# 一轮

1. 二叉树镜像

2. 在输入的数组中找出第k大的素数，输入数组可能包含不是素数的数字

思路1：除了2和3，其他所有的素数只能是6N-1或6N+1。

3.求无向无环题的直径

4.求二叉树的左视图看到的节点

5.二叉树求和，找出数字只和等于21的数组下边，要求复杂度要小于o(logn)

6.随机给两个单链表的数，如1->2->3->4，2->3->4，求和的代码实现

7.一个二叉排序树，自己实现其一个Iterator

8.一个顺序递增的数组，内含重复的数，现查找一个给定的数在不在数组，在的话给出最后一个出现的index

9. 堆排序相关

10. 动态规划

11. 一道算法，总共七八道题目吧(数据库，操作系统，缓存，数据结构)

算法：给定数组（数字可重复），求两数和为21的所有下标

数据库（mysql）：两种表引擎区别，索引，事务隔离级别

数据结构：redis中zset实现原理

操作系统：LRU原理

另外还有2,3个，一个是大文件放在有限内存排序，一个是负载均衡相关的，具体忘记了

12. mysql 两种存储引擎的区别

13.where a = x and b > y ,,, where a >x,,,,, where b >y 三种情况如何加索引。

14.四种事务隔离的区别

15.操作系统的页面置换算法有哪几种。

16.N 个集群节点 一致性哈希的长问道的问题。

17.大文件内存排序问题。

18.100 G文件 8G内存 如何排序。

19. 两个单链表求和

20. qps限制怎么做

21. 分布式锁的内容

22. 两个链表分别表示两个数，对这两个链表加和，产生一个新的链表

23. 给定一个价格数字列表，计算哪天买入，哪天卖出利润最大

24. http状态码/header/请求方法, 哪些请求方法是幂等的

25. https加密原理

26. . python协程原理/缺点

27.. python GIL, 为什么有GIL 还需要threading

28 . python gevent协程调度原理/缺点

29.. 设计一个文本编辑器实现插入/删除/查询

30.. 头条文章向用户推送避免重复推送的问题

31.. 工作中遇到的印象深刻的问题

32. 两种存储引擎

33. http的返回码

34. mysql 索引

35..链表操作(在线写代码)

3 6. http状态码

37. ->UDP / TCP 协议

38. ->链表操作 [手写代码]

39. ->乐观锁 / 悲观锁

40 ->三次握手

41. ->Socket

42. ->Hbase 底层原理

43. ->Map 底层原理

44. ->数据库分表 / 分库 / 集群

总结:比较注重CS修养、底层源码实现

45. redis内存不足，如何解决

46. mysql索引的使用和原理

47. . epoll和poll,select的区别

48. 给定一个二叉树，返回二叉树每层的最左的值。

49. 手里有一副扑克牌。按照下列规则吧他堆放桌上。一，拿出最上面的一张牌，放桌上，然后把接下来的一张牌放在扑克牌的最下面。循环，直到没有手牌。现在已知桌上牌的顺序。求原手牌的顺序。

50. 实现一个有顺序的LinkedList，实现deleteAll(int val)方法删除所有值为val的节点，和save(int val)方法添加一个值为val的节点。

