CS275 Web and Mobile App Development

Assignment 1

As with this and all future assignment, you must work individually.

Objective:

The purpose of our introductory CS275 assignment is two-fold:

1. Set up your client side environment
2. Get acquainted with the basics of HTML, CSS and JavaScript my developing a small application that encompasses some of the concepts recently learned.

Part 1: Selection and Configuration of Development Environment

In this course we are developing cross-compatible sites, so developing on your local machine will be both acceptable and convenient

All you will need for our client side applications is:

- A modern web browser (we recommend Chrome)
- A basic text editor (like Notepad or Textpad)
    - Using something like Microsoft World would add additional formatting and therefore most likely would result in issues with your code.

Part 2: Developing a Simple HTML Page

Now that you have decided on your client-side setup, let's start developing!

In this assignment we are going to create a webpage that shows information about Fibonacci numbers. You may want to refresh yourself about them here:

https://en.wikipedia.org/wiki/Fibonacci_number

In our first iteration of our webpage we will create a pure HTML page.
Do the following:

1. Open up a text editor (your choice!)
2. Create a simple HTML web page that has
    a. A title in the title bar
    b. A paragraph about the history behind the Fibonacci sequence.
3. An appropriate image representing this topic (just Google "Fibonacci images"; there are plenty of example images.
4. Save your page as "<studentID>_HW1.html"
5. Open your file in a web browser to see the result.

Part 3: Adding JavaScript

Now add to your page:

1. A text field
2. A button
3. An empty div

The behavior should be as follows:

1. The user types some non-negative number in the text box
2. The user clicks the button
3. On this event, get the string stored in the text field and convert it to an integer.
4. Compute the value of this number in the Fibonacci sequence.
5. Display the result in the (previously) empty div.
6. Although there's no actual consensus, for consistency let's consider the Fibonacci sequence to start at $F_0 = 0, F_1 = 1$. Therefore if the user entered 2 into the text field you could output the 2nd number in the Fibonacci sequence, F_1, etc..

Debugging Hints:

1. Most browsers allow you to right-click somewhere on a webpage and choose something like "Inspect". This will bring up a window that allows you to see errors in Javascript among other things.
2. JavaScript has a function `alert(object)` that pops open a window with the object printed out. This can be useful for debugging as well.

Part 4: Handling More Cases

What would happen if there's nothing in the text box? If there's a negative number? If there's non-integer text?

If we try to cast a string into an int, and Javascript cannot do it, then `parseInt` method will assign the variable a value NaN (not a number) to the variable.

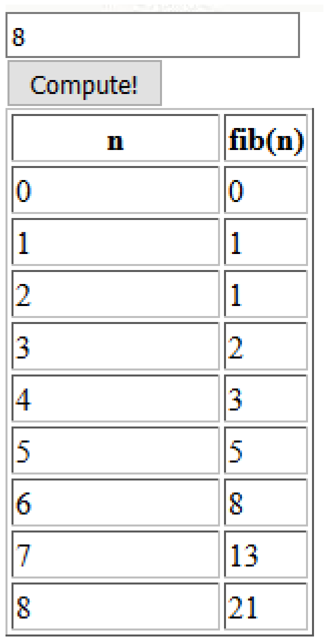
Use this information to make your program more robust:

1. If it can't convert the text to an integer, display "Invalid Input" in the div
2. If the integer is negative, display "Cannot compute Fib of a negative integer"
3. Otherwise do what you did in Part 3

Part 5: Make a Table

Now it would be really cool if we were given a non-negative integer then we create a table of the Fibonacci sequence up to an included the integer entered.

For example:



n	fib(n)
0	0
1	1
2	1
3	2
4	3
5	5
6	8
7	13
8	21

Change your program so that if there's a non-negative integer in the text box when the button is clicked that we generate this sort of table.

Hints:

1. This will be a good opportunity for you to play with loops and string concatenation.
2. You may want to construct a single long string that is the table and then set the div's content to that.

What to submit

For submission you are to submit:

- A screen cast video to Blackboard detailing a thorough code review of your program along with a demo execution of the application.
- Your source code, well internally documented.
- README file on how to run your code.

Grading (50) Points

- 40 points : program correctness and along with adherence to the stated requirements
- 5 points : quality of internal documentation and code style
- 5 points : README file