var" or "const" variety of variable, as appropriate. We will be testing in "strict" mode, so please follow "strict" interpretation rules.

7 Extra Practice

For extra practice, revisit these exercises after the lab and think about any parts of your flowcharts that were not a good match to your JS code. Be careful not to include any JavaScript specific notation in your flowcharts. Flowcharts should be as independent as possible from any particular programming language. That means your flowchart should be understandable to anyone who knows programming even if he/she does not know any JavaScript. This means your flowchart should not use functions like parseFloat, toString, etc., or keywords

like if, else, var, etc. Remember that you cannot use "=" as an assignment operator; use "←" instead.

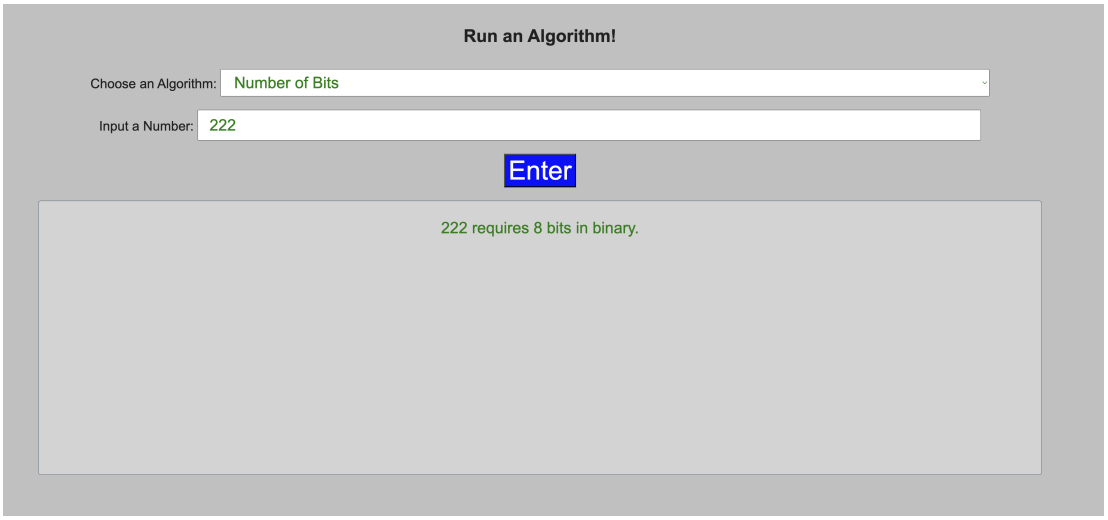

Figure 1: HTML displaying output of $bits(number)$ when input is 222.

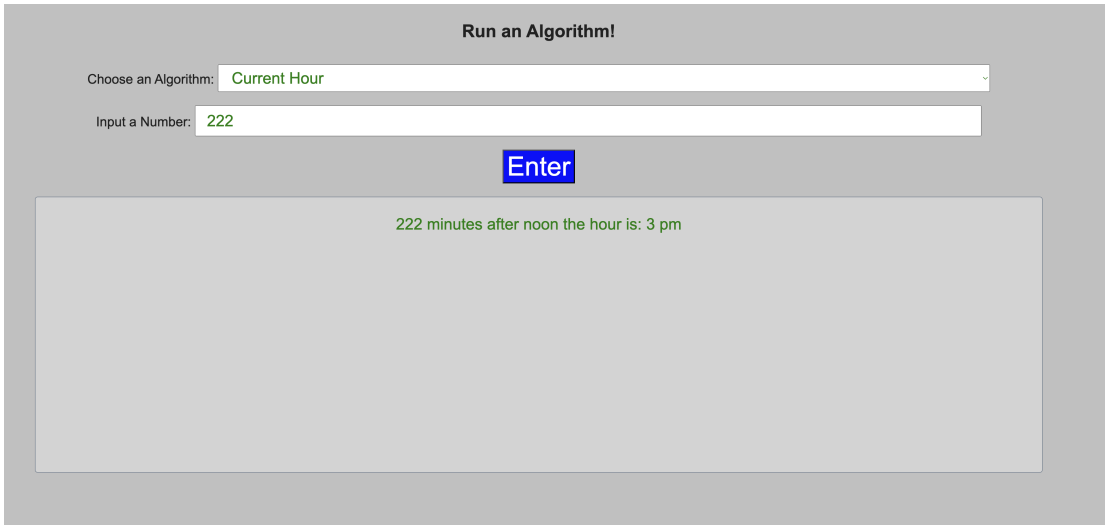

Figure 2: HTML displaying output of $currHour(number)$ when input is 222.

