【前端面试】console.log(a + a++ * ++ a)...

前言

有些小伙伴可能看到这道题目就已经蒙圈了！甚至可能看不懂这道题目在干什么。其实这是一道比较考察基础的面试题，当你明白原理之后，你可能会感觉这道题也就那么回事，但是如果你没有思绪，那你可能觉得这道题很难。

今天我们就来彻底看看这到底在做什么妖！

1.题目简介

这道题目很简单，面试官给出下面一段代码，问你输出什么？

代码如下：

```
let a = 3
console.log(a + a++ * ++a);
```

小伙伴们别看题目简单就不以为意，通常题目越简单，给我们留下的坑就越多！

2.到底考察什么？

代码如此简单的一道题到底能够考察些我们什么知识呢？

别看题目简单，考察的知识点还挺多的，大概总结如下。

知识点：

- 运算符优先级
- ++运算符
- 表达式

可以看出，如此简单的一行代码就考察了我们三个知识点。

如果我们仔细阅读题目的话，也不难看出它要考察哪些知识点，比如我们分析一下代码：

> 代码：a + a++ * ++a
>
> 1.代码是一个整体表达式，这无疑是考察我们最基本的表达式运算。
>
> 2.代码中含有多个不同运算符，无疑是考察我们对运算符优先级的理解
>
> 3.代码中还有++运算符，这是一个比较特殊的而运算符，需要单独考察我们。

3.运算符优先级复习

想要正确解答这道题目，运算符优先级的知识点我们是逃不过的，这里我们直接给出一张表，让大家复习一遍运算符的优先级规则。

运算符优先级：

优先级	运算符	结合律
1	后缀运算符：[] () · -> ++ --(类型名称){列表}	从左到右
2	一元运算符：++ -- ! ~ + - * & sizeof_Alignof	从右到左
3	类型转换运算符：(类型名称)	从右到左
4	乘除法运算符：* / %	从左到右
5	加减法运算符：+ -	从左到右
6	移位运算符：<< >>	从左到右
7	关系运算符：<<= >>=	从左到右
8	相等运算符：== !=	从左到右
9	位运算符 AND：&	从左到右
10	位运算符 XOR：^	从左到右
11	位运算符 OR：|	从左到右
12	逻辑运算符 AND：&&	从左到右
13	逻辑运算符 OR：||	从左到右
14	条件运算符：?:	从右到左
15	赋值运算符： = += -= *= /= %= &= ^= |= <<= >>=	从右到左
16	逗号运算符：，	从左到右

上面是一张完成的运算符优先级表格，小伙伴们可以简单过一遍，重点查看题目中涉及到的运算符。

注意：上表中的运算符优先级是基于 C 语言排的。

4.++运算符复习

++运算符是一个比较特殊的运算符，所以我们这里单独拿出来复习一遍。之所以特殊，是因为把++放在变量前面和放在变量后面是两种完全不同的效果，比如下面的案例。

++a 代码示例：

```
let a = 1;
console.log(++a); // 2
```

a++代码示例：

```
let a = 1;
console.log(a++); // 1
console.log(a); // 2
```

它们的区别也是显而易见的，从上面代码输出结果可以看出，++a 是先将 a 加 1 之后输出结果，而 a++是输出结果之后再让 a 加 1。

5.题目解答

解答题目之前我们必须知道运算符优先级以及++运算符的知识点，题目中的表达式比较长，为了更好的解题，我们很有必要将表达式划分为几段，这也就是我们常说的**逐个攻破**!

如何划分表达式就需要用到我们的运算符优先级知识了，我们先将题目中涉及到的运算符排个序。

5.1 运算符排序

代码：a + a++ * ++a

涉及的运算符：+ 、 ++ 、 *

运算符优先级排序：++ > * > +

5.2 表达式分段

我们可以看到题目中的运算符总共有 3 级，因此我们可以考虑把这个表达式分为 3 段，而且需要根据运算符的优先级来分。

通常来说，小伙伴们可能会觉得哪个优先级更高，就先从哪里入手，但是我们发现这道题目这样做似乎不太利于我们理解，所以我们不防反过来，先从优先级低的入手。

第一次分段：

> 通过+运算符分段，共分为两端：
>
> 1. a
>
> 2. a++ * ++a

第二次分段：

> 再通过*运算符分段，此时将代码分为了三段
>
> 1. a
>
> 2. a++
>
> 3.++a

我们发现经过两次分段题目中的表达式就变为了 3 段，也达到了我们的效果了，接下来就看如何计算了。

5.3 分段计算

表达式被分为了三段，那么我们分别计算 3 段表达式即可。

题目回顾：

```
let a = 3
console.log(a + a++ * ++a);
```

先计算 a++：

```
a++ = 3
```

之所这儿输出 3，是因为++表达式放在变量后面是先输出结果，再计算 a+1，所以 a++整体为 3.

再计算++a：

```
++a = 5
```

这里是最容易出错的地方，因为我们前面先执行了 a++，所以此时的 a 应该是 4，然后执行 ++a 的时候，这一个整体就变为了 5。

分块计算后的表达式：

我们可以将每一块的计算结果写成数字代入表达式看一下

```
3 + 3 * 5 = 18
```

上面那个表达式相信大家一眼就能看出来了吧，有些小伙伴可能对第一个 a=3 有一点点疑问：为什么++a 的时候 a 变为了 4，而第一个 a 还是为 3 呢?

因为我们再 JS 解释执行代码的时候其实都是从左往右开始的，当碰到第一个 a 时，就是 a 变量，直接赋为 3 即可，而到后面的++a 和 a++都是表达式，都是需要计算得到结果的。

5.4 最终结果

通过上面的分析和计算，我们很容易得出答案：18。

浏览器测试：

```
> let a = 3
< undefined
> console.log(a + a++ * ++a)
  18
< undefined
> console.log(a)
  5
< undefined
>
```

总结

有些小伙伴觉得这道题非常的变态，实际项目中根本不可能这样写。事实却是如此，但是为什么各大厂还是喜欢考这种面试题呢？大概是为了最快的考察你的基础知识和应变能力吧，毕竟每天面试的人那么多。