51. 线程池的工作流程。线程的几种状态，什么情况会进入阻塞

52. JDK基础 集合 锁

53. Mysql的索引结构 聚簇索引

54. 限流。限流算法。令牌桶的不足

55. GC CMS，CMS的参数

56. JVM的一些排查工具jstat, jstack ,jmap

57. 工作中遇到的问题如何排查

58. LRU的实现

59. MySQL的索引实现方法

60. MySQL的存储引擎的区别

61. 一排盒子，每个盒子上标了能走的最大步数，从第一个盒子开始，判断能否走到最后的盒子

62. 栈，增加一个max方法；

63. 自己设计hash

64. 微博限定用户每次输入最多140个字符，用户如果传字符串很长的链接，怎么办

65. 两个线程，一个只能存有数组1、2、3和另一个存有a、b、c，然后通过调度，最终结果输出1a2b3c

66. 输入一个int数组，返回一个数组，其中奇数都在左边，偶数都在右边 附加条件 不使用额外的空间

67. 数据库如何建索引，和如何分库分表

68. 找到一个文件里出现次数最多的的数字，文件大小远大于内存容量

69. 给一个数组和一个整数，输出数组中所有和为给定整数的元素下标。

70. MySQL两种存储引擎的差异；

71. where a>1 and b>1;where a = 1; where b = 2,如何为这种条件语句建立索引；

72. LRU的实现。。

73. 介绍一个项目中遇到的问题。

74. 一个100G的文件，内存只有8G，如何给文件排序，文件内容都是按行存储的URL

75. 10G文件1G内存，

.打印重复数字

查找前topK

76. 链表反转，每次反转5个node

77. 使用linux shell统计log中访问url的数量并排序

78. 你遇到过哪些慢查询，并如何优化

79. tcp3次握手个四次挥手，tcp time wait，tcp还有哪些状态

80. 线程的状态有几种

81. StackOverflowError和OutOfMemoryError的区别和分别处理

82. synchronized和volatile的区别

83. synchronized和static的作用范围

84. 500表示什么

85. 长度为m的int64中找top n

86. 进程间通信方法有几种

87. HashTable的实现原理

88. Oracle相对于MySQL有什么优势

89. 微信抢红包功能设计（用了SingleThreadPool 他不满意）

90. qps限流 给了他两个方案 一个缓存，一个滑动窗口

91. 排序数组中计算重复数字出现的次数

92. kafka 的原理，以及使用中遇到的问题

93. 手写堆排序

94. 把所有的排序算法捋一遍

95. 会问有没有消息队列 分布式 缓存 一系列后端架构的经验

96. 算法题：在一个有序但元素可能重复的数组中，输入一个目标值，返回这个目标值的左边界和右边界。[1,1,2,2,2,2,2,4,5]，得到2的左右边界。

97. 说下项目里标签推荐算法实现

98. cookie与session区别; 没有cookie的话session是否有效

99. HTTP与HTTPS区别

100. HTTPS如何做到安全

101. MySQL存储引擎区别

102. MySQL索引结构优缺点

103. 讲volatile和可见性

104. 二叉搜索树转双向链表写代码

105. Ctrl c操作后的系统执行

106. k路归并排序

107. :lru原理要非常详细的数据结果和过程结构

108. 给个数组，只有重复的，求所有和为21的下标:不用n2的暴力算法，要考虑数字相同的下标，先说思路没问题才能共享屏幕写

109. 根据二叉树前序遍历生成avl树

110. 两个字符串a和b，找a中包涵b的最短的字符串

111. 索引

112. redis，主从同步和持久化方式

113. 限流

114. 快排的时间复杂度和优化

115. b+树特性

116. tcp过程，和udp的区别

117、编程题：string转double

118、hashmap实现方式

119、数据库事务、隔离级别

120. Hashmap原理

121. 寻找链表倒数第K个节点

122.. 二叉搜索树找第K大

123. . 选择排序

124. B+tree 和红黑树的区别

125. 堆排序的实现

126. 10G文件,每一行一个uint32数字。有一台1G内存的机器

-A.找出最大的K个数

-B.找出重复数字

127. 我手中有一堆扑克牌，但是观众不知道它的顺序。

第一步，我从牌顶拿出一张牌，放到桌子上。

第二步，我从牌顶再拿一张牌，放在手上牌的底部。

第三步，重复第一/二步的操作，直到我手中所有的牌都放到了桌子上。

最后，观众可以看到桌子上牌的顺序是13\12\11\10\9\8\7\6\5\4\3\2\1 请问，我刚开始拿在手里的牌的顺序是什么？

128.. 一道编程题，给出一个数组代表围柱的高度，求能围柱的最大的水量，思路和代码实现

129. 数据库，mongoDB和mysql。主要问了索引，给出一段where表达式如

130.. where a=?

131. Where b=?

132.. Where b=? and c=?

133. 上面三条语句创建几个索引

134.. Where b=? or c=?

