https://thatjsdude.com/interview/js1.html

1. check Prime

Question: How would you verify a prime number?

Answer: a prime number is only divisible by itself and 1. So, i will run a while loop and increase by 1. (look at the code example. If you dont get it. this is not your cake. do learn javaScript basics and come back.)

```
function isPrime(n){
    var divisor = 2;

    while (n > divisor){
      if(n % divisor == 0){
       return false;
      }
      else
       divisor++;
   }
  return true;
 }

> isPrime(137);
   = true
> isPrime(237);
   = false
```

Interviewer: Can you make this better?

You: yes. the divisor are increased 1 at a time. after 3 i can increase by 2. if a number is divisible by any even number, it will be divisible by 2.

Extra: if you dont have to increase the divisor up to the number. you can stop much earlier. let me explain it in the following steps (just seat back and read as many times as needed)

- **step-1:** Any number will not be divisible by a number bigger than half of it. for example, 13 will never be divisible by 7, 8, 9 .. it could be as big as half of it for even number. for example, 16 will be divisible by 8 but will never be by 9, 10, 11, 12...
- **Decision:** a number will never be divisible by a number bigger than half of its values. So, we dont have to loop 50%
- **step-2:** Now, if a number is not divisible by 3. (if it is divisible by 3, then it wouldn't be a prime number). then it would be divisible any number bigger than the 1/3 of its value. for example, 35 is not divisible by 3. hence it will be never divisible by any number bigger than 35/3 will never be divisible by 12, 13, 14 ... if you take an even number like 36 it will never be divisible by 13, 14, 15
- **Decision:** a number could be divisible by numbers 1/3 of its value.
- **step-3:** For example u have the number 127. 127 is not divisible by 2 hence you should check upto 63.5. Secondly, 127 is not divisible by 3. So, you will check up to 127/3 approximately 42. It is not divisible by 5, divisor should be less than 127/5 approximately 25 not by 7. So, where should we stop?
- **Decision:** divisor would be less than Math.sqrt (n)

try 2

Don't worry! if you didnt get it. just leave it. If you are not applying for a Research post, you would be alright.

```
function isPrime(n)
 {
   var divisor = 3,
       limit = Math.sqrt(n);

  //check simple cases
   if (n == 2 || n == 3)
     return true;
  if (n % 2 == 0)
     return false;

   while (divisor <= limit)
  {
      if (n % divisor == 0)
        return false;
     else
        divisor += 2;
  }
  return true;
 }

> isPrime(137);
   = true
> isPrime(237);
   = false
```

2. Prime Factors

Question: How could you find all prime factors of a number?

Answer: Run a while loop. start dividing by two and if not divisible increase divider until u r done.

```
function primeFactors(n){
   var factors = [],
       divisor = 2;
```

```
 while(n>2){
      if(n % divisor == 0){
         factors.push(divisor);
        n= n/ divisor;
      }
     else{
       divisor++;
      }
  }
  return factors;
}

 > primeFactors(69);
   = [3, 23]
```

Interviewer: What is the run time complexity? can you make this better

You: this is O(n). you can increase divisor by 2 form divisor = 3. Because, if a number is divisible by any even number it would divisible by 2. Hence, you dont need to divide by even numbers. Besides, you will not have a factor bigger than half of its value. if you want make it complex use tough concept in no. 1 (try-2. if u get it)

3. Fibonacci

Question: How do get nth Fibonacci number?

Answer: I create an array and start from iterate through.

Fibonacci series is one of the most popular interview question for beginners. so, you have to learn this one.

try 1

```
function fibonacci(n){
   var fibo = [0, 1];


   if (n <= 2) return 1;


   for (var i = 2; i <=n; i++ ){
    fibo[i] = fibo[i-1]+ fibo[i-2];
  }


 return fibo[n];
```

```
}


> fibonacci(12);
  = 144
```

Interviewer: What is the run time complexity?

you: O(n)

Interviewer: can you make it recursive?

try 2

```
function fibonacci(n){
 if(n<=1)
    return n;
 else
   return fibonacci(n-1) + fibonacci (n-2);
}


> fibonacci(12);
  = 144
```

Interviewer: What is the run time complexity?

