1. 电子邮件的正则表达式:

答: \w+ 1个以上的合法字符

@ 必须有一个@

\w+ 1个以上的合法字符

\. 必须有一个. 但是.是通配符，要转义为\.

\w+ .后至少有一个后缀是一个以上的合法字符

\.cn 最后的.cn可有可无

\w+@\w+\.\w+(\.cn)?

2. 防守微信的正则表达式:

答: (微|w(ei)?) 汉字微或拼音wei，但是ei可有可无

\s\* 可能有空格，也可能没有空格

(信|x(in)?) 汉字信或拼音xin，但是in可有可无

(微|w(ei)?)\s\*(信|x(in)?)

3. 学了很多，记不住，好像什么也不会做？

答：没有做个人项目。

4. 数据库不会，nodejs不会，怎么做项目？

答：一点儿关系没有！

先做"原型"：就是和成品项目一模一样的静态页面

静态界面上的数据去原网站copy，或者将来可以找一些开放接口。

5. 为什么学nodejs?

答: 为了了解后端，为了自保！当功能出现问题时，可以判断出是那一端的责任。

6. 第一阶段学的不好，第二阶段boot记不住？

答: 第一阶段，只有JSBASIC 那9天是重点

第二阶段，只有CSS基础，CSS高级，CSS3动画，媒体查询。

7. 做项目可以仿写别人模板的样式吗？

答：现在大家正处在照猫画虎的阶段，就像达芬奇画蛋一样。不可能学完立刻原创。所以，现阶段，做静态页面，本就应该以模仿和参考别人优秀的模板为主。时间有限，精力有限，最忌讳事事亲力亲为，从头写到尾。但是，仿写别人的网站，要吸取别人的优秀做法和技巧。效果上，要尽量达到原来的要求。最好能够有超过原网站的效果作为亮点。所以，应该多参考一些优秀的模板。

8. 多个页面中相同的部分，比如页头，页尾怎么做？

答：多个页面中共用的部分，单独做成一个.html文件，比如,header.html，单独调试样式和内容。

但是！千万不要用iframe引入！

在我这个阶段，会教大家用正确的ajax和vue两种方式引入公共部分。

9. 如果git命令行有问题，怎么办？

答：下载有界面的github desktop

10. match返回的数组中，房间号必须叫index吗？可以自己改名吗？

答：不可以。必须用index。

虽然关联数组的下标名，可以自定义，但是，那是在你自己创建关联数组的情况下。

现在这个数组是人家match函数返回的，不是你自己创建的。所以，人家返回什么样，就只能按照人家的要求用。

11. 然哥教我们用的iframe，为什么你不让用？

答：那是因为你们当时学的少，然哥也没办法。

今后做项目就是不用iframe的。然哥也不用。

每个阶段都有每个阶段的局限。

12. 为什么输入0就可以拿到找到的敏感词呢？

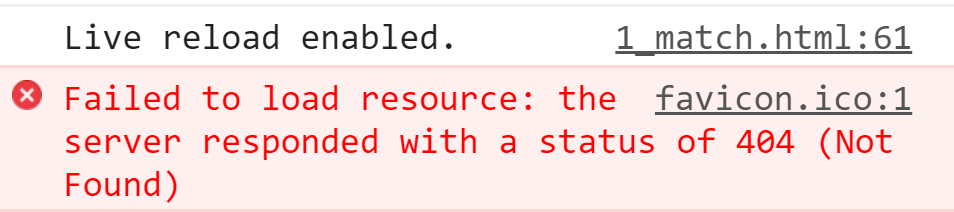
答：match在找到敏感词后，会自动在内部创建一个数组，然后将敏感词放入数组的0位置。然后再把数组交给我们。所以，人家放在哪个位置，我们就只能从哪个位置取。

就好像中午点盒饭，人家把肉菜放在第一个饭盒里，把素菜放在第二个饭盒里。那我们想吃肉菜，只能从第一个饭盒里取。

13. match返回的arr数组

长度应该是6个房间呀？为什么length为3？

答：因为length默认只能统计数字下标的房间个数。字符串下标的房间不参与统计。记住即可。没有为什么。

14. ？

答：这个右侧显示favicon.ico:1的错误不用管。是live servre运行时报的一个无关紧要的错误。

重新刷新页面就没有了。

15. vs code下弹出错误: open a folder or workspace

答：vs code要用Live server运行，必须先打开一个文件夹，文件夹中再包含网页文件。不能直接打开网页文件运行。

16. arr!=0 也可以？

答: !=是关系运算，关系运算中，左右默认隐式转为数字

如果arr返回null，也会被Number()隐式转为数字

Number(null)=0，所以可以和0作比较。

如果arr返回的是一个有内容个数组，也会被转为数字：Number([1,2,3])=NaN，NaN!=0 返回true条件成立。

但是，按照语义，必须写成arr!=null。

写程序，语义优先。还有，不要瞎改程序！

17. 返回的敏感词一定放在0位置吗？

答：一定的！不会例外！因为浏览器自带的原生函数，都是ECMAScript标准固定好的，必须这么实现。不会有例外。

还有，写程序要有基本的人和人之间的信任。

18. ajax四步或五步不会，怎么办？

答：根本没关系:

DOM: 我给你封装好了ajax()

jQuery: 人家也封装好了$.ajax()

