

JS: Interview Algorithm

part -1: beginner

January 26, 2014



1. Verify a prime number?
2. Find all prime factors of a number?
3. Get nth Fibonacci number?
4. Find the greatest common divisor of two numbers?
5. Remove duplicate members from an array?
6. Merge two sorted array?
7. Swap two numbers without using a temp variable?
8. Reverse a string in JavaScript?
9. Reverse words in a sentence?
10. Reverse words in place?
11. Find the first non repeating char in a string?
12. Remove duplicate characters from a sting?
13. Verify a word as palindrome?
14. Generate random between 5 to 7 by using defined function.
15. Find missing number from unsorted array of integers.
16. Check whether any two numbers in an array sums to a given number?
17. Find the largest sum of any two elements?
18. Total number of zeros from 1 upto n?
19. Match substring of a sting?
20. Create all permutation of a string?

More Questions [CSS Interview Questions \(css.html\)](#), [HTML Interview Questions \(html.html\)](#)

if you are little more comfortable or claim to be comfortable with javascript, these questions would not be enough for you. more coming

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)

Show Explanation

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 (<http://stackoverflow.com/a/360773/1535443>)

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 it 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 (http://en.wikipedia.org/wiki/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');
= "Lnmojvscriptu"
```

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,
      subtract;

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

    if(differ[subtract])
      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
(<http://programmingpraxis.com/2011/04/01/maximum-difference-in-an-array/>)

Even tougher Finding three elements in an array whose sum is closest to an given number
(<http://stackoverflow.com/questions/2070359/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 two pointers (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('abbc dabbb bck', 'ab')
= 0
> subStringFinder('abbc dabbb bck', 'bck')
= 9

//doesn't work for this one.
> subStringFinder('abbc dabbb bck', '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

similar question

- display prime numbers up to n.
- Display number which is repeated for event times in an array
- Find non repeating character not case sensitive programming praxis
(<http://programmingpraxis.com/2013/04/30/first-unrepeated-character-in-a-string/>)
- count words in a string
- In an integer array, there is 1 to 100 number, out of one is duplicate, how to find ?

- implement substring of a string. also make it case sensitive
- Find a square of a number. but you can only use addition or subtraction but no multiplication or division Odd way to Square (<http://programmingpraxis.com/2013/02/26/an-odd-way-to-square/2/>)

medium level

- From two sorted array how would you find common number?
- From web page, how would u find similar words like rat, cat, bat and broom, groom etc.
- get first 100 character long string from a big message but dont cut the last word (word break problem)
- Find the max difference of elements from two sorted array with non duplicate integer elements
- Second symbol starting index Array of two symbol (<http://programmingpraxis.com/2013/03/12/an-array-of-two-symbols/#comments>)
- Determine if a positive number can be expressed as a sum of two cubes? (<http://programmingpraxis.com/2012/11/09/taxicab-numbers/>)
- 4SUM (<http://programmingpraxis.com/2012/08/14/4sum/>)
- from an array of integers find 10 numbers closest to a given number amazon (<http://programmingpraxis.com/2012/11/27/amazon-interview-question/>)
- find a rotation point of a sorted array.
- Finding three elements in an array whose sum is closest to an given number (<http://stackoverflow.com/questions/2070359/finding-three-elements-in-an-array-whose-sum-is-closest-to-an-given-number>)
- Write a function to find the nearest link on a webpage given the mouse x,y coordinates. career cup (<http://www.careercup.com/question?id=3538036>)

ref: Coding interview tips (<http://www.interviewcake.com/tips-and-tricks>)

more problems (<http://projecteuler.net/problems>)

Need more: CSS Interview Questions (css.html), HTML Interview Questions (html.html)

Express anger!

Feel free to express your anger (sorry u have to use facebook). also find out mistakes (technical, wrong answer, spelling, grammar, sentence, whatever), let ur dude learn and grow.

39 comments



Add a comment as Jayakumar Srinivasan

Top comments



Khan Md 1 year ago - [JavaScript \(Discussion\)](#)

Rock the Interview. 20+ JavaScript algorithm for Front End Developers (part-1: beginners) [#javascript](#) [#frontenddeveloper](#) [#webdevelopment](#) [#html5](#) [#html5css3](#)

+25 1



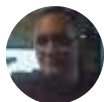
Rommel Badua 1 year ago **+1**

This is very helpful, thanks for sharing!



Khan Md 1 year ago

You are welcome **+Rommel Badua**



Jason Lough 8 months ago - Shared publicly

Good to learn with, but theres some (interesting!) problems.

on #17 largest sum, why not just use `array.sort(sortFunc)`, then `reverse()` the array. The two biggest would then be `arr[0]` and `arr[1]`.

1 · Reply



kyle hamilton 6 months ago

why reverse? the two biggest will be:
`arr[arr.length-1]` and `arr[arr.length-2]`



Jason Lough 6 months ago

That would work. My way the indexes of the biggest are always the same (0 and 1). Your way may have better performance though (no expensive `reverse()` call).



Madhav Patel via Google+ 7 months ago - Shared publicly