You: $O(2^n)$. detail about complexity

4. Greatest Common Divisor

Question: How would you find the greatest common divisor of two numbers?

```
function greatestCommonDivisor(a, b){
   var divisor = 2,
       greatestDivisor = 1;


  //if u pass a -ve number this will not work. fix it dude!!
   if (a < 2 || b < 2)
     return 1;


   while(a >= divisor && b >= divisor){
    if(a %divisor == 0 && b% divisor ==0){
       greatestDivisor = divisor;
     }
```

```
    divisor++;
  }
   return greatestDivisor;
 }

> greatestCommonDivisor(14, 21);
  =7
> greatestCommonDivisor(69, 169);
  = 1
```

fancy algorithm

Sorry. can't explain it. As i myself dont understand it 80% of the times. my algorithm analysis instructor told about this and stole if from class note (i am a good student, btw!)

```
function greatestCommonDivisor(a, b){
    if(b == 0)
      return a;
    else
      return greatestCommonDivisor(b, a%b);
 }
```

Note: use your brain to understand it.

5. remove Duplicate

Question: How would you remove duplicate members from an array?

Answer: will start a while looping and keep an object/ associated array. If i find an element for the first time i will set its value as true (that will tell me element added once.). if i find a element in the exists object, i will not insert it into the return array.

```
function removeDuplicate(arr){
  var exists ={},
       outArr = [],
      elm;

  for(var i =0; i<arr.length; i++){
    elm = arr[i];
    if(!exists[elm]){
```

```
      outArr.push(elm);
        exists[elm] = true;
    }
  }
  return outArr;
}

> removeDuplicate([1,3,3,3,1,5,6,7,8,1]);
 = [1, 3, 5, 6, 7, 8]
```

6. merge two sorted array

Question: How would you merge two sorted array?

Answer: I will keep a pointer for each array and (read the code. and be careful about this one.)

```
function mergeSortedArray(a, b){
  var merged = [],
      aElm = a[0],
      bElm = b[0],
      i = 1,
      j = 1;

  if(a.length ==0)
    return b;
  if(b.length ==0)
    return a;
  /*
  if aElm or bElm exists we will insert to merged array
  (will go inside while loop)
  to insert: aElm exists and bElm doesn't exists
              or both exists and aElm < bElm
    this is the critical part of the example
*/
  while(aElm || bElm){
    if((aElm && !bElm) || aElm < bElm){
      merged.push(aElm);
```

```
      aElm = a[i++];
    }
    else {
      merged.push(bElm);
      bElm = b[j++];
    }
  }
  return merged;
}

> mergeSortedArray([2,5,6,9], [1,2,3,29]);
 = [1, 2, 2, 3, 5, 6, 9, 29]
```

7. swap number without temp

Question: How would you swap two numbers without using a temporary variable?

Answer: Waste time about thinking it. though u know the answer, act like you are thinking and take your time to answer this one.

```
function swapNumb(a, b){
    console.log('before swap: ','a: ', a, 'b: ', b);
    b = b -a;
    a = a+ b;
  b = a-b;
    console.log('after swap: ','a: ', a, 'b: ', b);
}

> swapNumb(2, 3);
    = before swap:  a:  2 b:  3
     = after swap:  a:  3 b:  2
```

bit manipulation: Sorry, i can't explain this to you. Kinjal Dave suggested logical conjunction to understand it. Waste 30 min there at your own risk.

```
function swapNumb(a, b){
  console.log("a: " + a + " and b: " + b);
   a = a ^ b;
```

```
  b = a ^ b;
  a = a ^ b;
  console.log("a: " + a + " and b: " + b);
}

> swapNumb(2, 3);
 = a: 2 and b: 3
 = a: 3 and b: 2
```

8. string reverse

Question: How would you reverse a string in JavaScript?

