**COMM644  
Assignment 5**

**Practice with Arrays (20 points)**

1. Create a string array that contains your five favorite movies. Then, use the console to display the second movie in your array.
2. Declare an array called movies using the function constructor method. Add the length of 5 into the constructor. Then, assign one of your favorite movies to each index in the array until you have 5 total movies in your array. Then, use the console to display the first movie in your array.
3. Copy your code from step 2. Add a new movie into the 5th position within your array. Then, use the console to display the length of the array. You should now have 6 total movies stored in the array.
4. Declare an array called movies using literal notation. Then, assign one of your favorite movies to each index in the array until you have 5 total movies in your array. Now, use the delete operator to remove the first movie in the array. Use the console to display the contents of the array.
5. Declare an array called movies using literal notation. Then, assign one of your favorite movies to each index in the array until you have 7 total movies in your array. Now, use a for loop to iterate through the array and display each movie within the console window.
6. Copy the code from step 5. Now, use a for-in loop to iterate through the array and display each movie within the console window.
7. Copy the code from step 5. Using the for-in loop to iterate through the array, display each movie within the console window in a sorted view.
8. Copy the code from step 5. Under the existing array, create a new array called leastFavMovies. Populate the array with the 3 movies that you regret watching. Display both arrays within the console window so that it’s formatted to look like this (note the spacing, you must include that too):  
     
   Movies I like:  
     
   Movie 1  
   Movie 2  
   Movie 3  
   …  
     
   Movies I regret watching:  
     
   Movie 1  
   Movie 2  
   Movie 3  
   …
9. Copy the code from step 8. Now, use the concat() method to merge the two arrays together into a single array called movies. Use the console window to display the list in reverse sorted order.
10. Copy the code from step 9. Use an array function to return just the last item in the array and display it within the console window.
11. Copy the code from step 10. Remove the previous method and this time use a method to return just the first item in the array and display it within the console window.
12. Programmatically retrieve the movies in your array that you do not like and return their indices. Then, using those indices, programmatically add movies that you do like.
13. Create two associative arrays called employee1 and employee2 respectively. Each array should have an employeeid, name, title, department, and whether or not the employee is a current employee of the company. Now, add both of those associative arrays to a new array called employees. Then, programmatically display the name of the second employee within a console window.
14. Copy the code from step 13. Modify the code slightly so that you’re using a loop to iterate through the array and print out the names of both employees.
15. Copy the code from step 14. Now add a third employee and set his employment status to false. Then, programmatically loop through the array but don’t display an employee if they aren’t a current employee of the company. Basically, only the first two should appear in the console window.
16. Create a multi-dimensional array that contains your 5 favorite movies and their ranking from 1-5. The array should look something like this:   
      
    movies = [["Movie 1", 1], ["Movie 2", 2], ["Movie 3", 3], ["Movie 4", 4], ["Movie 5", 5]];  
      
    Now, loop through the array and filter out and display only the movie names. You must use the filter() method and you’ll need to filter out the names by their primitive data type.
17. Create a string array called employees using literal notation and populate the array with several employee names. Then, create an anonymous function called showEmployee. The function should accept a parameter. Call this function, passing in the employees array into the function as a parameter. Make sure to display the result in the console window. Within the function, loop through the passed in array and display the result so that it looks exactly like this in the console window:  
      
    Employees:   
      
    ZAK   
    JESSICA   
    MARK   
    FRED   
    SALLY
18. Write a JavaScript function to filter false, null, 0 and blank values from an array.  
    Test Data: window.console.log(filterValues([58, '', 'abcd', true, null, false, 0]));  
    Expected Result: [58, "abcd", true]
19. Write a JavaScript function to get a random item from an array. So if I create a numeric array with 10 numbers and then pass that array into my function, the function should randomly return one of those numbers.
20. Write a JavaScript function to get the largest number from a numeric array.

**The Product Inventory Management System (30 points)**

In this part of the assignment you will use multi-dimensional arrays to build a small inventory management system. Here are the specifics:

* The primary array will be “inventory”.
* The nested arrays (products) will each contain elements that represent sku (number), product (string), quantity (number), and cost (floating point number).
* You should have at least 5 products in your inventory.
* Similar to lab 9, your application should display a title in the console window along with a command menu that features commands to **view all products**, **update stock**, and **exit the program**.
* When the user select **view**, rather than displaying each product in a numeric list like you did in lab 9, you should display the details for each product, ordered by sku number. Here’s an example of what the user should see if they choose the view command:  
    
  2233 Hat (12) $14.99  
  3223 Socks (36) $9.99  
  4824 Shirt (10) $15.99  
  6343 Jeans (22) $39.99  
  9382 Jacket (5) $49.99
* If the user selects the **update** command, a prompt should appear that allows the user to enter a sku number. Once the user enters a correct sku number a second prompt should appear that allows the user to enter a new stock quantity. The quantity should then update the product of the sku number entered.
* If the user selects the **exit** command, the program should be terminated.
* All data should be persisted using Web Storage. If the user closes the browser and then reopens it, all of the data should be retained from the previous session.
* Make sure to validate all inputs. If the user enters a command that’s not valid, the application should be able to handle that. Similarly, if the user enters a bad sku number, or text instead of a number when updating stock, the application should be able to handle that too.