# Assignment: BMI Plus (JavaScript loops)

This challenge is to be begun in class, and completed on your own, outside of class. Work through the following steps and upload your work to your wwwroot on misdemo.temple.edu when you are done.

HINT: If, at any point in this challenge, you do not get the expected result, the ***very first thing*** you should do is to open the web developer tools in Chrome and look at the console. Very often, the hint you need to fix your mistake can be found there!

## Getting the project file

1. Using Windows file explorer (or Mac OS finder) open your mis2402workspace
2. Your instructor will provide you with a zip file named: bmi\_plus.zip
3. Open the zip file. Drag the bmi\_plus folder in to the mis2402workspace folder.
4. Now, open VS Code. Navigate to the bmi\_plus folder using the VS Code Explorer panel.
5. Observe the contents of the files in the folder.

## Overview

1. Our goal in this challenge is to solve three distinct problems. Each problem is meant to be a learning and practice opportunity for writing JavaScript “for” loops. Other than that, they don’t have anything to do with each other!

## Mathematic Trickery

1. In problem 1, you will need one arithmetic operator that might be new to you. (Although, you probably encountered it in Junior High, or even earlier!) It’s called “modulo” and it is nothing but a fancy word for what you called the “remainder” when you were learning how to divide. We would say that 14 modulo 3 is 2. We say that because 14 divided by 3 is 4 with a remainder of 2.

Again: 3 \* 4 + 2 = 14.

In JavaScript, we use the % to perform the modulo operation. Some examples follow:

x = 14 % 5; // x will equal 4

x = 14 % 6; // x will equal 2

x = 14 % 7; // x will equal 0

x = 14 % 8; // x will equal 6

x = 14 % 9; // x will equal 5

Instructions

1. **problem1.html** - Multiples of 5 and 7.

If we list all the natural numbers below 20 that are multiples of 5 or 7, we get 5, 7, 10, 14 and 15. The sum of these multiples is 51.

Find the sum of all the multiples of 5 or 7 below 1000.

* 1. *How do I solve that???* First, define a variable to hold your answer. You could call it “answer”. Give that variable an initial value of 0.
  2. Next, write a loop that iterates through the numbers 0 to 999.
  3. Inside the loop, check each number. Is the number a multiple of 5 or 7? Use an “if” statement to check the number and respond accordingly.

HINT: You know a number x is a multiple of 5 if x % 5 equals 0.

* 1. If a number is either a multiple of 5 or multiple of 7 then add it to the answer variable.

HINT: Be careful that you don’t add the same number twice. For example: 35 is a multiple of **both** 5 and 7, so be careful that you only add 35 to the answer once.

* 1. When your loop is done, the answer variable will hold the value you are interested in. Use the same JavaScript trickery we saw last week to put that value into the tag with the id of “answer”

$('#answer').html(answer);

**ATTENTION!** Here you have been instructed how to write the solution to a specific problem with step-by-step instructions. As you become more familiar with JavaScript, and programming in general, you will be expected to solve problems more independently.

Ultimately you should be able to think through a problem at a high level, outline a series of steps to solve the problem, and then use your knowledge of programming to implement those steps. When you can do that, you’re a programmer!

1. **problem2.html** – Powers of 2.

Write a loop that generates the powers of 2. Start at 21 and end at 220. Display the powers of two in an ordered list tag <ol> as shown in the following picture.

