1. js的时间 Date()
2. Document.getElementById(“demo”).innerHTML=”Hello JavaScript”;
3. Document.getElementById(“demo”).style.fontSize=”25px”;
4. Document.getElementById(“demo”).style.display=”none”;
5. 外部js的优势
   1. 分离了HTML和代码
   2. 使HTML和js更易于阅读和维护
   3. 已缓存的js文件可加速页面加载
6. Js显示方案
   1. 使用window.alert()写入警告框
   2. 使用document.write()写入HTML输出（仅用于测试的时候）
   3. 使用innerHTML写入HTML元素（更改HTML元素的innerHTML属性是在HTML中显示数据的常用方法）
   4. 使用console.log()写入浏览器控制台（用于调试的时候）
7. js语句由值、运算符、表达式、关键词和注释构成
   1. 在运算符旁边添加空格是个好习惯
   2. 如果运算符太长 对其进行折行的最佳位置是某个运算符
   3. Break 终止switch或循环
   4. Continue结束本次循环并继续循环
   5. Debugger停止执行js，并调用调试函数（如果可用）
   6. Do...while 执行语句块 并在条件为真时重复代码块
   7. For 标记需被执行的语句块 只要条件为真
   8. Function 声明函数
   9. If...else标记需被执行的语句块，根据某个条件
   10. Return 退出函数
   11. Switch 标记需被执行的语句块，根据不同的情况
   12. Try...catch对语句块实现错误处理
   13. Var 声明变量
8. js对大小写敏感
   1. 驼峰命名法
      1. 开头大写 之后的每个单词首字母大写
      2. 开头小写 之后的每个单词首字母大写
9. 注释
   1. 多行注释：以/\*开头 以\*/结尾
   2. 单行注释：以//开头 没有结尾
10. Js变量
    1. 是存储数据值的容器
11. 数据类型
    1. 值类型：
       1. 字符串string、数字number、布尔boolean、空null、未定义undefined、symbol
    2. 引用类型：
       1. 对象object、数组array、函数function、正则regexp、日期date
    3. js拥有动态类型，意味着相同的变量可用作不同的类型
    4. 变量的数据类型可以使用typeof操作符来查看typeof “json”-->返回string
    5. Undefined表示当前变量没有定义
    6. Null表示当前变量虽定义了但并未指向任何内存中的对象
12. 对象
13. 函数
14. 作用域：可访问变量，对象，函数的区域
    1. 局部变量在使用完后销毁，全局变量在页面关闭后销毁
15. 变量提升
    1. Js中，函数及变量的声明都将被提升到函数的最顶部，也就是可以先使用，在声明（因为声明的语句都会被提到函数的最顶部执行）
16. 严格模式（use strict）
    1. 通过在脚本或函数的头部添加use strict；来生明
    2. 消除Javascript语法的一些不合理、不严谨之处，减少一些怪异行为;
    3. 消除代码运行的一些不安全之处，保证代码运行的安全；
    4. 提高编译器效率，增加运行速度；
    5. 为未来新版本的Javascript做好铺垫。
    6. "严格模式"体现了Javascript更合理、更安全、更严谨的发展方向，包括IE 10在内的主流浏览器，都已经支持它，许多大项目已经开始全面拥抱它。
    7. 另一方面，同样的代码，在"严格模式"中，可能会有不一样的运行结果；一些在"正常模式"下可以运行的语句，在"严格模式"下将不能运行。掌握这些内容，有助于更细致深入地理解Javascript，让你变成一个更好的程序员。
17. 使用误区（一些需要注意的地方）
18. 表单、表单验证
19. This
    1. 方法中：表示该方法所属的对象
    2. 单独使用：表示全局对象
    3. 函数中，表示全局对象
    4. 事件中，表示接收事件的元素（html）
20. let和const
    1. Let声明的变量，只在let命令所在的代码块{}内有效，在{}之外不能访问
    2. Const声明的变量，不能通过再赋值修改。使用const定义的对象或数组是可修改的。
21. JSON：轻量级的数据交换格式
    1. Json对象：{"name":"Runoob", "url":"www.runoob.com"}
    2. Json数组：
       1. "sites":[ {"name":"Runoob", "url":"www.runoob.com"}, {"name":"Google", "url":"www.google.com"}, {"name":"Taobao", "url":"www.taobao.com"} ]
    3. JSON.parse()将一个json字符串转换为js对象
    4. JSON.stringify()将js值转换为JSON字符串
22. 在a标签中的href属性等于Javascript:void(0)和#的区别
    1. 前者是一个死链接，点击之后什么也不会发生
    2. 在页面很长的时候会使用#来定位页面的具体位置，格式为：#+id
23. 异步编程
    1. Ajax async
24. 浏览器对象模型
    1. Window。Open打开新窗口
    2. Window。Close关闭当前窗口
    3. Window。Moveto移动当前窗口
    4. Window。Resizeto调整当前窗口的尺寸
    5. Js window screen屏幕
    6. Js window location位置
    7. Js window history
       1. History。Back与在浏览器点击后退按钮相同
       2. History。Forward与在浏览器中点击向前按钮相同
    8. Window navigator对象包含有关访问者浏览器的信息
    9. Js弹窗
       1. Alert（警告框）
       2. Confirm（确认框）
       3. Prompt（提示框）
    10. 计时事件
        1. Setinterval（）间隔指定的毫秒数不停地执行指定的代码
        2. Settimeout（）在指定的毫秒数后执行指定代码
        3. Clearinterval（）用于停止setinterval（）方法执行的函数代码
        4. Cleartimeout（）用于停止执行settimeout方法的函数代码
    11. Js cookie
        1. Document。Cookie=“username=John Doe”--写
        2. document.cookie="username=John Doe; expires=Thu, 18 Dec 2043 12:00:00 GMT";
        3. document.cookie="username=John Doe; expires=Thu, 18 Dec 2043 12:00:00 GMT; path=/";
        4. Var x = document。Cookie--读
        5. Document。Cookie = “username=John Smith”--改
        6. Document。Cookie = “username=；expires=”--删除（设置过去的时间）
25. 1
26. 1
27. 1
28. 1
29. 1
30. 1
31. 1
32. 1
33. 1
34. 1
35. 1
36. 1
37. 1
38. 1
39. 1