135.. 这个查询可以用到索引吗

136.. Varchar（2000）创建索引会怎样

137.. Node和go语言的区别有优势

138. Express和gin框架的区别和优势

139. python的装饰器

140. mysql的innodb与myisam区别，索引

141. 10G文件，1G内存，找出最大的K个数，找出重复数

142. 链表快排:

8G文件1G内存，查找重复的数字

8G文件1G内存, 重复次数最多的数字

143. Xss csrf

144. .重定向，调转的前后端实现

145. innodb特性，主键索引 非主键索引区别

146. Ctrl c后操作系统做了什么

147. 4c8G机器 1000qps 怎么配置机器和线程池

148、out Of memory Error 跟 stack Over flow Error 的比较（可能单词没拼对）

149、数据库的隔离级别

150、java 的 array 跟 list的比较，底层原理上的

151、有sql的编程题，是问每门课程都大于80分的学生名字

152、java编程题，

a、反转一句英语的所有单词：hello world - > world hello

b、6 7 8 9 0 1 2 3 4 5，找一个数

153、还有tcp，三次握手；ssl。

154、对称性加密跟非对称性加密的比较，使用场景

155、数据库事务

156、java的多线程

157、涉及面好广 还有密码学的，加密算法RSA

158、搜索引擎倒排原理及实现方式

159、Redis高可用方案

160、主键与索引的联系和区别

161、索引的实现方式，为什么会采用这种数据结构

162、Redis缓存删除策略

163、编码实现LRU算法

164、 统计大量访问日志（分几百M 和 几百G的场景）；得出访问次数最多的前K个人 （单台机器实现）

165. 给你一个连续素数数组 让你找出增长比第k小的位置

166. 代码题：有序数组 A = [1, 2, 2, 3, 3, 5, 5, 5]，b = 5，在A中找出b的序号的上下界，考察思路和代码熟练度

167. 自我介绍、项目经历（会结合你的项目经历即兴问一些相关联的问题，以设计思路，框架性的思考为主）

168. HTTPS原理

169. Oracle MySQL特性、优劣势

170. Redis等前沿技术栈的知识面

171. 用redis做限流

172. Mysql各种基础知识

173. HTTP状态码 502/504

174. 系统设计题：微信扫码登录

175. mysql索引，

176. http状态码

177. java乐观锁悲观锁

176. 消息可达性和唯一消费

177. mybatis的Mapper接口映射到sql文件的实现原理

178. mysql联合索引,实现,优点

179. 一个算法题，两数组a,b，把a数组中在b数组出现的数字，按照它在B数组中出现的顺序进行位置调整。

180. 微博等热门评论，在分页到很深的时候，如何进行优化

181. 对简历所写的技能深度层次有准确表述，熟悉不要写成精通

182. go语言的三大特色，goroutine 调度方式，接口的底层实现，select 的触发顺序（如果多个通道同时准备好）等底层实现方面(该候选人主语言是golang)

183. 二叉树的最大距离（递归实现即可）

184. go语言底层socket 的调度算法（epoll 相关的编程注意点）(该候选人主语言是golang)

185. 数据库的索引建立实例，where 子句和order by 子句同时存在如何建立索引以及为什么

186. 数据库的事物隔离及其各自产生的赃读或幻读现象

187. 批量插入数据库有啥优化点，文件导入或单句执行（原因锁表开销少）

188. 项目中遇到的难点及解决办法

189. 结合redis 和本地缓存如何实现黑名单id过滤功能（考虑海量请求、同时会有读写情况）

190. 套接字底层选项功能及阻塞和非阻塞场景应用注意事项

191. 一个数组中取第K大的数

192. redis Zset数据结构、新增数据的时间复杂度

193. mysql的事务是怎么实现的

194. mysql 联合索引在什么情况能用到

195. 写一个线程安全的单例

196. 写一个熟悉的排序 说出时间复杂度

197. 有一个分布式不安全的文件系统，如何保证每次只有一个请求进行读写

198. 如何实现音乐随机播放

199. 登录的过程，cookie如何写入的

200. 301、302、500、502、504是什么含义

201. mysql相关：

引擎innodb特性，锁，事务

202. mq：

kafka，rocketmq，同步机制和事务机制

203. 编程实现的是泛型链表的添加和删除，主要是看敲代码的过程和怎样解释自己的代码

204. 删除链表中重复的结点

在一个排序的链表中，存在重复的结点，请删除该链表中重复的结点，重复的结点不保留，返回链表头指针。例如，链表1－＞2－＞3－＞3－＞4－4－＞5处理后为1->2->5

