JS 10

At this point, we notice one thing about our JS code. We use "**document.getElementById()**" alot. If we use it alot, we will often misspell it alot. Is there a way to make it easier for us? The answer is to store the data and then use it later on. We can use a variable, which can **ONLY** store 1 data for each variable. **For a large amount of data, we will use arrays instead, which stores many data per array.**

In this session, **we use arrays to hold the id of the different div blocks**. This makes our ribbon effect much easier to implement.

1. array 2. indexes 3. array manipulation

==========

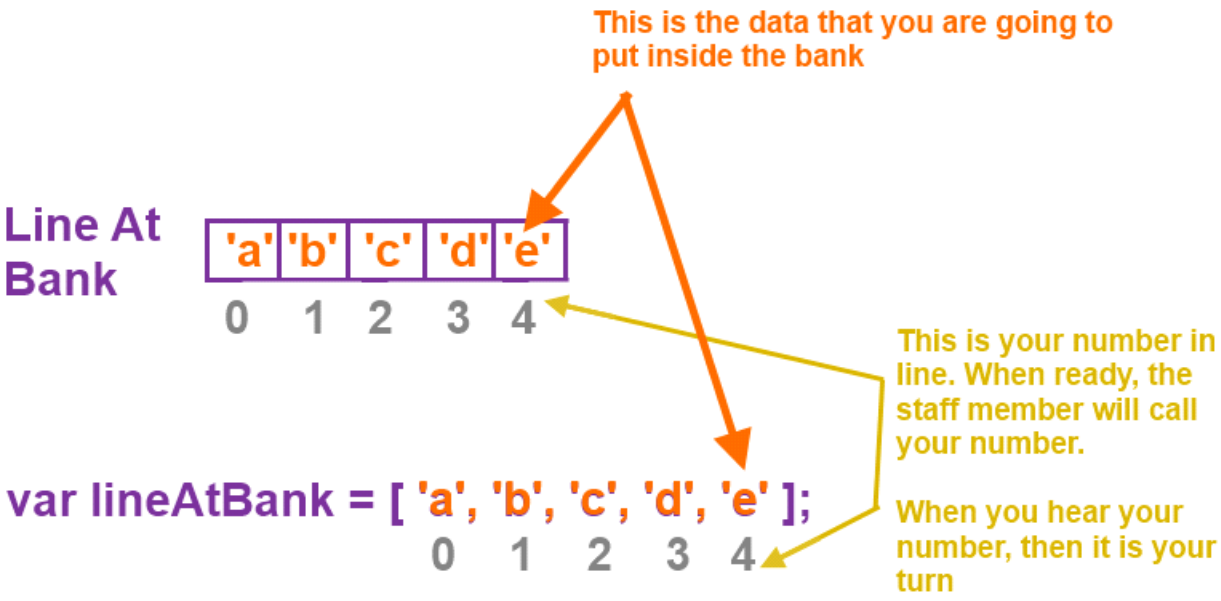
0. An array is a group of neighboring memory elements

0.1. Situation

There are many situations where a large amount of data is brought into our domain. The good news is that the large amount of data means we have more information to help us understand the problem statement. Unfortunately, the bad news is that there is **TOO MUCH** information.

The question now is: how do we organize so much information? The solution is to group all of the information together into an array. For example, when going to the bank or buying groceries, we wait in line for a staff member to help us. **An array is basically a line.**

When a staff member is done helping one person in line, the staff member will move onto the next position and help the person in that position. So, if the staff member is done helping the person in position 0, the staff member will then move to the next position in the line (the next position after 0 is position 1) and help the person in position 1. **The position that the staff member is on is called the index.**



Notice that an array is a group of neighboring memory elements and the word "**neighboring**" means that each memory element within the array is in sequential order (0, 1, 2, 3, and etc). The order **ALWAYS** starts at **number 0**.

With an array, we access the array by using a special format.

arrayName**[**index**]**;

The **total length** of the array can be found by using arrayName**.**length.

