1、什么是预编译？

JS代码预编译（解析JS代码、先检查是不是JS代码，是的话就开始检查语法是否有错误，如果没有错误则开始变量提升，然后赋值，赋值之后可操作，变成活动对象，如果错误了，直接跳过，执行下一块代码

2、为什么全局变量写多了，会污染全局环境？

全局变量写多后，会栈溢出

3、为什么eval不安全？

因为他将JS代码执行两遍，第一遍先将字符串转换成浏览器可以访问的对象，第二遍在进行编译

4、为什么typeof判断的null返回值为objcet？

因为电脑的cpu只能计算二进制的数据，例如：0011001。所有的引用数据类型翻译成二进制，只要是二进制的引用数据类型的数据，前三位都是000开头；null是开头为000的二进制数据，typeof的作用是通过进制来判断是基本数据类型还是引用数据类型，typeof只要碰到 000 开头的二进制数据，则都返回object，function除外；

5、原型链查找规则/机制：（浏览器调用原型链的查找规则/机制）

先从自身查找，找到就直接返回，结束原型链调用，没有找到则一直向上级查找，找到返回，找不到返回undefined，原型链的顶端：Object.prototype === null

6、url在地址栏经历了哪些过程？

1、输入url；2、先检查本地是否有缓存，有直接渲染页面；3、DNS服务器查询 查询IP地址；4、发送请求，建立连接（和服务器建立连接，三次握手）；5、TCP传输http报文到前端（报文：数据信息）；6、下载，然后调用浏览器的一切用到的进程进行渲染、解析、画图；7、断开连接（四次挥手）；8、呈现UI

7、三次握手是哪三次？

三次握手（和服务器建立连接）：由客户端发起的

1、客户端发起连接请求，携带SYN标识

2、服务端接收到了客户端的SYN标识，服务端重新给客户端返回当前标识，并且多携带一个确认码（SYN+ACK）

3、客户端收到SYN+ACK码，确认无误了，再次给服务器发送ACK确认码

8、四次挥手是哪四次？

四次挥手（和服务器断开连接）： 客户端和服务端均可发起断开连接的请求

 谁发起结束请求，谁最后决定彻底断开

1、当数据传输完毕后，客户端发起断开连接的请求，携带一个FIN结束码给服务端

2、服务端接收到FIN，则询问是否真的断开，发送一个ACK确认码

3、服务端再次发送一个最后的FIN结束码给客户端

4、客户端确定结束，返回ACK确认码 （断开连接了）

9、前后端交互过程

1、客户端发起连接请求，携带SYN标识

2、服务端接收到了客户端的SYN标识，服务端重新给客户端返回当前标识，

3、客户端收到SYN + ACK码，确认无误了，再次给服务端发送ACK确认码

4、传输数据

5、当数据传输完毕后，客户端发起断开连接的请求，携带一个FIN结束码给服务端

6、服务端接收到FIN，则询问是否真的断开，发送一个ACK确认码

7、服务端再次发送一个最后的FIN结束码给客户端

8、客户端确定结束，返回ACK确认码 （断开连接了）

10、get和post请求的区别 ？ （面试题）

get：1、请求数据携带在url上；2、不安全，因为数据暴露在地址栏中；3、传输数据大小受WEB服务器限制（2MB）；4、一般用于获取数据；5、get请求只能进行url编码；6、url地址栏会记录get请求的参数

post：1、请求数据在请求体内传输；2、相对安全，内容不是暴露出来的；3、大小基本不受限制，因为请求体可以传输很多数据；4、一般用于表单提交 例如：注册、登录、修改、添加用户；5、post支持多种编码格式；6、不会记录请求参数

11、图片为什么要转换成base64格式？有什么好处？

base64图片，随着html文档的下载一起被下载到本地，不用发送http请求，减少服务器压力，提高网页性能 【一般小于8kb的转成base64图片，大于8kb的不用转】

12、事件循环：

1）、所有的JS都在栈的主线程执行

2）、执行栈优先执行同步任务，从上到下

3）、等主线程执行完成同步代码了，主线程空闲了，才去检查任务队列中是否有异步任务，有则拿出来到执行栈（主线程）执行，当前任务执行完毕，继续执行下一个任务，直到任务队列中没有任何任务，停止执行

4）、优先检查上一个宏任务中是否有遗留下来的微任务，有就执行，没有就执行当前宏对象，然后在执行当前宏任务内的微任务

5）、一直循环3、4步骤，形成一个事件循环，直到任务队列中没有任务

6）、async和await后面返回的是promise对象或者一个值，当事件循环执行的过程中，遇到await的时候，立马返回结果，然后之后的同步代码都扔到微任务队列去，宏任务扔到下一个宏任务中去，如果async中没有遇到await的时候，则当同步代码执行