Assessment:	**Web Programming 27(8)1 Self-directed assessment**
Duration:	110 minutes

Assessor:	T. Mkwaira
Moderator:	-

WEB PROGRAMMING 27(8)1: SELF-DIRECTED ASSESSMENT

INSTRUCTIONS: GENERAL

- This a self-directed exercise for personal assessment. It will not be marked.

INSTRUCTIONS: SUBMITTING YOUR TEST

- You will not submit this assessment.

INSTRUCTIONS: INTEGRITY AND LIABILITY

- Integrity: Observe the Honor Code1. Each candidate is expected to maintain academic integrity.

INSTRUCTIONS: FORMAT OF THE ASSESSMENT

- This assessment includes T/F questions.
- Encapsulate all your logic inside functions.
- Create a separate file for each of the tasks. This could also be an HTML page with embedded JavaScript.
- The allocated time includes time for uploading/ saving your solution.

Assessment:	**Web Programming 27(8)1 Self-directed assessment**
Duration:	110 minutes

COMPLETE THE FOLLOWING TASKS:

TASK 1: [15 MARKS]

You can use any of JavaScript's inbuilt methods for this task, and there is no limit to the approach, e.g., a requirement to use functions.

- Create a JavaScript file and declare two arrays at the beginning. The first array is *fruit* and must contain any 3 fruit elements. Call the second array *vegetables* and it must contain any 3 vegetable elements.[2]
- Display the length of the *vegetable* array in the console. [1]
- Put the two arrays together into one array, *fruit* first. Call the new array *food*. You can use the array method called join or do it using a loop.[2]
- Remove the last 2 elements from your new array. [2]
- Reverse your *food* array. You could use an inbuilt method or do it manually. [2]
- Remove only the first element in *food*. [2]

TASK 2: [15 MARKS (11 + 4 BONUS)]

Create a user-input based application that performs some calculations on two numbers based on the chosen operation.

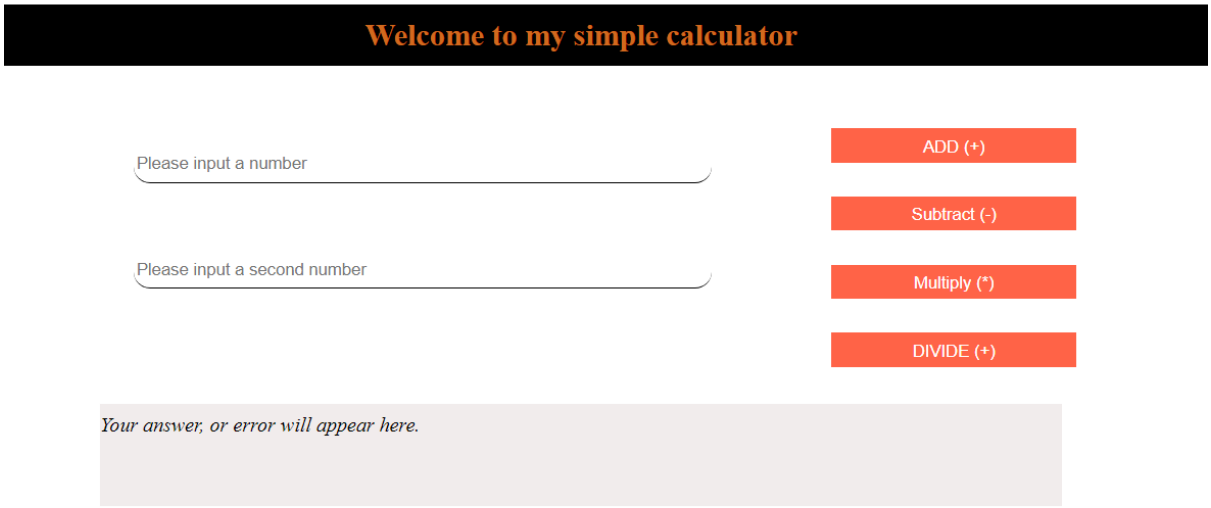
Note:

- The form has been provided, however note that none of the elements have been given an ID. Do not assume that the form does not have any other important information. I encourage you to test. You are welcome to create your own form if you wish, but you would have to match the styling.
- You should validate the input from the user. Remember that a user may provide a value which is not entirely made of numbers, or they may give input nothing at all. You are welcome to use input type = number for the relevant inputs.
- The answer should appear in the div that appears at the bottom. You can write a value by using:

```
document.getElementById("id_of_output_div").innerHTML = yourAnswer
```

- The form may need some additional attributes, for example if you want to use an ID to create objects that represent your elements.
- For bonus marks: [4]
    - *Change the colour of the text in the answer div to red if there is an error and green otherwise.*
    - *In the output, you should show what the calculation was, for example 1+3 = 4*

Assessment:	**Web Programming 27(8)1 Self-directed assessment**
Duration:	110 minutes



.

TASK 3: [MAX 25 MARKS]

Here is code that generates a random number in a range given by min and max.

```
const random = (min, max) => Math.floor(Math.random() * (max - min)) + min;
```

- Create a function that returns 10 random numbers between 1023 and 3201. [2]
- Create a second function that returns 10 random *but unique* numbers between 3201 and 1023. [2]
- Change all the numbers into strings by mapping through them with the JS *map* function (or forEach etc.), and by using string concatenation.
  You must encapsulate this logic in a function and return an array. [2]
- How many values in the new array of strings end with the value 2? Use a function to return the answer.[2]
- Change the code so that you can ask the user for input. [5]
- How would you give the user feedback depending on their input? [3]
- Can you think of another way to provide the answers to a user by using the document as an input/ output modality? [5]

Assessment:	**Web Programming 27(8)1 Self-directed assessment**
Duration:	110 minutes

HOW TO SUBMIT: ZIP ONLY! ZIP ONLY!

NAMING CONVENTION

You must submit a **ZIP** file that uses the Campus standard naming convention. Any assessment that a student creates for electronic submission must conform to the following naming convention:

WPR27(8)1 CT2_StudentSurname_StudentName.zip

The reason for this is to ensure that your assessment is received correctly by your lecturer; this prevents any inconvenience for you as the student and the lecturer.

PACKAGING YOUR SOLUTION

1. Create a folder that follows the naming convention explained above.
2. Move the Tasks folder into the named folder you created in Step 1.
3. Finally, create a **zip** archive of the folder you created in Step 2.
4. Upload your zip archive.