VUE: 人家也封装好了axois.get()

19. == === != !==

答: == !=是带隐式转换的等于比较

交给他们的两个值，都要先被隐式转换一下，再比较。可能和你的本意就不一致了。

=== !==是不带隐式转换的严格等于比较

绝对不会对给的两个值擅自加工。而是保持原样进行比较。多数情况下，都要用===和!==代替==和!=

但是，如果必须在不同类型数据之间作比较，需要先强制转换为一致类型，再用===和!==比较。

如果确实对js的隐式转换了如指掌，可以使用==和!=

20.\*\*\*\*\* Cannot read property 'xxx' of null或undefined

答: 凡是看到这个句式的错误，不是属性名写错了，也可能不是这句话写错了。很可能是xxx属性.前的变量，该拿到的值，没拿到！

解决: 往前找！找这个变量在哪里赋的值

21. replace中的function，换成箭头函数，也能用

答: 暂时禁止使用箭头函数。因为箭头函数有坑，不是所有情况都能换。所以，必须等我讲完再用！

22. replace(/xxx/,function(){...})中的匿名函数是因为那个()才被调用的？

答：回调函数，是一种我们只负责定义，不负责调用，而交给其他函数内部去自动调用的函数。所以，我们不可能看到回调函数被自动调用时的()。

这里replace()后的()是开始调用replace函数，不是调用内部匿名函数的()

function后的()，是定义函数的语法要求，不是函数调用。

所以，我们现在用肉眼看到的()，都和内部匿名函数自动调用无关。

23. console.log(ltrim(str)) ltrim是回调函数?

答: ltrim绝不是回调函数，而是主动调用过了一次！ltrim(str)已经构成了标准的函数调用语法。

为什么ltrim(str)直接放在console.log()里？因为我想将ltrim()执行的结果，直接交给console.log()去打印

console.log(<- 结果 <- ltrim(str))

24. ltrim()不加str，也能执行?

答: 每个函数都是一个独立的功能体，函数内部的变量应该尽量避免直接使用外部的变量名。

这里ltrim()不加str参数也可使用，是因为巧合。函数内的变量和全局变量str同名了。所以，就算你不给str，函数执行时会自动去全局找str变量用。

但是这样是不对的。因为外部变量名一旦变化，函数也连带着不能使用了。

所以，将来函数需要什么数据，必须通过形参从外部获得。而就不能直接使用外部的变量。

25. var reg=/^\d{6}$/; reg.test(str)验证的是数字吗？

答: 验证的不是数字。而是字符串中的数字字符。

比如: 程序中123和"123"

虽然在页面上输入的是123456，但是所有从页面传入的数据，都自动加上了""变成了字符串

年龄:

|  |
| --- |
| 26 |

↓

程序中

var age=input.value;

age:"26"

26. 将/^\d{6}$/换成new RegExp("^\d{6}$")为什么不对？

答： 正则的\d和js中字符串本身的\转义字符冲突了！解析时，js先将\d看做自己的转义字符，但是js中又没有\d这个转义字符，所以js擅自将\d前的\去掉了！

解决: 如果用new RegExp()其中的\d都要转为\\d，第一个\是在告诉js，后边的东西不要动，不需要你解析！

但是\u4e00和\u9fa5不用保护。因为js认识\u

27. if(arr!=null)  
 ...arr.index ... arr[0] ...

就正确

如果改为

...arr.index ... arr[0] ...

}while(arr!=null); 报错

答: 如果arr有可能是null的话应该先判断不是null，才能使用。如果将条件放在后边的while中，则顺序变为先使用，后判断，就失去了对arr=null的防护。

28. arr["index"]可以反问"index"位置的元素，也可以用.访问arr.index也可以。arr["length"]可反问数组长度，arr.length也可访问数组长度。为什么？

答: js中一切对象底层，其实都是关联数组！

任何对象和数组的成员，都可用["下标"]访问！

只要是非数字的下标，都可用.简写！

任何对象和数组都可在任何时候随意添加新成员

访问任何对象和数组的不存在位置，都不会报错，而是返回undefined！

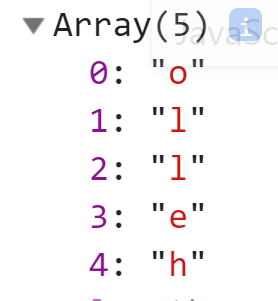
所以，其实arr["index"]可简写为arr.index。

其实arr.length也可改为arr["length"]

但是，数字下标不能简写！arr.0和小数点冲突了！

只能用[]方式访问！

29. 为什么arr.reverse()调用前输出的arr，却是已经翻转过的！



答: console.log(对象/数组)其实不是给对象拍照！而是输出对象的地址！

当程序执行完，我们去点地址旁边的时，才临时去内存中找到这个对象，显示对象的内容。

因为此时程序已经执行完了，且内存中只有一个arr数组。所以，无论何时点开三角，看到的对象内容都是一样的，且都是最终的结果！

解决: 如果非要看到数组中间瞬时的状态: 2个办法

1. 刷新页面，会在控制台，显示数组在中途执行时的照片。但是也不要展开。

2. 可以将数组或对象转为字符串，再输出

String(数组) vs toString(数组)

其实两个是极其相似的。