1. Select **five methods** that can be used on an Array and describe the following for each:

1) what the method signature is

2) what the method does

3) why would this method be useful (how could you use it)?

1. array.push()

This method in array allows to add items to the end of the array. It is simple to use and we can use any addition in this method. For example:

```
Let myArray = [1, 2, 3];

myArray.push(4);

console.log(myArray); //output: [1, 2, 3, 4]
```

2. array.unshift()

Unlike the previous method that added items to the end of the array, this method adds items to the beginning of the array. For example:

```
Let myArray = [1, 2, 3];

myArray.unshift(4);

console.log(myArray); //output: [4, 1, 2, 3]
```

3. array.splice()

This method is very beneficial in which it can do everything, such as inserting, removing, or even replacing elements at any specific position in the array. While there are other delete methods in the array that might just remove the item without occupying the freed space, array.splice will take care of the freed space. For example:

```
let array = ["I", "am", "Persian"];

array.splice(1, 1); // from index 1 remove 1 element

alert(array ); // ["I", "Persian"]
```

4. array.slice()

This method might look similar in index to the previous one, but in reality it is different. The slice method is simpler than the splice method is used to extract a section of an array and returns a new array with the extracted elements. It is a useful method for extracting a section of an array without modifying the original array. It can be used to create a new array with specific elements or to copy an entire array. For example:

```
let myArray = [1, 2, 3, 4, 5];
let newArray = myArray.slice(1, 4);
console.log(newArray); // Output: [2, 3, 4]
```

5. array.concat()

This method creates a new array that includes values from other arrays and additional items. It accepts any number of arguments including numbers or values. The syntax is : array.concat(arg1, arg2, …). For example:

```
let array = [1, 2];

// create an array from: array and [3,4]
alert(array.concat([3, 4]) ); // 1,2,3,4

// create an array from: array and [3,4] and [5,6]
alert(array.concat([3, 4], [5, 6]) ); // 1,2,3,4,5,6
```

Reference: https://javascript.info/array-methods

2. What is the difference between == and ===?

Although both are two comparison operators in JavaScript, Double equals (==) is the equal comparison operator, which compares 2 variables to see if they hold the same values, but it doesn't consider their types. However, triple equals (===) compare the variables and becomes true only if the variables have the same type and value. For example:

1 == '1' // true

In JavaScript, == coerces string into numbers and shows them as true. But,

1 === '1' // false

Because === shows that string '1' and number 1 are not the same in type.

Reference: https://www.java67.com/2013/07/difference-between-equality-strict-vs-operator-in-JavaScript-Interview-Question.html