

c/c++学习：

https://github.com/Ewenwan/ShiYanLou

面试roadmap

面试知识点：

https://github.com/CyC2018/Backend-Interview-Guide/blob/master/doc/%E4%B8%80%E6%96%87%E5%B8%AE%E4%BD%A0%E7%90%86%E6%B8%85%E9%9D%A2%E8%AF%95%E7%9F%A5%E8%AF%86%E7%82%B9.md

剑指offer：

https://github.com/CyC2018/CS-Notes/blob/master/docs/notes/%E5%89%91%E6%8C%87%20offer%20%E9%A2%98%E8%A7%A3.md

计算机操作系统：

https://github.com/CyC2018/CS-Notes/blob/master/docs/notes/%E8%AE%A1%E7%AE%97%E6%9C%BA%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F.md

**汇总：**

**https://github.com/CyC2018/CS-Notes**

数据库：

数据库系统原理🡪SQL🡪Leetcode-Database题解🡪SQL Server书籍🡪Redis

算法：

时间/空间复杂度🡪剑指offer题解(68题)🡪Leetcode题解

操作系统：

计算机操作系统🡪Linux

网络：

计算机网络🡪HTTP🡪Socket

面向对象：

设计模式🡪面向对象思想

面试经验：

https://github.com/FangWW/Document

时间复杂度：反映了程序执行时间随输入规模增长而增长的量级。

时间消耗决定因素：

1算法采用的策略、方法

2编译产生的代码质量

3问题的输入规模

4机器执行指令的速度

从算法中选取一种对于所研究的问题(或算法类型)来说是基本操作的原操作，以该操作的重复执行的次数作为算法的时间量度。

时间频度：一个算法中语句执行的次数称为时间频度或语句频度。记为T(n)

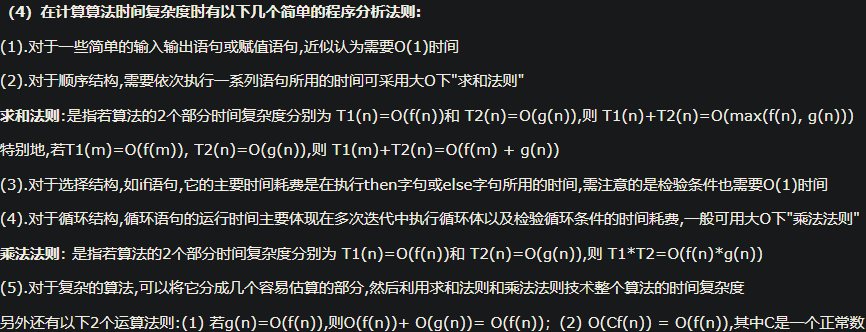
时间复杂度：n称为问题的规模，一般情况下，算法中基本操作重复执行的次数是问题规模n的某个函数，用T(n)表示，若有某个辅助函数f(n),使得当n趋近于无穷大时，T(n)/f(n)的极限值为不等于零的常数，则称f(n)是T(n)的同数量级函数。记作T(n)=Ｏ(f(n)),称Ｏ(f(n)) 为算法的渐进时间复杂度，简称时间复杂度。

计算时间复杂度：

1找出算法中的基本语句。算法中执行次数最多的那条语句，通常是最内层循环的循环体。

2计算基本语句的执行次数的数量级。保证基本语句执行次数的函数中最高次幂正确即可，可以忽略低次幂及最高次幂系数.

3用大O表示算法的时间性能。



空间复杂度：该算法所耗费的存储空间。包括存储算法本身所占用的存储空间，算法的输入输出数据所占用的存储空间和算法在运行过程中临时占用的存储空间。

算法：

1设置到数组，关于寻值，判断。一般在进行查找的同时，需要对数组进行排序，降

低算法的时间复杂度。

2递归和动态规划将一个问题划分成多个子问题求解，但是动态规划会把子问题的解

缓存起来，从而避免重复求解子问题(如：斐波那契数列)

3首先要想出一个完整可行的算法，如果条件允许的话，再考虑有无更好的算法。

4一眼就能知道一种解决算法时，往往是需要利用其他算法降低时间、空间复杂度。

5递归和迭代，递归代码量少，但是难理解；迭代代码多，需要用到额外的指针变量，

但是容易理解；

02-18 算法10.4 计算机系统-进程状态的切换

02-20 算法 16

02-21 算法 23

02-22 算法 25

02-25 算法 33

涉及递归的题：24、25、26

适合直接看面试总结的：

三星：

--算法

--剑指Offer题解()：面试常见题

--Leetcode题解()：面试常见题

--算法(7晚上)：一些常见的算法，比如排序，树的查找

--操作系统：

--Linux(12小时)

--计算机操作系统(13小时)

--网络

--计算机网络(12小时)

--HTTP(8小时)

--Socket(4小时)

--系统设计

--系统设计基础(1小时)

--分布式(12小时)

--集群(12小时)

--缓存(1晚上)

--消息队列(1小时)

--攻击技术(1小时，可不看)

**系统设计基础**

一：性能

性能优化

集群：使用负载均衡将请求转发到集群中。

缓存：

缓存数据通常位于内存中，读操作快；

缓存数据可以位于靠近用户的地理位置上；

可以将计算结果进行缓存，从而避免重复计算。

异步：

二：伸缩性

指不断向集群中添加服务器来缓解不断上升的用户并发访问压力和不断增长的数据存储需求。

三：扩展性

指添加新功能时对现有系统的其他应用无影响，这就要求不用应用具备低耦合的特点。

实现可扩展的方式：

1使用消息队列进行解耦，应用之间通过消息传递进行通信；

2使用分布式服务将业务和可复用的服务分离开来，业务使用分布式服务框架调用可复用的服务。新增的产品可以通过调用可复用的服务来实现业务逻辑，对其他产品没有影响。

四：可用性(和伸缩性的区别)

冗余

保证高可用的主要手段是使用冗余，当某个服务器故障时就请求其他服务器。冗余通过主从复制实现。

监控

服务降级

系统为了应对大量的请求，主动关闭部分功能，从而保证核心功能可用。

五：安全性

**消息队列**

一：消息模型

二：使用场景

三：可靠性

幂等性：多次执行的效果与一次执行的影响相同。

现阶段可以去学习的：Redis、深入学习C#、net core(eshoponcontainer可以学习了)