So far, we have used Python to get you familiar with basic coding concepts, such as variables and assignments. Today we will be using JavaScript to apply the same knowledge to create simple GUI messaging. GUI stands for graphical user interface that uses icons and other visual designs.

A. JavaScript in HTML
We’ve now spent a couple of weeks learning HTML and how to use it to create basic web pages.
   I. HTML Pros:
      a. Good for displaying information (like a brochure)
      b. Easy to use
   II. HTML Cons:
      a. Can’t really interact dynamically with the user
      b. Can’t perform calculations

JavaScript is a solution to most of the HTML Cons issues. We will outline many of the concepts behind JavaScript to make your webpages even better in this lab.

Step 1:
Create a new folder named Lab7 in your CSE3 folder. Open up Notepad++ and save your file in that folder, named IntroJS.html

Step 2:
Create a basic HTML skeleton. Please refer to the other labs if you do not remember how to write the HTML skeleton. In the head tags, set the title to “JavaScript Trees” using the title tags. In order to use JavaScript in your HTML, you have to put all of that code in between a pair of script tags. Place these tags below the <title> tags in your <head> tags:

   <script type = "text/javascript">

   </script>

Step 3:
Inside of the script tags, we are going to be writing JavaScript code, not HTML code. We will start by creating some variables and practicing with JavaScript so that you can familiarize yourself with it before creating the webpage for your lab.

Variables in JavaScript, for the most part, operate similarly to variables in Python. They can be assigned a numeric, string, or Boolean value and each unique variable can hold one value at a time. However, there are two key differences with how variables are handled in JavaScript:
1. To declare a variable in JavaScript, you **MUST** have the keyword `var` before your variable name.
2. All variable assignments/declarations **MUST** end with a semi-colon. As you will see in later examples, the same applies for almost every code you write in JavaScript.

**Below are examples of valid variable usage in JavaScript:**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>var object;</code></td>
<td>Valid declaration of var “object”</td>
</tr>
<tr>
<td><code>var name = “Emily”;</code></td>
<td>Valid string assignment of var “name”</td>
</tr>
<tr>
<td><code>var age = 22;</code></td>
<td>Valid int assignment of var “age”</td>
</tr>
<tr>
<td><code>var isTrue = true;</code></td>
<td>Valid boolean assignment of var “isTrue”</td>
</tr>
</tbody>
</table>

**Below are examples of invalid variable usage:**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>var item</code></td>
<td>Invalid declaration; Missing semicolon</td>
</tr>
<tr>
<td><code>width;</code></td>
<td>Invalid initial declaration; Missing “var” keyword</td>
</tr>
<tr>
<td><code>length = 5;</code></td>
<td>Invalid initial declaration; Missing “var” keyword</td>
</tr>
<tr>
<td><code>var height = 10;</code></td>
<td>Valid int assignment of var “height”</td>
</tr>
<tr>
<td><code>var height = 11;</code></td>
<td>Invalid declaration + initialization; Cannot Redeclare a variable using the same name</td>
</tr>
</tbody>
</table>

**Step 4:**
Let’s use JavaScript to figure out your Summer plans.

I. Create 4 variables:
   - `name, age, summerPlan, vacationMessage;`
   a. You can have each variable declared on its own line with the keyword `var`:

   ```javascript
   var name;
   var age;
   var summerPlan;
   var vacationMessage;
   ```

   b. Or you can combine them on one line and separate them with commas like so:

   ```javascript
   var name, age, summerPlan, vacationMessage;
   ```

II. Now we can assign values to `name` and `age`. Go ahead and put down your own name and age.
   a. `name` is of type string, so we have to put its value inside quotation marks.

   ```javascript
   name = “Emily”;
   ```

   b. `age` is a number, so no quotations are needed.
Step 5:
Now that we have all of our information, we need to display it somehow. In JavaScript, there are TWO ways to display info on a webpage:

1. `document.write();`
2. `alert();`

document.write() prints out information directly onto the webpage, whereas alert() shows the information in a pop-up box. Use `document.write()` to write name and alert() to write age:

```javascript
document.write("My name is" + name);
alert(age);
```

B. Operator Overloading
We all know what the plus sign (+) does right? It adds two numbers together. But, in JavaScript, it can do much more. There are many things we want computers to do and limited symbols on the keyboard, so to overcome this, we can **overload** a symbol by having it do different things.