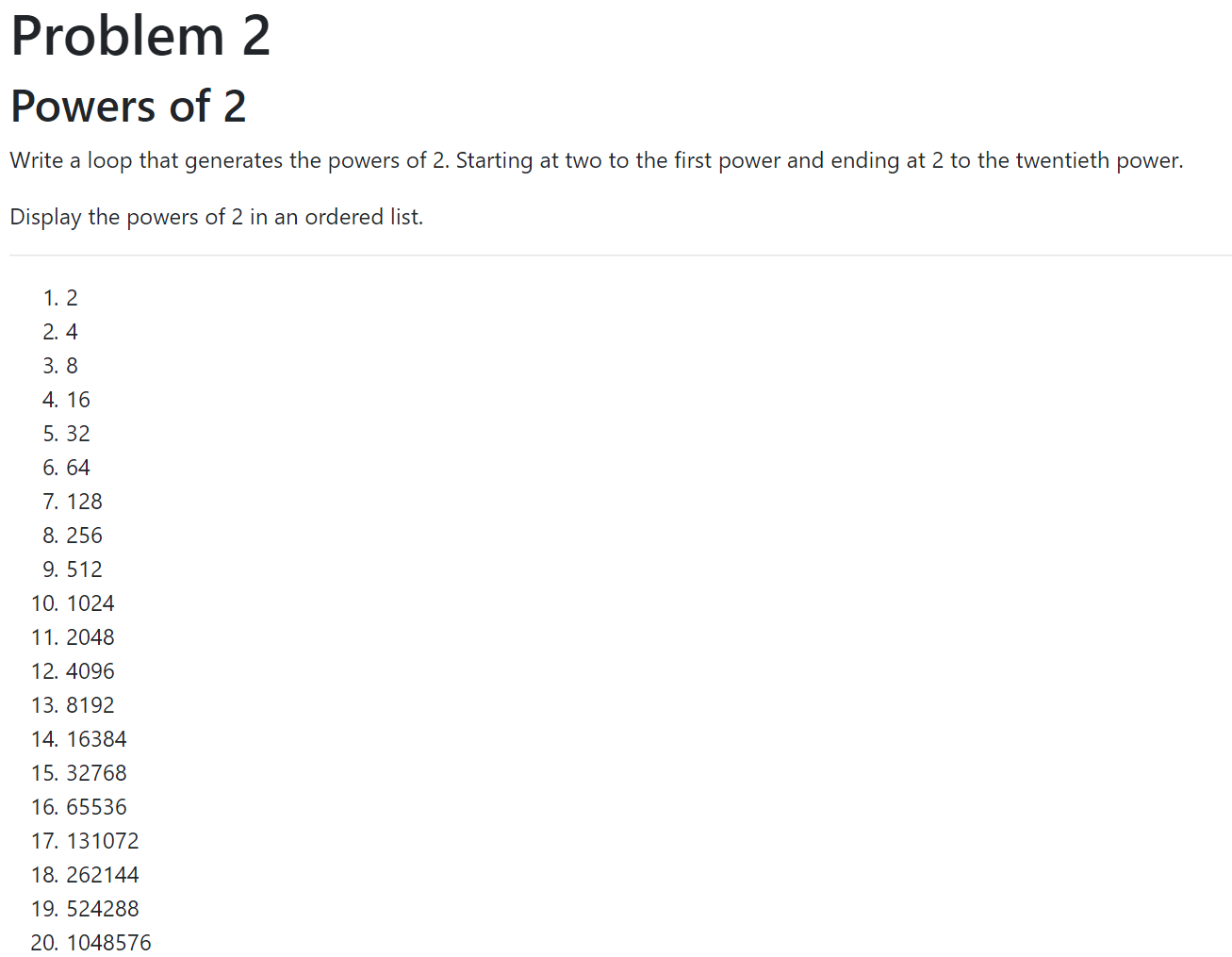
* 1. *How do I solve that???* First, define a variable to hold your answer. You could call it “x”. Give that variable an initial value of 2.
  2. Now write a loop that steps through 20 iterations. You could define a loop variable of “i” that steps through the values 0 to 19.
  3. In each step of the loop, construct a piece of text, stored in a string variable, that looks like this:

var the\_li\_tag = '<li>' + x + '</li>';

* 1. Now, prepare to be amazed! You can ***append*** html content to the existing <ol> tag with a command like the one below.

$('#answer').append(the\_li\_tag);

* 1. Now all that remains to be done is to multiply x by 2 and store the new answer back into x.
  2. Save your work. Now try running / reloading problem2.html. The results are shown below.



**ATTENTION!** Please notice that we now have new option for adding html to a page. The .**append** method of the mysterious **$** function. We’ll talk more about the **$** function later on in the course, but for now you should know that you can use it to add a *single* piece of html content to an existing tag using the **.html** method, or you can add *multiple* pieces of content to an existing tag, on piece at a time, by appending the content to an existing tag using the **.append** method.

