多态的实现原理

虚函数表，延申到Linux内核文件系统实现

指针和引用的区别

引用的本质是常指针

Struct{

Char

Float

Short

char

}

问sizeof() 字节对齐 = 12

const char\* str[] = {“aa”,”bbb”,”cccc”} 指针数组

问sizeof(str)=8\*3=24

问需要多大内存（3+4+5）

问ip的字符串转数字

inet\_pton

C程序内存的情况？

text,rodata,data,bss,heap（向上）,.so,stack（向下）,env，内核区

如何实现不能继承的类

final

构造函数private

算法：一个N长度的数组，有M个取值，其中一个值在其中出现了50%以上，如何找出这个数？

计数排序思想

描述一下三次握手和四次挥手

为什么握手是三次，挥手是四次？

因为需要将被动端的数据发完，才能关闭连接。

Close\_wait？Time\_Wait？

Close\_wait：被动端收到fin并回复ack以后进入这个状态，此时输出缓冲区不再允许写数据。

Time\_wait：主动端发出最后一个ACK后变。等待2MSL后关闭socket。是为了确定对方能收到这个ACK。如果被动端未收到ACK，则会认为主动端没收到fin，则会再发一次fin给主动端。设置这个time\_wait是为了确保最后一个ack能发出去。

用过线程吗？如何处理多线程数据共享

加锁

无锁队列？

IPC有哪些方法？

Pipe,fifo,mmap,domain

分别什么时候使用？

。。。。。。。讲述了一下

select,poll,epoll的区别

epoll et epoll lt的区别，什么时候使用

二面

const char\* str[] = { "aa","bbb","cccc" };

const char\*\* pstr;

printf("str %d pstr = %d\n", sizeof(str),sizeof(pstr));

24！！！str是一个指针数组

class A

{

public:

void function3()

{

std::cout<<"hello world!"<<std::endl;

//C++ 优化：没有使用this指针，不会报错

flag = 0;

}

int flag;

};

int main()

{

A\* a = NULL;

a->function3();

}

实现一个String类

## **C++ 面试一般考察的内容**

# 1. 语言基础

C++ 虚函数这是面试初、中级 C++ 职位一个概率 95% 以上的面试题。一般有以下几种问法：

1.在有继承关系的父子类中，构建和析构一个子类对象时，父子构造函数和析构函数的执行顺序分别是怎样的？

2.在有继承关系的类体系中，父类的构造函数和析构函数一定要申明为 virtual 吗？如果不申明为 virtual 会怎样？

3.什么是 C++ 多态？C++ 多态的实现原理是什么？

4.什么是虚函数？虚函数的实现原理是什么？

5.什么是虚表？虚表的内存结构布局如何？虚表的第一项（或第二项）是什么？

6.菱形继承（类 D 同时继承 B 和 C，B 和 C又继承自A）体系下，虚表在各个类中的布局如何？如果类B和类C同时有一个成员变了m，m如何在D对象的内存地址上分布的？是否会相互覆盖？

另外，时至今日，你一定要熟悉 C++11/14/17 常用的语言特性和类库，这里简单地列一下：

统一的类成员初始化语法与 std::initializer\_list

注解标签（attributes）

final/override/=default/=delete 语法

auto 关键字

Range-based 循环语法

结构化绑定

stl 容器新增的实用方法

std::thread

线程局部存储 thread\_local

线程同步原语 std::mutex、std::condition\_variable 等

原子操作类

智能指针类

std::bind/std::function

C++11/14 网上的资料已经很多了，C++17 的资料不多，重头戏还是 C++11 引入的各种实用特性，这就给读者推荐一本我读过的：

《深入理解 C++11：C++11 新特性解析与应用》

《深入应用 C++11：代码优化与工程级应用》

《C++17 完全指南》

《Cpp 17 in Detail》

# 2. 算法与数据结构基础

说到算法和数据结构，对于社招人士和对于应届生一般是不一样的，对于大的互联网公司和一般的小的企业也是不一样的。下面根据我当面试官面试别人和找工作被别人面试经验来谈一谈。