I. A plus sign can:
   a. **Add two numbers together**
      i. \(5 + 5 = 10\)
   b. **Concatenates two strings together** (think of concatenating as stapling 2 strings together to get 1 bigger string)
      i. “Super” + “Bowl” = “Super Bowl”
      NOTE: the space after the letter ‘r’ in “Super”. If you had “Super” + “Bowl”, then JavaScript would do “Super” + “Bowl” = “SuperBowl”.
   c. **Concatenates a string and a number together** (when it sees a string and a plus sign, JavaScript will just slam everything together):
      i. “Super” + 8 = “Super 8”
      ii. “Super” + 47 = “Super 47”
      iii. 6.7 + “Super” = “6.7Super”
   d. If there are more than 2 items in the expression, follow the order of operations:
      i. 5.5 + 5.5 + “Super” = “11Super”
      ii. 5.5 + (7.4 + “Super”) = “5.57.4Super”
      iii. “Super” + 5.5 + 7.4 = “Super5.57.4”

C. Interacting with the program
So far, we have learned how to use variables, calculate values, and display information on the webpage. We haven’t yet shown how a user can INTERACT with the program.

To get information from the user, we can use the `prompt` function to ask for the user’s input.

**Step 1:**
Modify your program to ask for the user’s name and age using the `prompt` function:

```javascript
name = prompt("What is your name?");
age = prompt("How old are you?");
```

Try it and see what happens!

**Step 2:**
Use concatenation to create a sentence with the name and age; display both variables with `document.write()`. And Let’s remove `alert(age)`.

I. Texts that are contained within quotation marks are considered to be *literals*. This means that whatever is inside the quotations is the exact text that gets printed.

II. `name` and `age` are considered to be *variables*, so they must go OUTSIDE the quotation marks. This way, JavaScript knows to look for their assigned value instead of just printing the texts “name” and “age”.

```javascript
document.write("My name is" + name + " . I am " + age + " years old. <br />");
```

When you run the program, the user should be prompted to input their name and the year that they were born in. Once they do that, the webpage should print the appropriate sentence containing the user’s name and age.

**D. Conditional Statements**
Conditional statements allow you to make choices and do different things with your code depending on the situation. You can think of conditional statements as acting like branches in a flow chart or decision tree.
Remember if statements from Python? JavaScript (and all other real programming languages) also contains if statements and they function pretty much the same way. **NOTE:** Unlike Python, JavaScript utilizes {curly braces} and does not use colons.

As a review, if statement checks if the part inside the parentheses (called the condition) is true.

I. If the condition is **TRUE**, then the code inside the first set of curly braces is executed.

II. If the condition is **FALSE**, then the code inside the second set of curly braces (underneath the else statement) is executed.

Example:

```javascript
if (condition)
{
    alert("First Set!");
}
else
{
    alert("Second Set!");
}
```

**Step 1:** Write an if/else statement to check if:

I. Your age is greater than or equal to 21. If this is true, output “You’re OK to drink!”

II. Your age is less than 21. If this is true, output “You should stick with water.”

III. Try it with different age values to make sure it works!

```javascript
if (age >= 21)
{
    document.write("You're OK to drink!" З);
}
else
{
    document.write("You should stick with water.");
}
```

The if/else statement above can be illustrated using the following decision tree diagram:
Step 2:
Let’s add if statements to our own code! You might notice that Summer Vacation is coming up soon. Let’s figure out your plans using JavaScript.

I. Prompt the user to ask them if they any plans for the upcoming summer vacation. We are going to save their response to the summerPlan variable.

```javascript
summerPlan = prompt("Do you have any plans for this summer?");
```

II. If they say “yes,” prompt the user where they are planning on staying and save this new a new variable named vacationLocation. We can declare and assign value to a variable in one line.

```javascript
var vacationLocation = prompt("Where do you plan on staying?");
```

III. Once we know where the user is going, let’s create a message and save it to the vacationMessage variable.

```javascript
vacationMessage = "I hope you have fun at " + vacationLocation + "!";
```

IV. If they don’t have plans for the summer vacation yet, let’s display a different message by assigning different value to the vacationMessage variable.

```javascript
vacationMessage = "You still have time to plan for summer vacation!";
```

V. Then, we are going to update our document with this new information.

```javascript
document.write(vacationMessage);
```
NOTE: Recall the difference between equals signs. The ‘==’ (double equals) and ‘=’ (single equals) are the main two we will use. ‘==’ is used for COMPARING two values and ‘=’ is used for ASSIGNING values on the right-hand side to the variables on the left-hand side.

E. Functions in JavaScript

Once you run this program, you will notice that our responses are in one line, which is a little difficult to read. We can fix this by adding <br /> (breaks tag) at the end of each line.