205. 输入一个二叉树，以及指定一个层次，从右向左输出这一层的所有节点

206. 设计一个LRU，尽可能的支持高并发

207. session 怎么用 cookie 实现的，session 和 cookie 的区别。

208. java 线程池是怎么工作的。

209. java bio nio 的区别

210. nio是怎么实现的

211. CMS的流程

212. CMS 的参数

213. 线上的 GC 是怎么配置的？

214. redis 主从怎么实现的。

215. 链表逆序，设计一个王者的组队系统（人选一起反馈的 看不懂是不是同一道题......）

216. sql什么时候不适合用索引，索引太多会有啥问题，

217. 同步io和异步io区别，

218. coding考察的是多个数组合并，follow up 如果数组个数非常多但是每个里面的内容比较少

219. http和https的区别，https的实现

220. 单链表，每k个进行一次翻转

221. 设计一个限流工具，每5s允许一个请求，多的请求丢弃

222. 设计一个限流工具，在第1，6，11，16秒允许一个请求，多的丢弃

223. 设计一个限流工具，在第1，6，11，16秒允许一个请求，多的排队等待

224、502 504

225、cookie，session

226、mysql优化

227、mysql MyISAM与innodb的区别

228、redis分布式锁

229、缓存一致性

230、mysql索引存储

231、jvm垃圾回收算法

232、go string的数据结构

233、goroutine 的调度

234、 数组中第k大的数

235、ORACLE 和mysql的区别

236、工作中Spring Aop的案例

237. activemq和rabbitmq的区别

238、 数据库中的索引和事务

239、 Go比Python的优缺点。

240、 Go的任务调度。

241、 Go的channel为什么可以被goroutine安全操作而不需要上锁。

242、 网络知识：TCP的三次握手、四次挥手。

243、 分布式锁。

244、 数据库的分布式事务。

245、 算法题：遍历树，求从根节点到叶子节点和为某值的路径。

246、 算法题：一排相邻的糖果屋，不能从相邻糖果屋买糖，请问最大能买多少颗糖？

输入[1,2,3,1] 输出4

247、 ctrl c过程；

248、 去除重复列表；

249、 单链表求和；

250、 504状态；

251、 http方法；

252、聚簇索引/非聚簇索引

253、数据隔离级别

254、读已提交怎么解决的幻读问题

255、数据库特性acid中的i(隔离性)怎么实现的

256、lock与sychronized的区别

257、sychronzied是可重入锁吗

258、非阻塞io的具体实现

259. tcp协议，实现36进制数，mysql的索引机制；

260. 字符串数组公共前缀

261. 5张扑克牌，判断是不是同花顺

262. 数据库问了前缀索引，为什么用b+树，不用b树或者红黑树

263. 进程线程协程区别

264. go垃圾回收

265. nginx惊群效应

266. io多路复用原理，在redis和mysql的实现

267. 数据库前缀索引

268. innodb和miysam存储实现区别

269. redis，rehash的实现

270. 设计一个历史浏览记录

# 二轮

1、hashmap底层原理，同步和锁的区别，数据库这些，还给了一道情景题，如何限制每分钟每个手机号短信发送数量

头条的面试跟其他公司不是一个套路，一般就问几个问题，几个里面还有一半是手写算法实现

2、mysql索引原理，联合索引

项目里的技术hbase原理

二叉搜索树的打印

二维码登陆设计

HashMap实现

发短信业务，1分钟内一个号一个业务1000条

3.   场景设计：设计一个直播中答题获百万奖金的设计。分析场景中的难点

4. 场景设计：设计一个IP数据库，表中包括IP数据和地址。用户可以通过IP地址查询到地点

5. oracle和mysql

6. 聚簇索引和非聚簇索引

7. B+树和B树的区别

8. 同步和锁的概念

9. volatile

10. 算法题leetcode 670

11. 介绍经历、java线程池基础知识、自己实现一个线程池，然后是二叉树的bfs，最大公共子串的长度，最大不重复子串的长度

12. 1.2.3的全排列

13. mysql索引的实现，b＋树的特点

14. session和cookie

15. 如果浏览器禁用cookie会发生什么

16. 为什么有跨域访问，跨域中的get和post有什么不同

17. mysql的mvcc，多版本并发控制