先说考察的内容，除了一些特殊的岗位，常见的算法和数据结构面试问题有如下：

1.排序（常考的排序按频率考排序为：快速排序 > 冒泡排序 > 归并排序 > 桶排序）

一般对于对算法基础有要求的公司，如果你是应届生或者工作经验在一至三年内，以上算法如果写不出来，给面试官的影响会非常不好，甚至直接被 pass 掉。对于工作三年以上的社会人士，如果写不出来，但是能分析出其算法复杂度、最好和最坏的情况下的复杂度，说出算法大致原理，在多数面试官面前也可以过的。注意，如果你是学生，写不出来或者写的不对，基本上面试过不了。

## 1.二分查找

二分查找的算法尽量要求写出来。当然，大多数面试官并不会直接问你二分查找，而是结合具体的场景，例如如何求一个数的平方根，这个时候你要能想到是二分查找。我在2017年年底，面试agora时，面试官问了一个问题：如何从所有很多的ip地址中快速找个某个ip地址。

## 2.链表

无论是应届生还是工作年限不长的社会人士，琏表常见的操作一定要熟练写出来，如链表的查找、定位、反转、连接等等。还有一些经典的问题也经常被问到，如两个链表如何判断有环（我在2017年面试饿了么二面、上海黄金交易所一面被问过）。链表的问题一般不难，但是链表的问题存在非常多的“坑”，如很多人不注意边界检查、空链表、返回一个链表的函数应该返回链表的头指针等等。

## 3.队列与栈

对于应届生来说一般这一类问的比较少，但是对于社会人士尤其是中高级岗位开发，会结合相关的问题问的比较多，例如让面试者利用队列写一个多线程下的生产者和消费者程序，全面考察的多线程的资源同步与竞态问题（下文介绍多线程面试题时详细地介绍）。栈一般对于基础要求高的面试，会结合函数调用实现来问。即函数如何实现的，包括函数的调用的几种常见调用方式、参数的入栈顺序、内存栈在地址从高向低扩展、栈帧指针和栈顶指针的位置、函数内局部变量在栈中的内存分布、函数调用结束后，调用者和被调用者谁和如何清理栈等等。某年面试京东一基础部门，面试官让写从0加到100这样一个求和算法，然后写其汇编代码。

## 4.哈希表

哈希表是考察最多的数据结构之一。常见的问题有哈希冲突的检测、让面试者写一个哈希插入函数等等。基本上一场面试下来不考察红黑树基本上就会问哈希表，而且问题可浅可深。我印象比较深刻的是，当年面试百度广告推荐部门时，二面问的一些关于哈希表的问题。当时面试官时先问的链表，接着问的哈希冲突的解决方案，后来让写一个哈希插入算法，这里需要注意的是，你的算法中插入的元素一定要是通用元素，所以对于 C++ 或者 Java 语言，一定要使用模板这一类参数作为哈希插入算法的对象。然后，就是哈希表中多个元素冲突时，某个位置的元素使用链表往后穿成一串的方案。最终考察 linux 下 malloc（下面的ptmalloc） 函数在频繁调用造成的内存碎片问题，以及开源方案解决方案 tcmalloc 和 jemalloc。总体下来，面试官是一步步引导你深入。（有兴趣的读者可以自行搜索，网上有很多相关资料）

## 5.树

面试高频的树是红黑树，也有一部分是B树（B+树）。红黑树一般的问的深浅不一，大多数面试官只要能说出红黑树的概念、左旋右旋的方式、分析出查找和插入的平均算法复杂度和最好最坏时的算法复杂度，并不要写面试者写出具体代码实现。一般 C++ 面试问 stl 的map，java 面试问 TreeMap 基本上就等于开始问你红黑树了，要有心里准备。笔者曾经面试爱奇艺被问过红黑树。B树一般不会直接问，问的最多的形式是通过问 MySQL 索引实现原理（数据库知识点将在下文中讨论）。笔者面试腾讯看点部门二面被问到过。

