九十一、进程的组成？进程和线程的区别？

程序段、数据段、PCB(进程的控制块)(操作系统管理的数据存放在PCB)

> 进程之间相互隔离。同进程的线程之间的堆、方法区共享。
>
> 进程是资源分配基本单位。
> 线程是程序执行流的最小单位，即调度的基本单位。
>
> 过去的进程间并发，需要切换进程的运行环境，系统开销很大。
> 同进程内线程间并发，不需要考虑切换进程的运行环境，系统开销较小。

九十二、原语是什么？

原语是一系列不允许被中断的代码，确保要同时完成，是通过内核态在原语前后加上“关中断指令”和“开中断指令”实现。

九十三、处理机调度有哪些方式？

高级调度（作业调度）：外存->内存

按照某种调度规则，从外存中后备队列选择作业将其调入内存，并未其创建进程，发生频率最低

中级调度（内存调度）：外存->内存

按照某种规则，从外存中的挂起队列选择合适的进程将其调入内存(因为内存的空分复用技术)，发生频率中等

低级调度（进程调度）：内存->CPU

按照某种规则，从内存中就绪队列选择一个进程为其分配处理机，发生频率高

九十四、生产者、消费者模型？

王道P23

九十五、静态集合类引起内存泄漏？

像HashMap、Vector等的使用最容易出现内存泄露，这些静态变量的生命周期和应用程序一致，他们所引用的所有的对象Object也不能被释放，因为他们也将一直被Vector等引用着。

例如

```
Static Vector v = new Vector(10);
for (int i = 1; i<100; i++)
{
    Object o = new Object();
    v.add(o);
    o = null;
}
```

在这个例子中，循环申请Object 对象，并将所申请的对象放入一个Vector 中，如果仅仅释放引用本身（o=null），那么Vector 仍然引用该对象，所以这个对象对GC 来说是不可回收的。
因此，如果对象加入到Vector 后，还必须从Vector 中删除，最简单的方法就是将Vector对象设置为null。

九十六、C++的指针函数，函数指针？

int* fun(int x,int y); 指针函数：指针函数本质是一个函数，其返回值为指针。

int (*fun)(int x,int y); 函数指针：函数指针本质是一个指针，其指向一个函数。

九十七、状态码301和302的区别

301 Moved Permanently即永久重定向，对应着302 Found，即临时重定向。

比如你的网站从 HTTP 升级到了 HTTPS 了，以前的站点再也不用了，应当返回301，这个时候浏览器默认会做缓存优化，在第二次访问的时候自动访问重定向的那个地址。

而如果只是暂时不可用，那么直接返回302即可，和301不同的是，浏览器并不会做缓存优化。

九十八、HTTP/1.1 比 HTTP/1.0 有哪些提高？仍有哪些不足？

1、长连接。

2、支持管道网络传输：只要一个请求发出了，不需要等待返回，就可以发第二个请求，减少整体的响应时间。

HTTP/1.1不足：

1、头部未压缩：请求头、响应头未经压缩就发送，首部信息越长发送越慢。
2、队头阻塞：服务器按请求的顺序响应，若服务器响应慢，会导致后面一直阻塞。
3、没有优先级控制。
4、请求只能客户端开始，服务器只能被动响应。

九十九、HTTP/2 比HTTP/1.1有什么优化？

1、头部压缩:如果你同时发出多个请求，他们的头是一样或者是相似的，协议会帮你消除重复的部分。

使用HPACK算法：在客户端和服务器上同时维护一张头部信息表，只需发送索引号即可。

2、二进制报文：HTTP/2使用的是二进制格式的报文，计算机收到二进制报文，无需再转为二进制，直接解析即可。增加传输效率。

3、优先级：客户端可以指定数据流的优先级。

4、多路复用：HTTP/2可在一个连接中并发多个请求和响应，不用按照顺序一一对应。解决了队头阻塞问题，降低了延迟，提高了连接利用率。

5、服务器推送：服务器可以主动向客户端发送消息。例如：浏览器请求HTML，服务器可以提前把用到的js，css等静态资源发送给客户端，减少延时的等待。

一百、HTTP/3比HTTP/2做了哪些优化？

HTTP/2:

多个HTTP请求在复用一个TCP连接，下层的TCP协议并不知道有多少个HTTP请求的。

所以一旦丢包，就会TCP重传。所有HTTP请求都要等待这个包被重传。

HTTP/3：

把TCP协议改成了UDP。由于UDP是不可靠传输，但是基于UDP的QUIC协议可以实现类似TCP的可靠性传输。

QUIC有一套自己的机制保证传输的可靠性，某个流丢包时，只会阻塞这个流，其他流不会受到影响、

但是QUIC是新协议，很多设备只会把QUIC当做UDP。所以普及缓慢。