The **first index** is **always 0** and the **last index** is arrayName**.**length - 1

lineAtBank**[**0**]**

lineAtBank**.**length - 1

lineAtBank**.**length

0.2. access group using array name

Since all memory elements are in the same group, we can move all memory elements of the group by simply using the name of the array.

var **lineAtBank** = ['a', 'b', 'c', 'd', 'e'];

var **myArr** = **lineAtBank**;

The code above transfers the **ENTIRE CONTENT** of the array named **lineAtBank** into the array named **myArr**. The key operation is the assignment operator, which is used to transfer data from the right side of the assignment operator to the left side of the assignment operator.

0.3. access single element using square bracket

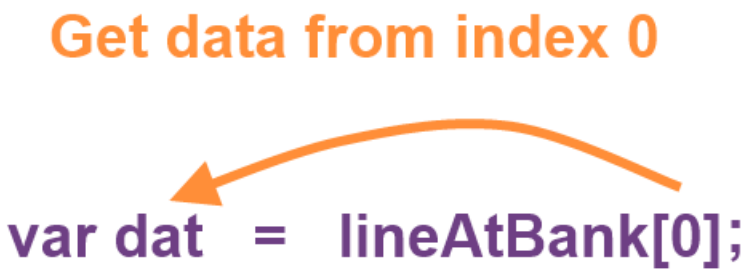
What if we want to access **ONLY A SINGLE MEMORY ELEMENT** within the array? **The answer is to use square brackets, which look like this [ ]**

lineAtBank**[**1**]** = 'v';

The **square brackets** are used to select **ONLY A SINGLE MEMORY ELEMENT** within the array. Inside the square brackets is the index (ie. inside the square brackets is the position in the line we are currently on).
a. The code above uses square brackets to select **index 1** of the array ➔ the code above means we are at **position 1** and putting the data 'v' into **position 1.**

After the line of code above is finished executing, the content of the array named dataGroup is updated to be `var lineAtBank = [ 'a', 'v', 'c', 'd', 'e' ];`

0.4. get data from a single array index



0.5. put data into a single array index

0.6. digital key

All of the examples from above put a number inside the square bracket. This is called a **hardcoded index**. We can't change the index because we are not using a variable.

lineAtBank[0]

Can we replace the number with a variable and make a generic index? The answer is yes and using variable as a generic index makes the variable a digital key.

```
var pos = 0;

lineAtBank[pos] = 'v';

pos +=1;

lineAtBank[pos] = 'z';
```

Explanation

The variable pos has the value of 0 stored inside of it. The code above puts the data 'v' into the memory element at the index found in variable **`pos`**. So, we are putting the data 'v' into the memory element at index 0. Since we are using the variable **`pos`** as an index, then **`pos`** becomes a digital key.

Next, we use the special ability of a variable to retain information and use this ability to perform index manipulation. Variables store data and retains data until new data is loaded into it. We execute the line of code **`pos += 1`**, which add 1 to the current value of **`pos`**. The variable **`pos`** has the value of 0 inside of it and we add 1 to it.

So now, 0 + 1 = 1. We then store the new value of 1 into **`pos`** using the assignment operator. The variable **`pos`** now has the value of 1 stored inside of it. We then use **`pos`** at an index and put the data 'z' into memory element at index 1.

0.7. reading a "for" loop using an array

```
var sum    = 0;
var numArr = [ 1, 2, 10, 11, 50, 100 ];
// -- position     0  1  2   3   4   5

for( var i = 0; i < numArr.length; i++ )
{
    sum += numArr[i];
}
// -- next
return sum/numArr.length;
```

Position 0 has the data 1 store inside of it

begin
a. begin only happens once

end

jump by
a. if not at the end, then jump by this much

loop body

i++ --> jump by 1 position
i+= 2 --> jump by 2 position

Reading a "for" loop

1. set i = 0
2. check if at the end
   Yes, go to line next and continue
   No, run line of code in loop body
3. jumpy by i++

Array Review

var myTabs = ["Home", "Projects", "Resume", "College"];

1. what is the first index number?
2. what is the data stored inside index 0?
3. what is the length of the array? Give a math equation instead of a number.

