1. 正则表达式是一个对象，可以进行强大的模式匹配和文本检索以及替换功能。
2. 所有的字母和数字都是按照字面含义进行匹配的。前面加上“\”前缀用以进行转义。
3. 很多标点符号都有特殊意义，如果想要对这些字符的直接量进行匹配，必须使用“\”，有时候不需要记住那些标点符号需要转义，可以都加上反斜线。
4. […] 方括号内的任意一个字符
5. [^…] 不在方括号内的任意一个字符
6. . 除了换行符和其他Unicode终止符之外的任意字符
7. \w ASCII字符，同[0-9A-Za-z]
8. \W 非ASCII字符，同[^0-9A-Za-z]
9. \s 任何Unicode空白符
10. \S 任何非Unicode空白符
11. \d ASCII数字，同[0-9]
12. \D 任意的非ASCII数字字符，同[^0-9]
13. 重复
14. {m,n} 不能少于m个，不能多于n个
15. {m,} 不能少于m个
16. {m} 就是m个
17. ? 0或一个，相当于{0，1}
18. + 1个或多余1个，相当于{1，}
19. \* 0个或多余0个，相当于{0，}
20. 这些重复都是贪婪的匹配，尽可能多的匹配，在上述这些描述符后面加上“？”可以让他们变成非贪婪的匹配。
21. 选择，“|”，匹配的是左边的子表达式或右边的子表达式。
22. （….）  
    第一个作用：将单独的项组合成子表达式，以便处理“|”、“\*”等等。  
    第二个作用：在完整的模式中定义子模式。当一个正则表达式成功地与目标字符串相匹配时，可以从目标字符串中抽出和圆括号中的子模式相匹配的部分。如  
     var quote=/“([^”]\*)”/g; text.replace(queto,’$1’).  
    第三个作用：允许在统一正则表达式的后部引用前面的子表达式。对正则表达式的引用，不是对子表达式模式的引用，而是与那个模式相匹配文本的引用。如：  
     /[‘”][^’”]\*[‘”]/ 这样会导致前后两个引号的类型不相同  
     /（[‘”]）[^’”]\*\1/ \1表示前面个括号匹配的文本  
     /（[‘”]）[^\1]\*\1/
23. （？：）括号中间加入“？：“，表示这个括号只组合，不记忆与该组相匹配的字符。
24. 指定匹配位置  
    ^ 表示匹配的字符串的开头  
    $ 表示匹配的字符串的末尾  
    \b 表示单词的边界  
    \B 表示单词的非边界  
    (?=p) 正向先行断言，要求接下来的字符与p匹配，但不包括匹配p的那些字符  
    （？！p） 反向先行断言，要求接下来的字符不与p匹配
25. 修饰符  
    i：不区分大小写  
    g：全局搜索  
    m：多行模式。因为有的字符串很长，在这种模式下，^和$不仅匹配字符串的开头和结尾，还匹配每行的开头和结尾。
26. 字符串的正则表达式方法
27. String.prototype.search() 返回数字  
    search传入的参数可以是一个字符串，也可以是一个正则表达式。如果是字符串，会将字符串转化为正则表达式，并忽略正则表达式的g。返回第一个与之匹配的字符串的起始位置，如果没查找到，那么返回-1。
28. String.prototype.replace() 返回一个新的字符串  
    replace传入的参数是字符串也可以，但不会转换为正则表达式，此时，会替换目标字符串中的所有被匹配的。  
    replace传入的是正则表达式时，如果不是g，只替换第一个被匹配的。如果是g，则全局替换。
29. String.prototype.match() 返回的内容由传入的正则表达式决定  
    如果该正则表达式没有g，没有匹配的内容时，返回null。如果有，返回一个数组。第一个元素是匹配的整个模式的字符串，其他元素是括号内的子模式所匹配的字符转。还有一个input属性是对该字符串的引用，index指定了匹配文本在字符串中的位置。  
    如果改正则表达式有g，返回的是null或数组。如果是数组，那么不包括input和index，只是所有匹配的内容。也就是，全局搜索，match不会提供关于圆括号子表达式的信息。
30. String.prototype.split() 返回数组  
    第一个参数可以是字符串，也可以是正则表达式。将字符串切割为数组，匹配正则表达式和字符串的不会被放在数组中。  
    第二个参数指定返回数组的最大长度，如果分割后的数组大于这个，会被舍弃后面的。  
    如：”1:2:3:4:5”.split(“:”); 返回[“1”,”2”,”3”,”4”,”5”];  
    如果分隔符匹配到字符串的首部或尾部，那么首部和尾部返回一个““空字符。  
    如: “a|b|c|”.split(“|”); 返回[“a”,”b”,”c”,””]
31. 正则表达式对象包括5个属性，source是一个只读字符串，ignoreCase、mulitiline、global布尔值，lastIndex是一个可读可写属性，如果是全局搜索，这个属性会被存储在整个字符串下一次检索的开始位置。  
    exec（）  
    不管这个正则表达式有没有g，最终的结果都是字符串的match（）全局正则表达式的方法。  
    test（）  
    检测一个字符串中是否有匹配的，悠久返回true，没有就返回false。  
    String的方法只是简单地将lastIndex重置为0；  
      
      
      
      
      
      
      
      
      
      
    字符串的其他方法：
32. charAt()  
    返回在索引处的字符。
33. charCodeAt（）  
    返回在索引处的字符的Unicode编码。是一个16为数字。
34. concat（）
35. 静态方法 String.fromCharCode（）  
    通过指定的Unicode编码数字来创建字符串。
36. indexOf（）
37. lastIndexOf（）
38. slice（） 同数组的slice方法。
39. trim（） 去除开头和结尾处的空白字符，返回一个新的字符串。