Assignment 6B Reflection

One of the bugs I encountered was calling a function in the onload portion in the <body> tag. However, this created a bug since the html file is loaded before some of the variables and functions are defined in the attached javascript file. In order to mitigate this, I instead chose to call the functions directly at the bottom of my javascript file so that things will get initialized the way I want them to be without the onload function. From this, I learned that we have to thoroughly understand the order and level of the code before implementing them. Otherwise, the countless HTML null type input bugs will take up a lot of time later.

Programming Concepts

- Local storage: starting from the zoo lab, I have been trying to use local storage to store different types of data to achieve different goals. I started with hard coding the homework and storing plain strings in the local storage. Later on, in order to better parse different values from the data, a constructor was used and instances were created and stored in a list in the local storage.
- For loop: in order to correctly update information on my cart page, a for loop is needed to iterate through all stored instances in the currApps variable, and the number of iteration depends on the length of the list.

```
function updateCart() {
    if (localStorage.currApps)
    {
        if(document.getElementById("appt1")!==null)
        {
            current_appointments = JSON.parse(localStorage.getItem("currApps"));
            for (let i = 0; i < current_appointments.length; i+= 1)
            {
                // console.log(current_appointments[i]);
                var array_value = current_appointments[i];
                document.getElementById("appt"+(i+1)).innerHTML = displayApptText(appts[array_value]);
            }
        }
    }
}
```

- Console.log: this is the print function in javascript and is hugely helpful when I debug or simply want to look into the data structure within a variable. In the displayCount() function, this function is called twice to showcase the difference between a list and its

length, making me more aware of the data structure.

```
function displayCount() {
    let displayCount = [];
    if(localStorage.currApps)
    {
        displayCount = JSON.parse(localStorage.getItem("currApps"));
        console.log(displayCount);
        console.log(displayCount.length);
        //console.log(displayCount);
    }
    else
    {
        let currApps = []
    }
    document.getElementById("appointmentCount").innerHTML = displayCount.length;
}
```

- String operations within function: when prompting confirmation message or updating cart, different properties of the instance object should be parsed and displayed separately and string operations are used here.

```
function displayApptText(current_appointment)
{
    return current_appointment.type+": "+current_appointment.date+", "+current_appointment.time+" @"+current_appointment.location;
}
```

- Json operations: to properly store and retrieve instances that I created for the calendar, json.parse is used. With these operations, I was able to get the exact instance whenever I needed for the functions.