4. how do I select **ONLY ONE** array element?

5. write code to put the data "HTML" into the array at index 1

6. write code to

6.1. get the data from the array at index 2.

6.2. make a new variable called “tabData”

6.3. load the data from point 6.1 into a variable called "tabData"

var myTabs = [0, 10, 1, 10, 3, 4, 5];

7. write code to

7.0. create a variable and name it “myData” and load the data “0” into it.

7.1. create a variable and name it "**key**" and load the data "0" into it

7.2. use the variable named "**key**" to **ONLY SELECT ONE** array element. Next, load the data from the array at index "**key**" into a variable named "**myData**"

7.3. move onto the next array index using the increment operator (ie. plus 1) and then repeat step 7.2.

7.4. if we start at index 0 and want to load the data “**CN**” into every index of the array, how many times do we have to write code to increment the digital key?

Continue to the next page

1. Write JS Code In Between <script> </script>. Write ALL of it!!! Color code is for explanation

```
var eventArr = [];
var tabArr   = [];

function init_array()
{
      eventArr[0] = document.getElementById("urgent_events");
      eventArr[1] = document.getElementById("high_events");
      eventArr[2] = document.getElementById("theme_selection");

      tabArr[0] = document.getElementById("t0");
      tabArr[1] = document.getElementById("t1");
      tabArr[2] = document.getElementById("t2");

      alert( "done with init_array" );

}
```

Explanation

We are creating two arrays and they are named "**eventArr**" and "**tabArr**". Next, we use the assignment operator to load empty data into both arrays.

1. if we see square brackets and there is nothing inside the square brackets, then we are creating an empty array

Look at the code in blue and notice that they have the same index of **0 (ie. the same position of 0)**. **This means that they are linked together.** If I click on the HTML element with id "**t0**", the "**urgent_events**" will be displayed.

On the other hand, if I click on the HTML element with id of "**t1**", which event is displayed? The chosen element is in index **1** (ie. the chosen element is in position 1). What **else** is also in the position of index **1**? That would be "**high_events**". Since they both have the same index number, they are linked together. So, when I click on the HTML element with id of "**t1**", the result is that "**high_events**" is displayed.

Inside the function definition of "**init_array()**", notice that we are using the square brackets. Remember that square brackets are used to select **ONLY ONE ELEMENT** of the array. Inside the square brackets is the index (ie. inside the square brackets is the index we are selecting).

At first, we created an empty array with the code

```
var eventArr = [];
var tabArr   = [];
```

Next, we **PUT DATA INTO THE ARRAY** by writing the line of code below. Scroll back up to chapter 0.4 and 0.5 to review the difference between **putting in data** and **getting data out**.

```
eventArr[0] = document.getElementById("urgent_events");
eventArr[1] = document.getElementById("high_events");
eventArr[2] = document.getElementById("theme_selection");



tabArr[0] = document.getElementById("t0");
tabArr[1] = document.getElementById("t1");
tabArr[2] = document.getElementById("t2");
```

2. Write JS Code In Between <script> </script>.

```
// -- tCode means tab code
function tabSel( tCode )
{


    alert( "tCode: " + tCode );
    alert( eventArr[0] );

    for( var i = 0; i < eventArr.length; i++ )
    {
        //alert( "id: " + eventArr[tCode].getAttribute( "id" );

        eventArr[i].style.display = "none";
        tabArr[i].style.backgroundColor = "#e0e0e0";
    }

    eventArr[tCode].style.display = "block";
    tabArr[tCode].style.backgroundColor = "#1E90FF";


}
```

Continue on next page for “Explanation”

Explanation

The computer can only run one line of code at a time. THIS IS WHY THE CODE BELOW IS NOT ALLOWED AND WON'T WORK.

```
eventArr[0,1,2,3,4].style.display = "none";   // -- selecting too many indexes
```

Since the computer can only run one line of code at a time, we can only **SELECT ONE ARRAY ELEMENT AT A TIME**. The code below **ONLY SELECTs ONE ARRAY ELEMENT and that is why it works.**

```
eventArr[i].style.display = "none";       // -- only 1 index, good
```

In the code above, we are using a "for" loop to put data into the array.