Answer: I can loop through the string and concatenate letters to a new string

try 1

```
function reverse(str){
  var rtnStr = '';
   for(var i = str.length-1; i>=0;i--){
     rtnStr +=str[i];
  }
   return rtnStr;
}

> reverse('you are a nice dude');
  = "edud ecin a era uoy"
```

Interviewer: You know concatenation performed well in modern browsers but becomes slow in older browsers like IE8. Is there any different way, you can reverse a string?

Answer: sure. i can use an array and also add some checking. if string is null or other than string this will fail. let me do some type check as well. Using this array is like using string buffer in some server side languages.

try 2

```
function reverse(str){
  var rtnStr = [];
 if(!str || typeof str != 'string' || str.length < 2 ) return str;

   for(var i = str.length-1; i>=0;i--){
```

```
        rtnStr.push(str[i]);
    }
    return rtnStr.join('');
}
```

Interviewer: What is the run time complexity?

You: O(n)

Interviewer: Can you make this better?

You: I can loop through half of the index and it will save little bit. (this is kind of useless, might not impress interviewer)

try 3

```
function reverse(str) {
    str = str.split('');
    var len = str.length,
        halfIndex = Math.floor(len / 2) - 1,
        revStr;
    for (var i = 0; i <= halfIndex; i++) {
        revStr = str[len - i - 1];
        str[len - i - 1] = str[i];
        str[i] = revStr;
    }
    return str.join('');
}
```

Interviewer: That works, but can u do it in a recursive way?

You: sure.

try 4

```
function reverse (str) {
    if (str === "") {
        return "";
    } else {
        return reverse(str.substr(1)) + str.charAt(0);
    }
}
```

try 5

Interviewer: Can you use any build in method to make it little cleaner?

You: yes.

```
function reverse(str){
   if(!str || str.length <2) return str;

  return str.split('').reverse().join('');
}
```

try 6

Question: Can you make reverse function as string extension?

Answer: I need to add this function to the String.prototype and instead of using str as parameter, i need to use this

```
String.prototype.reverse = function (){
 if(!this || this.length <2) return this;

  return this.split('').reverse().join('');
}

 > 'abc'.reverse();
  = 'cba'
```

9. reverse words

Question: How would you reverse words in a sentence?

Answer: You have to check for white space and walk through the string. Ask is there could be multiple whitespace.

```
  //have a tailing white space
//fix this later
//now i m sleepy
 function reverseWords(str){
 var rev = [],
       wordLen = 0;
   for(var i = str.length-1; i>=0; i--){
    if(str[i]==' ' || i==0){
      rev.push(str.substr(i,wordLen+1));
```

```
        wordLen = 0;
      }
      else
       wordLen++;
    }
    return rev.join(' ');
}
```

A quick solution with build in methods:

```
function reverseWords(str){
  return str.split(' ').reverse();
}
```

10. reverse in place

Question: If you have a string like "I am the good boy". How can you generate "I ma eht doog yob"? Please note that the words are in place but reverse.

Answer: To do this, i have to do both string reverse and word reverse.

```
function reverseInPlace(str){
    return str.split(' ').reverse().join(' ').split('').reverse().join('');
}

> reverseInPlace('I am the good boy');
= "I ma eht doog yob"
```

Interviewer: ok. fine. can you do it without using build in reverse function?

you: (you mumbled): what the heck!!

```
//sum two methods.
//you can simply split words by ' '
 //and for each words, call reverse function
 //put reverse in a separate function



 //if u cant do this,
```

```
//have a glass of water, and sleep
```

11. First non repeating char

Question: How could you find the first non repeating char in a string?

Answer: You must ask follow up questions.

Clarification: Is it case sensitive?

Interviewer: interviewer might say no.

Clarification: is it very long string or couple hundred?

Interviewer: Why does that matter

you: for example, if it is a very long string say a million characters and i want to check whether 26 English characters are repeating. I might check whether all characters are duplicated in every 200 letters (for example) other than looping through the whole string. this will save computation time.