We have a few sentences right now so it’s not difficult to add <br /> tags to all of them. But imagine having more than twenty sentences! It will be tedious and time-consuming to add them individually.

We can use function to do this efficiently. It is similar to functions in Python but in JavaScript, you need to declare function using the function keyword instead of def. Like Python, a list of parameters to the function is enclosed in parentheses and separated by commas.

I. Function Declaration/Definition

A function definition consists of the function keyword, followed by:
- The name of the function
- A list of parameters to the function
- JavaScript statements

```javascript
function functionName (parameter1, parameter2) {
    // JavaScript Statements that get execute when the function gets called
}
```

Example:

```javascript
function displayName (name) {
    var greetingMessage = "My name is " + name;
    return greetingMessage;
}
```

Just declaring a function, does not execute the function. We need to invoke the function by calling it.

II. Function Call

You can call or invoke function by using the function name and parameters in parentheses if there is one. For example, the displayName function can be called like this:
III. Return value
A function can return value and the return value can be used outside of the function. For instance, the `displayName` function returns the `greetingMessage`, which we can save it to another variable or use it directly.

```javascript
var greet = displayName("Emily");
document.write(greet);
```

Or

```javascript
document.write(displayName("Emily"));
```

**Step 1:** Below the if-statement blocks, let’s declare a function named `addNewLine` with one parameter `message`.

```javascript
function addNewLine(message){
}
```

**Step 2:** Inside the function brackets, let’s add the `<br />` tag to the passed-in parameter and save it to a new variable named `addNewLine`.

```javascript
function addNewLine(message){
    var newMessage = message + "<br />";
}
```

**Step 3:** We need to return this updated value, so we can use them outside the function. Let’s write a return statement above the closing bracket.

```javascript
function addNewLine(message){
    var newMessage = message + "<br />";
    return newMessage;
}
```

**Step 4:** Let’s call the `addNewLine` function and pass in the `vacationMessage` as a parameter.

```javascript
addNewLine(vacationMessage);
```
Step 5: Save the returned value into a new variable named `newMessage` and modify `document.write(vacationMessage)`.

```javascript
var newMessage = addNewLine(vacationMessage);
document.write(newMessage);
```

Step 6: Let’s also modify other statements. This time, we are going to use the returned value directly.

```javascript
if(age >= 21){
    document.write(addNewLine("You are OK to drink!
```

F. Putting it all together

Task: We’re going to write a new program to ask users whether they are graduating this quarter and what their plans are for after graduation. It will output different questions/responses depending on how the person answers each question.

Step 1:
Create a new file called `Graduation.html` in Notepad++ and save it in your Lab7 folder.

Step 2:
Using what you’ve already learned, follow along with your TA to create a program that:

I. **Creates new** isGraduating, goingGradSchool, graduateYear **variables.**

II. Ask users if they are graduating this quarter.

III. If the user is graduating, the program will ask them if they are planning on going to grad school.

IV. If the user is not graduating, the program will ask them what year they will be graduating: 2019, 2020, beyond.

V. Respond with some comment about their answer.
Step 2:
Use statements of the if/else if/else structure to create the following program:

```javascript
var isGraduating = prompt("Are you graduating this quarter?");

if (isGraduating == "yes") {
    var goingGradSchool = prompt("Are you planning on going to grad school?");
    if (goingGradSchool == "no") {
        document.write("Congratulations! I hope you find a job that you love!");
    } else {
        var collegeYear = prompt("What year will you graduate?");
        var graduateYear <= 2019 ?
            document.write("You are almost done!");
        : graduateYear <= 2020 ?
            document.write("You're halfway there!");
        :
            document.write("You still have a lot of time left, enjoy yourself!");
    }
} else {
    var collegeYear = prompt("What year will you graduate?");
    var graduateYear <= 2019 ?
        document.write("You are almost done!");
    :
        document.write("You're halfway there!");
}
```

Step 3:
Use the question structure above and the JavaScript template below to complete the program.

**NOTE:** Conditional statements and strings are case-sensitive so be careful with your answers.
G. Putting it Online

**Step 1:**
Modify your CSE3Page.html to include a link to your Graduation.html and IntroJS.html page.

**Step 2:**
Put everything online and get checked off.

**Checkoff:** Go to your homepage via the class webpage and demonstrate to the TA/Tutor that your Graduation.html and IntroJS.html are complete. You must run through the program and show that ALL branches of your decision tree work.

REMEMBER to drag your Lab7 folder into the WHITESPACE inside your public_html/CSE3 folder!