**Coding Dojo Pre-Class Warm Up Notes**

* **Predicting the output.** 
  + **Variables can be a letter for example ‘x’ and it has a value that it oupouts.**
    - **Var x = 5;**
    - **Console. log(x);**
  + **So what it is asking for is what is the out put of the x value.**
  + **The answer would be 5 because that is what the x variable equals to.**
  + **Challenge 2**
    - **Var x = 3; “Line one, variable x. Set x to be the value of 3”**
    - **X = x + 1; “line two, set x to be the value of x+1**
      * **🡪set x to be 3+1 -> x=4**
      * **Note the value in the values diagram as 3->4**
    - **Console.log(x);**
      * **The answer would be 4**
* **Every time you go down the line the line before it affects the output.**
  + **Challenge 3**
    - **Var x = 3;** “x is 3”
    - **X = x+2;** “we know that x is 3 but now add 2 it equals 5 (3+2)
    - **Console. log(x+2);** “the new value of x is now 5 bc of previous equation so 5+2=7”
    - **Console. log(x); “**the x variable would be 5 bc of the new variable x+2.
  + **Every consol.log(x) is asking for the output. So this challenge has two answers 7,5**
  + **Challenge 4**
    - **Var x = 3;**
    - **X = x\*x;** “The x value is now 3\*3 which is 9 so the set value of x is now 9”
    - **Console.log(x);** “x=9”
    - **Console.log(x\*2)** “since x=9. 9\*2 is 18
      * **The answer would be 9,18**
* **Understanding Arrays**
  + **Var x = [1,3,5]** “this is an array that has 3 values. X of 0 is 1, x of 1 is 3, and so forth.
    - X[0]=1 x[1]=3 x[2]=5. So basically it is signifying the order of the array.
  + **Console.log(x[0]) “**so they are asking what the what number is in the position of 0 which is 1.
  + **Challenge 6**
    - Var x = [1,2,4] “so we know the variable x is set as an array of [1,2,4]
    - X[0] = x[1]; “now the number at position 0 is equal to the one at position one. So basically x[0]=1🡪x[1]=2
    - Console.log(x);
      * The answer would be a new array: [2,2,4]
  + **Challenge 7**
    - **Var x = [1,3,5]; “**so we know the value x=1,3,5”
    - **Var y = x.length; “**x.length is asking for the number of values in the array. Basically asking how many ‘drawers’ there are in the x ‘cabinet’**.** *So the answer would be: y=3;*
    - **Console.log(y);** “the answer would be 3”
  + **Challenge 8**
    - **Var x = [2,4,6,3,7];**
    - **Var y = x.length – 2; “**x.length is 5 var y=5.2 so y=3
    - **Console.log(x[y]); “**we log x of y equivalent ot saying consol.log(x[3]) so the output would be 3
      * **So this problem is a little bit trickier but has the same formula as before. So we know the length is 5**
        + **But y is asking for that -2 so it equates to y=5-2 which is 3 so we know y=3**
      * **Now console.log(x[y]) is asking for the number that is in the position of 3.**
        + **We know the value of y is 3 so if we look at the array the third position would be 3**
* **Advanced Array Manipulations**
  + **Var x = [1,3,5,8]** this is an array with 4 values.
  + **Temp = x[x.length-1];** we have a variable called Temp and temp equals whatever it says to its right.
  + **X[x.length -1] = x[0];**
  + **Console.log(x);**
    - So basically what it is asking for is what x equals to which is the array that I have to figure out.
    - Temp has nothing to do with it, it is just another variable that is introduced to the equation.
  + **Challenge** 
    - **Var x = [1,3,5,8];**
    - **X.pop();** when *pop* comes up they are asking to kick out the last variable in the array and *push* means to tack on the new variable in the last position.
    - **X.push(7);**
    - **Console.log(x);**
      * **So the new array would be [1,3,5,7]**
    - As a not the difference between PUSH and POP is their function.
      * PUSH is used to add an item to the ‘stack” while POP is used to remove an item from the stack
  + **Challenge** 
    - **Var x = [1,3,5];**
    - **X[0] = 15;**
    - **X[3] = 77;**
    - **Console.log(x);**
      * Basically, this is a “trick” question in that some programing languages will make a new value and add it into the array so the answer would come out to be
        + **[1,3,5,77]**
        + But the other times the language will just throw an error message out.
* **Understanding If/Else Statements**
  + **var counter = 1;** the variable “counter” is equal to 1
  + **if(counter == 1)** if counter is “equal” (==) to 1 then it executes what is in between the curly brackets.
    - **{**
      * **Console.log(‘hello’);**
    - **}**
    - **Then they would log hello. Since it satisfies the condition stated in the curly brackets.**
* the difference between = and == is their function.
  + == is a reference comparison, i.e. both objects point to the same memory location
  + = or .equals() evaluates to the comparison of values in the objects.
* When using “if(variable < some number) then it has to satisfy those conditions. If it doesn’t you skip the curly brackets that follow it.
  + **Challenge**
    - **Var x = [2,4,5];**
    - **Var counter = 0;**
    - **If (x[counter] > 0)**
    - **{**
      * **Console.log(‘coding’);**
      * **Counter = counter + 1;**
    - **}**
    - **Console.log(x[counter]);**
  + The only trouble I had was not reading the code all the way through.
    - **The answer is Coding, 4**
    - We know that x has the array of 2,4,5
    - Variable counter is = to 0
    - If x[counter which is 0] is bigger than 0 then you do the curly bracket
      * Which you would log Coding
      * And now counter+1 (which you do not need to write down. It’s just setting a new variable for counter.
    - Now you finish off with the final line which asks for the new variable of counter.
* **For Loops**
  + **Challenge**
    - **For (var i=0; i<10; i++)**
      * **{**
        + **Console.log(i);**
      * **}**
  + When line one is executed it creates a variable ‘i’ which is currently equal to 0
  + And then it runs a condition as an ‘if statement’ and each time it finishes you log it.
  + And when the loop is done you do the i++ statement and keep adding 1 to it unless otherwise noted.
    - So the answer would be 0,1,2,3,4,5,6,7,8,9 and finishes it.
    - Okay so what I had trouble with was when to log.
      * You log after you finish when the condition is satisfied and then you log whatever it is and move onto the final move which was the i++
      * And stop when the condition can’t be satisfied.
  + **Challenge**
    - **Var x = [1,3,5,8]**
      * **For(var i=0; i<x.length; i++)**
      * **{**
        + **If(x[i] >4)**