Interviewer: For simplicity, you string is "the quick brown fox jumps then quickly blow air"

Clarification: (silly question) ascii or unicode.

```
function firstNonRepeatChar(str){
  var len = str.length,
      char,
      charCount = {};
  for(var i =0; i<len; i++){
    char = str[i];
     if(charCount[char]){
      charCount[char]++;
    }
     else
      charCount[char] = 1;
  }
  for (var j in charCount){
    if (charCount[j]==1)
        return j;
  }
}

>firstNonRepeatChar('the quick brown fox jumps then quickly blow air');
 = "f"
```

this has one problem. It is not guaranteed that for in loop will be executed in order.

12. remove duplicate char

Question: How will you remove duplicate characters from a sting?

You: This is very similar to first non repeating char. You will asks similar question. Is it case sensitive.

If interviewer says, this is case sensitive then life become easier you. If he says no. you can either use string.toLowercase() to make whole string lower. he might not like it. because return string will not posses the same case. So

```
function removeDuplicateChar(str){
   var len = str.length,
        char,
        charCount = {},
         newStr = [];
   for(var i =0; i<len; i++){
     char = str[i];
      if(charCount[char]){
        charCount[char]++;
      }
      else
       charCount[char] = 1;
  }
   for (var j in charCount){
    if (charCount[j]==1)
         newStr.push(j);
  }
  return newStr.join('');
 }


> removeDuplicateChar('Learn more javascript dude');
  = "Lnmojvsciptu"
```

Note:this has one problem. It is not guaranteed that for in loop will be executed in order.

For case insensitive: when u r setting property of charCount or increase counter, just make the char.toLowerCase(). or you can do something fancy with charCode (if you can deal with it.)

13. check palindrome

Question: How will you verify a word as palindrome?

Answer: if you reverse a word and it becomes same as the previous word, it is called palindrome.

```
function isPalindrome(str){
 var i, len = str.length;
 for(i =0; i<len/2; i++){
      if (str[i]!== str[len -1 -i])
        return false;
 }
 return true;
}

> isPalindrome('madam')
  = true
 > isPalindrome('toyota')
  = false
```

or you can use build in method

```
function checkPalindrom(str) {
      return str == str.split('').reverse().join('');
}
```

Similar: Find whether a string contains a contiguous palindromic substring in O(n) time. Can you solve the problem in O(1) time?

14. random between 5 to 7

Question: If you have a function that generate random number between 1 to 5 how could u generate random number 1 to 7 by using that function?

Answer: Very simple. think of some basic arithmetic and you will get it.

```
function rand5(){
  return 1 + Math.random() * 4;
}

function rand7(){
  return 5 + rand5() / 5 * 2;
}
```

15. missing number

Question: from a unsorted array of numbers 1 to 100 excluding one number, how will you find that number.

Explanation: You have an array of numbers 1 to 100 in an array. Only one number is missing in the array. The array is unsorted. Find the missing number.

Answer: You have to act like you are thinking a lot. and then talk about the sum of a linear series of n numbers = n*(n+1)/2

```
function missingNumber(arr){
  var n = arr.length+1,
    sum = 0,
  expectedSum = n* (n+1)/2;

    for(var i = 0, len = arr.length; i < len; i++){
      sum += arr[i];
  }

  return expectedSum - sum;
 }

> missingNumber([5, 2, 6, 1, 3]);
  = 4
```

Note: this one will give u missing one number in any array of length

16. Sum of two

Question: From a unsorted array, check whether there are any two numbers that will sum up to a given number?

Answer: Simplest thing in the world. double loop

```
function sumFinder(arr, sum){
  var len = arr.length;

  for(var i =0; i<len-1; i++){
     for(var j = i+1;j<len; j++){
          if (arr[i] + arr[j] == sum)
              return true;
     }
 }
```

```
    return false;
  }


> sumFinder([6,4,3,2,1,7], 9);
   = true
> sumFinder([6,4,3,2,1,7], 2);
   = false
```

Interviewer: What is the time complexity of this function

You: O(n^2)

Interviewer: Can you make this better

You: Let me see. I can have an object where i will store the difference of sum and element. And then when i get to a new element and if i find the difference is the object, then i have a pair that sums up to the desired sum.