18. 算法题：二叉树前序遍历的递归和非递归解法

19. 一道sql题，已知一个表字段：<course, score , stumentName> 然后得出

语文 数学 英语 这样的纵向视图

分数平均值

分数最大值

分数最小值

20. redis master和slave 的同步机制

21. 消息队列如何保证可靠

22. 设计秒杀系统要求保证公平

23. Mac地址如果被改会怎么样

24. 路由寻址过程做了哪些事

25. HTTP1.1 2 4特性; HTTP里面的各种攻击及应对策略

27. 如何信任CA

28. Linux如何查看应用是否在运行

29. 代码题:给一个亿级用户登录登出时间戳日志，统计用户在线量峰值及持续时间

30. 主键索引和非主键索引的区别

31. 数据库事务的隔离级别

32. redis和memcached

33. 数据库查询优化

34. 编程题，贪心算法

35. 数据库索引：表有3列a,b,c，需支持查询（a,b）(b) (b,c)，索引如何建

36. 反转二叉树

37. 数据库表（id, name, parentId）如何组装成树?

38. 问题有tcp 与 udp的区别，网页应用中的登录的实现，

39、http 的请求方式 GET POST 跟其它的

40、最后的编程题，两个单链表：

1）:a b， a x->y->z->m->n->p->q

2）b k->l->p->q，后面的p 跟q是同样的对象

41、要求在O(n)时间复杂度，O(1)空间复杂度内求出相同片段的长度

42、不同字符最大子串根据给定的输入字符串，返回包含有最多k个不同字符的子字符串的长度及子字符串在输入字符串的起始下标。如果有多个符合条件的子字符串，则返回第一个子字符串的下标。输入描述输入为一个待校验的字符串输出描述两个数字，分表表示子字符串的长度以及子字符串首字符在输入字符串的下标

43、限流 zset redis pipieline http请求过程详解 mysql 索引 堆排序 求前k大个数（一二面）

1、一千个棋子，甲先取乙后取，每次最多取七个最少取一个，问是否有一个方案让甲一定赢

2、3×7的格子，从左上角到右下角，只能往右或者往下，有多少种走法

3、一个不均匀硬币，如何抛出均匀概率

4、然后有一个生成0到13随机数的算法，如何用它均匀生成0到9随机数

5、两千万高考生成绩如何排序

6、用链表表示的大数求和

44、一千个棋子，甲先取乙后取，每次最多取七个最少取一个，问是否有一个方案让甲一定赢

45、3×7的格子，从左上角到右下角，只能往右或者往下，有多少种走法

46、一个不均匀硬币，如何抛出均匀概率

47、然后有一个生成0到13随机数的算法，如何用它均匀生成0到9随机数

48、两千万高考生成绩如何排序

49、用链表表示的大数求和

50. Redis的分布式锁，单节点和集群的优缺

51. 自己项目里的mysql规模

52. 10g文件，1g内存，求top k和 重复数字

53. .两个单向链表按照十进制求和，不能使用已经有的数据结构和注意链表方向，比如1-＞3-＞5-＞9，5-＞4，等于1-＞4-＞1-＞3

54、MySQL中的事务隔离级别

55、协程的原理和实现

56、gevent中的协程的实现

57、反转链表

58、距离矩阵，从左上开始，只能往右或往下，求从左上到右下的最短距离

59、内存段页式管理，内存上数据的大序与小序

60、tcp协议timewait状态的作用

# 三轮

1、设计一个订单查询加缓存优化的

2、leetcode 670，然后系统设计题，设计群消息已读功能

3、都是结合简历问的 项目经历、实现的方法、遇到的难点，穿插着所用的技术。代码是考察设计一下搜索模块。

4. 算法题： 一个无序的数组，求这个数组的中位数

5. redis实现分布式锁的原理，redis实现会带来什么样的问题

6..code review

7. 数据库索引+前缀匹配

8. 缓存设计

9. 并发控制（乐观锁，悲观锁）

10. xss && csfr

11. java写单例模式，

12. 链表反转1->2->3->4 变 2->1->4->3,

13. 简单的字符串中' '替换'%20'

14. 推送的频率控制

15. 抖音评论列表的设计及缓存实现

16. MySQL索引怎么建、

17. http头、

18. http header

19.    问了redis数据结构和实现

20.    问hashmap

21.    浅拷贝深拷贝实现

# 四轮

1. 对微服务体系架构的认识和理解。

2. 平时项目中自己发现并做出的优化和改进。

3. 在微服务体架构中，如何能保证接口的可靠性，（我这边主要从接口幂等性校验和安全角度回答）。

4. 分布式事务，两阶段提交协议，失败重试补偿。