(1.) **Sigma** - Implement function sigma(num) that, given a number, returns the sum of all positive integers up to the given number (inclusive). Ex: sigma(3) = 6 (or 1+2+3); sigma(5) = 15 (or 1+2+3+4+5).

```
function sigma(num){
  var sum = 0;
  for(var i = 0; i<=num; i++){
    sum = sum+i;
    console.log(sum);
  }
  return num;
 }
sigma(8);
```

(2.) **Factorial** - Write a function factorial(num) that, given a number, returns the product (multiplication) of all positive integers from 1 up to the given number (inclusive). For example, factorial(3) = 6 (or 1*2*3); factorial(5) = 120 (or 1*2*3*4*5).

```
function factorial(num){
  var multiply=1;
  for(var i = 1; i<=num; i++){
    multiply=multiply*i;
  }
  num = multiply
  return num;
}
factorial(5);
```

(3.) **Fibonacci** - Create a function to generate Fibonacci numbers. In this famous mathematical sequence, each number is the sum of the previous two, starting with values 0 and 1. Your function should accept one argument, an index into the sequence (where 0 corresponds to the initial value, 4 corresponds to the value four later, etc). Examples: fibonacci(0) = 0 (given), fibonacci(1) = 1 (given), fibonacci(2) = 1 (fib(0)+fib(1), or 0+1), fibonacci(3) = 2 (fib(1) + fib(2)3, or 1+1), fibonacci(4) = 3 (1+2), fibonacci(5) = 5 (2+3), fibonacci(6) = 8 (3+5), fibonacci(7) = 13 (5+8). Do this without using recursion first. If you don't know what a recursion is yet, don't worry as we'll be introducing this concept in Part 2 of this assignment.

```
 function Fibonacci(num){
  var a = 1;
  var b = 0;
  var fib = 0;
  for(var i=1; i<=num; i++){
    fib=a+b;
    a=b;
    b=fib;
  }
  return fib
}
console.log(Fibonacci(9));
```

(4.) **Array: Second-to-Last:** Return the second-to-last element of an array. Given [42, true, 4, "Liam", 7], return "Liam". If array is too short, return null.

```
function secondLast(arr){
  if(arr.length == 1){
    return null;
  }
  else{
    return arr[arr.length-2];
  }
}
secondLast([42, true, 4, "Liam", 7]);
```

```
(5.) function nToLast(arr,num){
  if(arr.length < num){
    return null;
  }
  else{
    return arr[arr.length-num];
  }
}
nToLast([5,2,3,6,4,9,7],3);
```

(6.) **Array: Nth-to-Last:** Return the element that is N-from-array's-end. Given ([5,2,3,6,4,9,7], 3), return 4. If the array is too short, return null.

```
function secondLarge(arr){
var max = arr[0];
var secondMax = arr[0];

for(var i = 1; i<arr.length; i++){
  if(arr.length<2){
    console.log(null);
    }
  if(max<arr[i]){
    secondMax = max;
    max = arr[i];
    }
  else if(secondMax<arr[i]){
    secondMax = arr[i];
    }
  }
  return secondMax;
}
secondLarge([3,17,1,16,6,14,5]);
```

(7.) **Array: Second-Largest**: Return the second-largest element of an array. Given [42,1,4,3.14,7], return 7. If the array is too short, return null.

```
function makeDouble(arr){
  var newArr = [];
  for(var i = 0; i<arr.length; i++){
    newArr.push(arr[i]);
    newArr.push(arr[i]);
  }
  return newArr;
}
makeDouble([4, "Ulysses", 42, false]);
```

PART 2

```
function fibonacci(n){
  if(n===0 | n==1){
    return(n);

  }
  return fibonacci(n-2)+fibonacci(n-1);
}

n = fibonacci(7);
console.log(n);
```