try 2

```
function sumFinder(arr, sum){
 var differ = {},
      len = arr.length,
      substract;


 for(var i =0; i<len; i++){
     substract = sum - arr[i];


    if(differ[substract])
       return true;
     else
       differ[arr[i]] = true;
 }


   return false;
}


> sumFinder([6,4,3,2,1,7], 9);
   = true
> sumFinder([6,4,3,2,1,7], 2);
   = false
```

similar problem: find the maximum difference of two numbers in an array max difference

Even tougher Finding three elements in an array whose sum is closest to an given number

17. Largest Sum

Question: How would you find the largest sum of any two elements?

Answer: this is actually very simple and straight forward. Just find the two largest number and return sum of them

```
function topSum(arr){

  var biggest = arr[0],
      second = arr[1],
      len = arr.length,
       i = 2;

  if (len<2) return null;

  if (biggest<second){
    biggest = arr[1];
    second = arr[0];
  }

   for(; i<len; i++){

    if(arr[i] > biggest){
        second = biggest;
        biggest = arr[i];
     }
    else if (arr[i]>second){
        second = arr[i];
    }

  }
 return biggest + second;
}
```

18. Counting Zeros

Question: Count Total number of zeros from 1 upto n?

Answer: If n = 50. number of 0 would be 11 (0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100). Please note that 100 has two 0. This one looks simple but little tricky

Explanation: So the tick here is. if you have a number 1 to 50 the value is 5. just 50 divided by 10. However, if the value is 100. the value is 11. you will get by 100/10 = 10 and 10/10. Thats how you will get in the more zeros in one number like (100, 200, 1000)

```
function countZero(n){
  var count = 0;
 while(n>0){
   count += Math.floor(n/10);
    n = n/10;
 }
  return count;
}

> countZero(2014);
   = 223
```

19. subString

Question: How can you match substring of a sting?

Answer: Will use to pointer (one for string and another for the substring) while iterating the string. And will have another variable to hold the starting index of the initial match.

```
function subStringFinder(str, subStr){
 var idx = 0,
       i = 0,
       j = 0,
       len = str.length,
     subLen = subStr.length;

  for(; i<len; i++){

      if(str[i] == subStr[j])
         j++;
     else
        j = 0;

     //check starting point or a match
```

```
            if(j == 0)
              idx = i;
            else if (j == subLen)
              return idx;
        }

        return -1;
    }

> subStringFinder('abbcdabbbbbck', 'ab')
  = 0
> subStringFinder('abbcdabbbbbck', 'bck')
  = 9

  //doesn't work for this one.
> subStringFinder('abbcdabbbbbck', 'bbbck')
  = -1
```

Question: How can you fix the last problem?

Answer: figure out yourself.

20. Permutations

Question: How would you create all permutation of a string?

Answer: This could be a tough one based on you level of knowledge about algorithm.

```
function permutations(str){
    var arr = str.split(''),
        len = arr.length,
        perms = [],
        rest,
        picked,
        restPerms,
        next;

    if (len == 0)
        return [str];
```

```
    for (var i=0; i<len; i++)
    {
        rest = Object.create(arr);
        picked = rest.splice(i, 1);

        restPerms = permutations(rest.join(''));

        for (var j=0, jLen = restPerms.length; j< jLen; j++)
        {
            next = picked.concat(restPerms[j]);
            perms.push(next.join(''));
        }
    }
  return perms;
}
```

Explanation:

- **Idea:** Idea is very simple. We will convert the string to an array. from the array we will pick one character and then permute rest of it. After getting the permutation of the rest of the characters, we will concatenate each of them with the character we have picked.
- **step-1** First copy original array to avoid changing it while picking elements
- **step-2** Use splice to removed element from the copied array. We copied the array because splice will remove the item from the array. We will need the picked item in the next iteration.
- **step-3** [1,2,3,4].splice(2,1) will return [3] and remaining array = [1,2,4]
- **step-4** Use recursive method to get the permutation of the rest of the elements by passing array as string
- **step-5** Finally, concat like a+permute(bc) for each