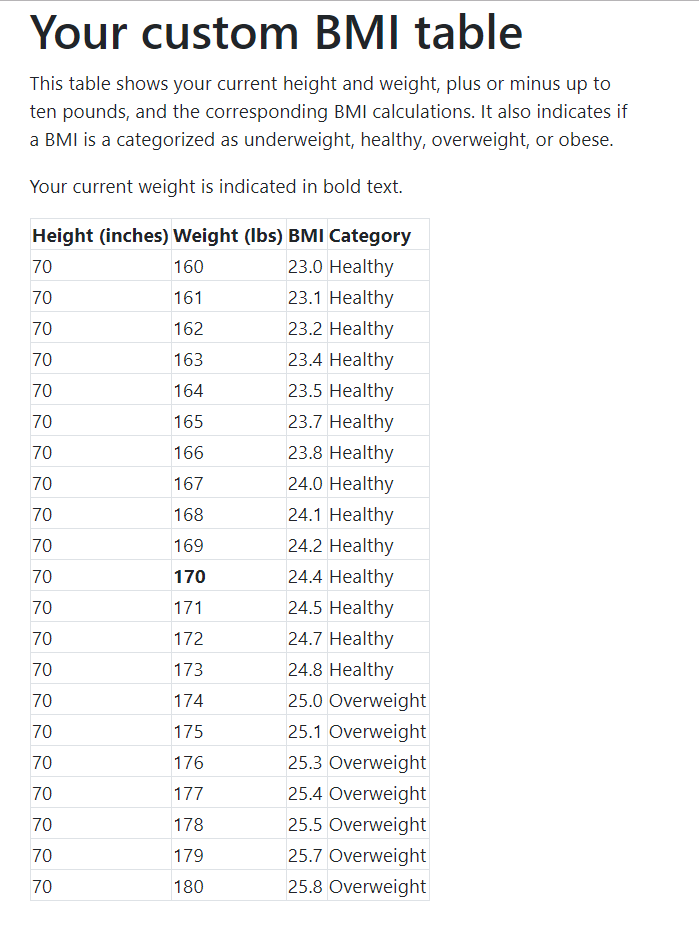
CONTINUED…

1. **problem3.html** – Your custom BMI table.

Add rows to an existing HTML table to shows a user’s current height and weight, plus or minus up to ten pounds, and the corresponding BMI calculations. Also indicate if a BMI score is categorized as underweight, healthy, overweight, or obese.

The user’s current weight should be indicated with bold text.

So, answering the prompts with 70 inches and 170 pounds would generate the following table.



1. To solve this problem, you should look at the instructions found in the problem3.html file itself.

There are comments there, intended to help you. The exact BMI formula is there too. Go look! Give it a try!

This, problem, incidentally, would make a good exam question. It uses variables, math, loops, and conditional statements. It also requires some knowledge of HTML. If you can solve this problem on your own, you are doing great!

1. Test your work and upload it to the class server when you are done.
2. Be sure to upload your work before the due date / time. This assignment will be graded.

**MORE LOOP PROBLEMS (OPTIONAL)**

If you want more practice here are two more loop problems for you to consider. These problems are not necessarily easy for the novice programmer, but they are good learning exercises. Not only do they require you to write a loop or two, but they give you a chance to sharpen your problem-solving skills.

**Optional problem 1** – Fibonacci numbers. If necessary, use Google to look up the definition of a Fibonacci number. Write a loop to generate 30 Fibonacci numbers. Append them to an ordered list. If the number is divisible by three, make it bold.

**Optional problem 2** – Prime numbers. If necessary, use Google to look up the definition of a prime number. Write a loop to generate all the numbers from 1 to 100. Append them to an ordered list. If the number is prime, place an asterisk next to it.

How will this assignment be graded?

|  |  |
| --- | --- |
| Item | Point Value |
| Uploaded OK? (solution must be ***exactly*** where requested!) | 10 |
| Problem 1 correct (must be calculated with JavaScript!) | 30 |
| Problem 2 correct (must be calculated with JavaScript!) | 30 |
| Problem 3 loop correct (must be calculated with JavaScript!) | 20 |
| Problem 3 – used conditional statement in the loop to bold text. | 10 |