Homework: Introduction to Javascript, JSI

1. Explanations of terms as explained to a 12-year-old –

* **Variables** are words used to represent the values of things in a computer program, kind of like the variables x and y in algebra problems. Examples would be using the word ‘distance’ as a variable to represent how far you travel to get to school. Usually, variables can be changed to calculate different values, like plugging in different values for distance to calculate the distance you travel to get to your best friend’s house after you calculate the first distance to school. Sometimes the variable is given a constant value and is set to never change, so then it can only have one value.
* **Strings** are like the alphabet and numbers connected together in a sentence. The strings could be just numbers when they are used to represent something without using them for calculating with math. An example of a number string is the number part of your street address for your house. It is used like a name, or symbol, to represent your house, but not as a mathematical value for calculations. Strings can, also, use other symbols besides letters and numbers, such as punctuation in a sentence. Basically, any character that can be typed from your keyboard and shown on your computer screen, including the empty white space between characters, can all be included in a string by putting quotes around the string to contain the characters together. It can include any number of characters, from 0 characters to multiple characters in the same string. If the string has 0 characters, it is called an ‘empty string’.
* **Functions** are like many programs within a computer program. We can write functions to do simple tasks such as add numbers together, or find the average of a set of numbers. Another type of function that could be written would be a function to display a greeting on the screen every time someone opens a computer program. The function can be repeated as many times as you like in the program just by calling the function, so that you don’t have to repeatedly type out the steps of the function every time. Functions can be very flexible by giving them the ability to accept different values, called **arguments**, that are input into the function to perform the functions task. So, if your program adds numbers together, you could enter the numbers 5 and 9 to be added the first time you call the function, and you could add different numbers every time you call it by inputting different arguments every time, such as 23 and 36 the second function call, and maybe 100 and 1,000 the third time you call it. To get the answer back from the function to use in your computer program, you need to use a **return** statement. The return statement uses the key word ‘return’ at the beginning of the statement, and it returns the value of the expression that follows it. The return statement will, also, end the function and return back to the main program.
* **If statements** are used to control the flow of the computer program. A computer program usually performs each statement one after the other from the beginning of the program to the end. However, sometimes we want to be able to have the program perform different tasks and output something different for different situations. The if statement allows us to have the computer perform a check to see if a certain condition is met, like a variable having one particular value, and then if the condition is met the program will do the task that is defined to do within the if statement. If the condition is not met, then the computer will skip over that task and move on to the next task in the program. In this case, the task defined in the if statement will never be performed. Here is an example. If you have a shopper reward’s program for your clothing store and you want to give every 10th person who enrolls in the rewards program online a special discount on their next purchase, then you would use a variable, such as ‘count’, to count the shoppers that enroll. You could then use the if statement to check to see if the variable ‘count’ is evenly divisible by 10. If it is then you could give the instructions to print out a statement to the shopper that says, “Congratulations! You are shopper number 20. You have won a reward of 30% off your next purchase up to $200!”, but if the variable ‘count’ is not evenly divisible by 10, then this instruction would be skipped over and the shopper would not receive this special discount.
* **Boolean values** are special values in computer language because they can be used to control the flow of the computer program. Boolean values only have two values, either ‘true’ or ‘false’. These values are used a lot with control statements, such as the if statement that I explained above. When the computer checks the condition that has to be met (in the case above the condition is that the variable ‘count’ has to equal 10), it is checking to see if it is true or false. If count equals a number that is evenly divisible by 10, then it is true. If count equals any number that is not evenly divisible by 10, then it evaluates to false. This is what determines if the instructions within the if statement will be performed, or skipped over, whether the Boolean value of the condition is true or false.