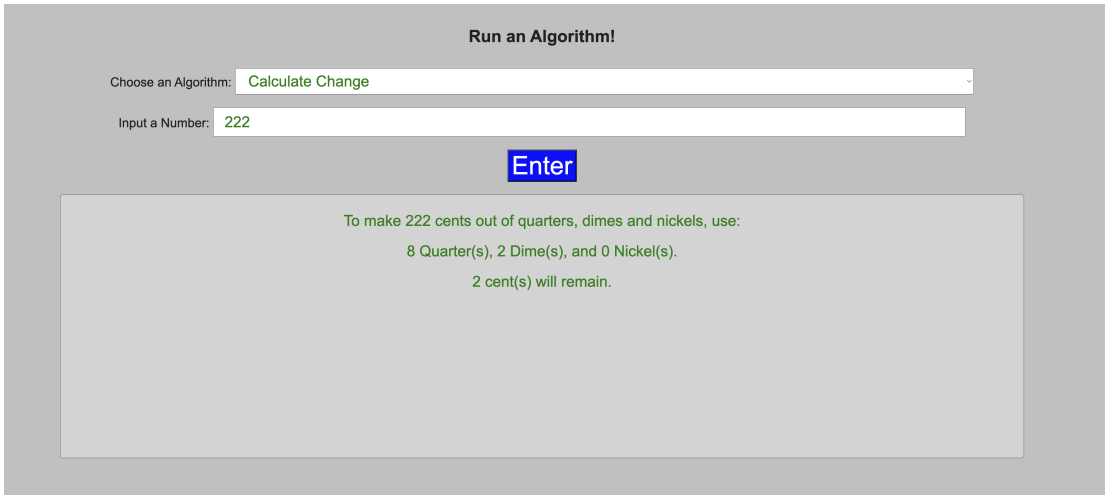

Figure 3: HTML displaying output of $calcChange(number)$ when input is 222.

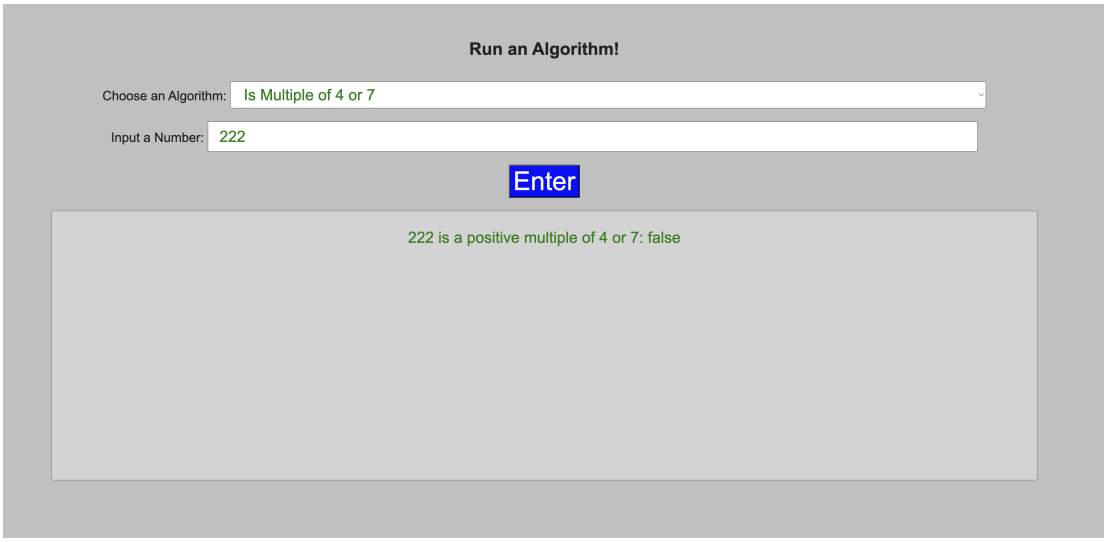

Figure 4: HTML displaying output of $isPositiveMultipleof4or7(number)$ when input is 222.