**{**

**X[i] = 0;**

**}**

* + - * **}**
      * **Console.log(x)**
    - the answer comes out to be [1,3,0,0]
      * basically we have the x variable which is [1,3,5,8]
      * and now we have a line that we need to execute.
        + It creates a variable ‘i’ which is = to 0
        + And we know that ‘i’ is less than x.length which is 4 so know we do what is inside the curly bracket.

*If x[i] is larger than 4 go on to the next bracket but since x[i] is currently x[0]=1 it does not satisfy it so we do not proceed to the second curly brackets.*

*We go back and now i=1*

*Is x[i]>4🡪x[1]=3 it is less than 4 so we do not do the 2nd curly bracket.*

*Now i=2 and is still less than x.length*

*Is x[2] >4? Yes because x[2] = 5*

*So we move onto the next curly bracket which states that x[i] =0*

*Basically saying x[2]=0*

*So the current array would be [1,3,0,8]*

*Now just repeat the whole steps all over again until the condition stated can’t be satisfied.*

* + - **OMG JULIE READ THE LINES MORE CAREFULLY, YOU KEEP MAKING MISTAKES BC YOU DON’T READ IT THROUGH. GAWD.**
* **Understanding “Functions” and “Returns”**
  + **Functions declarations load before any code is executed.**
  + **Function expressions load only when the interpreter reaches that line of code.**
    - **if you try to call a function expression before it’s loaded, you’ll get an error. If you call a function declaration instead, it’ll always work, bc no code can be called until all declarations are loaded.**
  + **Challenge**
    - **Function a()** *doesn’t actually run the line of code between the curly brackets until it is called which is a(). It will then say “oh, you want me to run the curly brackets which is the function.”*
    - **{**
    - **Console.log(5);**
    - **}**
    - **a();**
      * *the answer is simply 5*
      * ***Remember it will not run the function if it is not called for.***
      * ***Also, when dealing with a longer function code always do the innermost functions first.***
  + **Challenge**
    - **Function a(b, c)**
    - **{**
      * **Return b+c;**
    - **}**
    - **Console.log(a(1, a(2,3)));**
      * we start with the inner most function which is a(2,3) now we have values for b and c which is 2 and 3 consecutively.
        + We see return b+c which is now 5
      * Now we are left with the function
        + Console.log(a(1, 5));
      * We then go back and return b+c again but now the values of b and c are 1 and 5 consecutively
      * And we are now left with console.log(a(6))
      * **The answer is 6.**
* **Using and Understanding For Loops** (Lots of trouble with this concept of for loops.)
  + **Challenge**
    - **For(num1 = 0; num1 <= 3; num1 ++)**
    - **{**
      * **For(num2=0; num2 <=2; num2 ++)**
      * **{**
        + **If(num1==5)**
        + **{**