<http://www.thatjstdude.com/interview/js1.html>

1



Khan Md 1 year ago - [Web Development \(Discussion\)](#)

Rock JavaScript Interview. 20+ interview Algorithm in JavaScript for Front End Developers. (part-1: beginners) [#frontenddevelopment](#) [#javascript](#) [#webdevelopment](#)

+7 1



Khan Md 1 year ago - [Web Developers, Web Designers, Web Coding \(Questions & Answers\)](#)


Rock your Interview. 20+ JavaScript Interview Algorithm for front end developers. (part-1: beginners) [#webdevelopment](#) [#webdesign](#) [#javascript](#)


+6 1

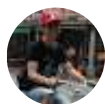


Khan Md 1 year ago

+Harington Darkholme That's a great point. I would be very happy if you could

 **+Herrington Darkholme** thats a great point. I would be very happy, if you could suggest improvement of one or more questions in JS way. that will help me to grow.

 **Khan Md** 1 year ago
+Herrington Darkholme implicit global is a really good catch. Thank you very much for this. I corrected it. I will appreciate if you can provide more suggestion to improve. Don't worry about altercation. I am here to learn and grow. Feel free to comment or provide any kind of feedback.



Kos Huang 1 year ago - [APP & Web 技術交流 \(分享與閒聊\)](#)



Khan Md originally shared this
Rock the Interview. 20+ JavaScript algorithm for Front End Developers (part-1: beginners) [#javascript](#) [#frontenddeveloper](#) [#webdevelopment](#) [#html5](#) [#html5css3](#)

+2



Khan Md 1 year ago - [Google Geeks \(Other Google Topics\)](#)

20+ JavaScript Interview Algorithm for Front End Developers. (part-1: beginners). Rock your JavaScript Interview

+1



Khan Md 1 year ago - [node.js \(Discussion\)](#)

20+ JavaScript Interview Algorithm for Front End Developers. (part-1: beginners). Rock JavaScript Interview.

+2



Khan Md 1 year ago - [CSS Community \(Discussion\)](#)

Rock JavaScript Interview Questions. 20+ JavaScript Algorithm for Front End Developers

+1



Khan Md 1 year ago - [Web Design \(Discussion\)](#)

20+ JavaScript Interview Algorithm for beginners.

+1



Khan Md 1 year ago - [HTML5 \(Share\)](#)

Rock JavaScript Interview. 20+ interview Algorithm for Front End Developers.
[#javascript](#) [#frontenddeveloper](#) [#webdevelopment](#) [#html5](#) [#html5css3](#)

+4

View all 3 replies



Christopher Cook 1 year ago **+1**

The most cost efficient way (for your employers perspective) to find an answer for a common algorithm is to Google it, rather than work out your own version :)



Khan Md 1 year ago

+Christopher Cook Thats absolutely right after u r hired. Unfortunately, in the interview at the whiteborad, u dont have google



Ser Bir 11 hours ago - Shared publicly

solution for #18 is wrong. It returns the same result for 100 and 109

1

· Reply



Denis Savenok via Google+ 2 months ago (edited) - Shared publicly

20+ JavaScript Interview Algorithm for Front End Developers.

1

· Reply

View all 3 replies



Denis Savenok 1 week ago

+Михаил Климов, мне на интервью попадалось что-то из этого <http://www.toptal.com/javascript/interview-questions> и один раз <https://github.com/h5bp/Front-end-Developer-Interview-Questions>
· Translate



Михаил Климов 1 week ago

Ну да, опросник quirks. Если еще и интернета не дают то совсем дебилы. Т.е. спрашивают не на решение задач (quicksort это не задача), а на память ..

Представьте вот такой вопрос. Дается ТЗ примерное. Задача рассказать как будешь делать, как будешь избегать technical debt что лучше использовать по

· Translate



محبوب الرشيد 4 months ago - [JavaScript \(Guides/Tutorials\)](#)

Interview questions for a JavaScript Developer

1



kyle hamilton 6 months ago (edited) - Shared publicly

#20 - with permutations you have to ask if the letters should be treated as distinct:

if you treat the same letter as unique, then you will end up with $n!$ elements, which is what your algorithm does.

[1](#) · Reply**Phani Katakam** 11 months ago - Shared publicly

Dear JS DUDE, your site has awesome content.. I wish you would contribute about Closure and OPTIMIZED way of Writing Object Oriented Code in Javascript, so that Learning all other Frameworks - their internal concepts would be clear to a normal developer of JS! Hope you like this idea!

[1](#) · Reply**Denis Savenok** 2 months ago (edited) - Shared publicly

Please fix rand7 (#14) condition: it should be "how could u generate random number 5 to 7 by using that function".

[1](#) · Reply**melvin ho yk** via Google+ 1 year ago - Shared publicly**Khan Md** originally shared this

Rock the Interview. 20+ JavaScript algorithm for Front End Developers (part-1: beginners) [#javascript](#) [#frontenddeveloper](#) [#webdevelopment](#) [#html5](#) [#html5css3](#)

+1 [1](#) · Reply**Andrey Sventitskiy** 8 months ago - Shared publicly

Number can be checked by smaller prime numbers, which are less than a half of the given number. It's useful when you already have the list of prime numbers and looking for greater one.

[1](#) · Reply**sandeep joshi** 9 months ago - Shared publicly

for question 19 we can use `stringObjext.indexOf(subString)` ;

[1](#) · Reply[Show more](#)

9 Comments

Sort by [Top](#)



Add a comment...

**William Harris** · Owner at Self-employed

That optimized algorithm for gcd here

<http://www.thatjsdude.com/interview/js1.html>... is euclids algorithm.Explained on this page <http://introcs.cs.princeton.edu/java/23recursion/>

Like · Reply · Jul 5, 2015 12:42pm

**Benjamin Dobler** · Chief Technology Officer at Parleys.com

I know they probably want you to do it without native functions but in case they want to see if you know them you could also write:

```
function subStringFinder(a,b) {
  return a.indexOf(b);
}
```

for the subStringFinder right?

Like · Reply · Jun 28, 2015 5:45am

**Huan Wang** · University of Pittsburgh

#2, console.log(primeFactors(20)); // [2,2,5]

It's duplicated.

Like · Reply · 2 · Nov 23, 2014 4:03pm

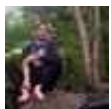
**Chess Roke** · Columbia, Marylandits not necessarily duplicating, $4 * 5 = 20$, but 4 is not a prime number so it can be broken down further to $2 * 2 * 5 = 20$, this result is all prime numbers

Like · Reply · Jul 21, 2015 1:11pm

**Mookie Farina** · Web Developer at LinkedIn

#18 - You should be checking if 'n' > 10', as opposed to 'n > 0' because after 10, there will be no more zeros, and the program has to make dozens of unnecessary loops.

Like · Reply · 1 · Oct 3, 2014 10:10am

**Chris Bitsakis** · Montreal, Quebec

couldn't #6 simply be:

```
function mergeSortedArray(a, b) {
  return a.concat(b).sort(function(a, b){ return a - b});
}
```

at least with the example i get the same result...

Like · Reply · Jul 15, 2014 7:00pm

**Jhankar Mahbub** · Sr. Web Developer at The Nielsen Companyyes. it would be simpler to code but has higher running time complexity. sorting would be at least $n \log n$ and you have concat on top of it

Like · Reply · Jul 24, 2014 2:52pm

Load 4 more comments

 Facebook Comments Plugin

©thatJSDude 2014