We start at index 0 and we know this because the variable i is our digital key and has the value 0. We put data into the array using the square bracket and then the digital key of i.

"for" loop. Follow the steps
0. create a var called "i" and set it to 0
1. check if we are at the end
   by using eventArr.length
2. if we are NOT at the end of the line,
   then do the work
3. move onto the next person in line

```
// -- tCode means tab code
function tabSel( tCode )
{

    alert( "tCode: " + tCode );
    alert( eventArr[0] );
              0          1                  3
    for( var i = 0; i < eventArr.length; i++ )
    {
        //alert( "id: " + eventArr[tCode].getAttribute( "id" );

2       eventArr[i].style.display = "none";
        tabArr[i].style.backgroundColor = "#e0e0e0";
    }

    eventArr[tCode].style.display = "block";
    tabArr[tCode].style.backgroundColor = "#1E90FF";


}
```

This code loops through the array and resets
1. the display to "none" to make it hidden
2. backgroundColor to "#e0e0e0"

The parameter tCode is a var that holds the block to choose
1. we use this tCode as a digital key
2. we use the tCode to choose one to be visible and change the backgroundColor to #1E90FF

HTML Challenge

1. create 2 more table row headers ➔ `<th> </th>`

    1.1. the first table row header has id of "**t3**". Next put "**user input**" in between the open and closing tags

    1.2. the second table row header has id of "**t4**". Next put "**home**" in between the open and closing tags

2. create 2 more div blocks

    2.1. the first div has an id of "**search_bar**"

    2.2. the second div has an id of "**home_page**"

JS Challenge

1. add onto the function definition of "`function init_array()`" and make it do the following.
    a. "**t3**" is linked to "**search_bar**", what index should be used?
    b. "**t4**" is linked to "**home_page**", what index should be used?

```
function init_array()
{
      eventArr[0] = document.getElementById("urgent_events");
      eventArr[1] = document.getElementById("high_events");
      eventArr[2] = document.getElementById("theme_selection");

      tabArr[0] = document.getElementById("t0");
      tabArr[1] = document.getElementById("t1");
      tabArr[2] = document.getElementById("t2");

      alert( "done with init_array" );

}
```

Continue to the next page

JS Challenge 2

Go back to vuongducnguyen.com and click on JS 1 website. The code for JS Challenge 2 will be written on the JS 1 website. Inside the function main(), do the following

1. Declare a digital key named i and load the data 0 into i
2. Declare an array named "myNums" and load the data [0, 3, 10, 12] into "myNums"
3. Write a "for" loop that **begins** at position 0, **ends** at the total length of the array, and **jumps** by 1 position

    3.1. inside the body of the "for" loop, user alert to show the number stored at that position in the array "myNums"

JS Challenge 3

Go back to vuongducnguyen.com and click on JS 1 website. The code for JS Challenge 2 will be written on the JS 1 website. Inside the function main(), do the following

4. Declare a digital key named n and load the data 0 into n
5. Declare an array named "ageOC" and load the data [11, 7, 9, 10, 13, 14, 8, 13, 6, 9] into "ageOC"
6. Write a "for" loop that **begins** at position 0, **ends** at the total length of the array, and **jumps** by 2 positions

    3.1. inside the body of the "for" loop, user alert to show the number stored at that position in the array "ageOC"

JS Challenge 4

How is JS Challenge 2 different from JS Challenge 3?

1. When does JS Challenge 2 end? When does JS Challenge 3 end?
2. How many numbers are printed in JS Challenge 2? How many numbers are printed in JS Challenge 3?