**Console.log(‘coder’);**

* + - * + **}**
      * **}**
      * **Console.log(‘ninja’);**
    - **}**
    - *It keeps running the highlighted function repeatedly until it inevitably reaches the fact that num2 is <= to 3* 
      * *It then jumps out of the loop and logs “ninja” and then returns to the first line before repeating the for loop again.*
      * *It never logs ‘coder’ because num 1 never = 5*
    - *The answer would be* ***ninja, ninja,ninja,ninja***
    - *As a side note: when it resets the for loop the variables are all reset.*
  + ***Challenge***
    - **For( var num1 = 1; num1 <= 3; num1++)**
    - **{**
      * **For(var num2=1; num2 <= 2; num2 ++)**
      * **{**
        + **Console.log(‘Dojo’);**
      * **}**
    - **}**
    - *The answer is* ***Dojo,Dojo,Dojo,Dojo,Dojo,Dojo***
      * *How you get to that answer is that we know that num1=1 and it satisfies all of it’s limits*
      * *Then we get into the for loop and it logs dojo twice before it doesn’t meet its limits and then we jump back to the first for loop*
        + *Now we have num1=2 which is still <=3 and we run the second for loop again which once again logs dojo two times before returning to the first for loop.*
        + *AND THAT’S HOW WE END UP LOGGING DOJO 6 TIMES.*
* **‘BREAK’ and use of ‘==’** 
  + **Var message = ‘CodingDojo’;**
  + **For(var i = 5; i <=20; i++)**
  + **{**
    - **If( i== message.length)**
    - **{**
    - **Break;**
    - **}**
    - **Console.log(i);**
  + **}**
    - the answer for this would be ***5,6,7,8,9***
      * *How we get there* is that we know that “i” is = to 5 and that it is currently less than 20 so we log it.
        + Cont. to log it until we get to 10 in which we know realize that ‘i’ = to message length, which is 10. And we now see that the next line reads *BREAK* which tells us to jump out of the code and end it all.
      * The **break statement can break loops and continues executing the code after the loop if there is any.**
        + ***continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.***
* **Getting more into FOR LOOPS AND IF STATEMENTS**
  + **for(num1 = 0; num1 <=3; num1++)**
  + **{**
    - **for(num2=0; num2<=2; num2++)**
    - **{**
      * **If(num1 == 5)**
      * **{**
      * **Console.log(‘coder’);**
      * **}**
    - **}**
    - **Console.log(‘ninja’);**
  + **}**
    - *The answer is* ***ninja, ninja, ninja, ninja*** *but I don’t understand why it is logged the 4th time.* OKAY I see where I got confused with the for loop here.
      * We know that num1=0 and is less than 3 and num2=0 which is less than 2 and goes to the if statement. But num1 does not equal to 5 so it goes back to the for statement and now num2= 1 yadda yadda yadda, num2= 2 breaks the loop and logs ninja
      * Now it goes back to num1=1 and conintues the loop as stated as before
      * It never actually runs console.log(‘coder’) since num1 never =5
        + *you could skip the whole middle part and just take the first line as the actual line of code to run…but as they say, “read the whole code.”*
* **Challenge:**
  + **Var array = [5,3,4,1]**
  + **For (var x = 0; x <array.length; x++)**
  + **{**
    - **For(var y = array.length -1; y >=x; y--)**
    - **{**
      * **if(array[x] > array[y]**
      * **{**
        + **Var temp = array[y];**
        + **Array[y] = array[x];**
        + **Array[x] = temp;**
      * **}**
    - **}**
  + **}**
  + **Console.log(array);**
    - The answer is ***[1,3,4,5]***
    - The reason being we have variable *array=[5,4,3,1]*
    - *We know var x= 0; x < 4; x++*
      * *Y=(4-1)🡺 y=3; y >=x; y—*
      * *If (array[x] > array[y]*
        + *If(array[0] > array[3])*
        + *If(5 > 1)*
      * *Variable temp = array[3]*
        + *Temp = 1*
      * *Array [3] = array[0]*
        + *1 = 5*
      * *Array [0] =temp (which is array of [3]*
        + *5 = 1*
    - So the answer is [5,4,3,1] always be aware of where the values are going. Work through it line-by-line.
* **USING THE WHILE FUNCTION**
  + **Challenge:**
  + **Var x = 1;**
  + **Var y = 48;**
  + **While(x<100)**
  + **{**
    - **If(x==y)**
    - **{**
      * **Break;**
    - **}**
  + **X++;**
  + **}**
  + **Console.log(x)**
    - This was kind of a trick question. The answer is simply **48**
      * We know that x = 1 and y = 48
      * And while x is less than 100 we continue to increment x bc x==y currently does not happen. We don’t even log anything because we are in the while loop and it will continue until x reaches 48 in which we break from the loop and log x which is 48.
* **DEALING WITH MULITPLE ARRAYS**
* **x= [[ ‘coding’, ‘boom’] , [‘dojo’, ‘yeah’]];**
* **console.log(x[0][0] + ‘-‘ + x[1][0]);**
  + the answer is **coding-dojo**
  + we need to pare down the variable x array
    - x[0] = coding, boom
    - x[1] = dojo, yeah
  + so now it wants us to log x[0] which we know = above and it’s also asking for the [0] in that same set which we know is coding.
  + Now we add the “-“ between them
  + And now it is asking for x[1] which I wrote above. But its asking for the [0] of that specific set which we know is dojo
  + So the answer would be coding-dojo
* **Challenge** 
  + **Var x = [[2,4,6,8], [12], [10,32,50]];**
  + **If(x[1][0] – x[2][0] == x[0][0]**
  + **{**
    - **Console.log(x[1][0] + x[2][1] + x[0][2]);**
  + **}**
  + **Else**
  + **{**
    - **Console.log(x[0][0] + x[1][0] + x[0][3]);**
  + **}**
    - Just like the 1st problem it just mixes in if and else statements to confuse you. But as before, parse down the arrays to something more digestible and go on from there.
      * The answer is **50**