## 6.图

图的问题就我个人面试从来没遇到过，不过据我某位哥哥所说，他在进三星电子之前有一道面试题就是深度优先和广度优先问题。

7.其他的一些算法

如 A\*寻路、霍夫曼编码也偶尔会在某一个领域的公司的面试中被问到。

# 3. 编码基本功

还有一类面试题不好分类，笔者姑且将其当作是考察编码基本功，这类问题既可以考察算法也可以考察你写代码基本素养，这些素养不仅包括编码风格、计算机英语水平、调试能力等，还包括你对细节的掌握和易错点理解，如有意识地对边界条件的检查和非法值的过滤。请读者看以下的代码执行结果是什么？

for(char i = 0; i < 256; ++i)   
{  
 printf("%d\n", i);  
}

下面再列举几个常见的编码题：

（1）实现一个 memmov 函数 这个题目考查点在于 memmov 函数与 memcpy 函数的区别，这两者对于源地址与目标地址内存有重叠的这一情况的处理方式是不一样的。

（2）实现 strcpy 或 strncpy 函数，这类函数写出来没啥难度，但是**一定要注意边界条件检查**，还有就是**其返回值**一定要是目标内存地址，以支持所谓的链式拷贝。**这两类细节没考虑到基本上算不通过。**

即：

strcpy(dest3, strcpy(dest2, strcpy(dest1, src1)));

（3）实现atoi函数 这个函数的签名如下：

int atoi(const char\* p);

容易疏忽的地方有如下几点：

小数点问题，如数字 0.123 和 .123 都是合法的；

正负号问题，如 +123 和 -123；

考虑如何识别第一个非法字符问题，如 123Z89，则应转换成应该 123。

# 4. 多线程开发基础

现如今的多核 CPU 早已经是司空见惯，而多线程编程早已经是“飞入寻常百姓家”。对于大多数桌面应用（与 Web 开发相对），尤其是像后台开发这样的岗位，且**面试者是社会人员**（有一定的工作经验），如果面试者不熟悉多线程编程，那么一般会被直接 pass 掉。

这里说的 **“熟悉多线程编程”** 到底熟悉到什么程度呢？一般包括：知道何种场合下需要新建新的线程、线程如何创建和等待、线程与进程的关系、线程局部存储（TLS 或者叫 thread local）、多线程访问资源产生竞态的原因和解决方案等等、熟练使用所在操作系统平台提供的线程同步的各种原语。

对于 C++ 开发者，你需要：

对于 Windows 开发者，你需要熟练使用 Interlock系列函数、CriticalSection、Event、Mutex、Semphore等API 函数和两个重要的函数 WaitForSingleObject、WaitForMultipleObjects。

对于 Linux 开发者，你需要熟练使用 mutex、semphore、condition\_variable、read-write-lock 等操作系统API。

可以使用 C++ 实现一个简单的线程池，当然支持优先级、动态创建线程功能就更好了。

# 5. 数据库

数据库知识一般在大的互联网企业对应届生不做硬性要求，对于小的互联网企业或社会人士一般有一定的要求。其要求一般包括：

（1）熟悉基本 SQL 操作 包括增删改查（insert、delete、update、select语句），排序 order，条件查询（where 子语句），限制查询结果数量（LIMIT语句）等

（2）稍微高级一点的 SQL 操作（如 Group by，in，join，left join，多表联合查询，别名的使用，select 子语句等）

（3）索引的概念、索引的原理、索引的创建技巧

（4）数据库本身的操作，建库建表，数据的导入导出

（5）数据库用户权限控制（权限机制）

（6）MySQL的两种数据库引擎的区别

（7）SQL 优化技巧

以上属于对开发的基本的数据库知识要求，你可以找一本相关入门级的数据库图书学习即可。

高级开发除了以上要求还要熟悉高可用 **MySQL、主从同步、读写分离、分表分库** 等技术，这些技术的细节一定要清楚，它们是你成为技术专家或者高级架构的必备知识。

**我们在实际面试时，在讨论高可用服务服务方案时，很多面试者也会和我们讨论到这些技术，但是不少面试者只知道这些技术的大致思想，细节往往说不清楚，细节不会就意味着你的高可用方案无法落地，企业需要可以落地的方案。**

这些技术我首推 **《高性能 MySQL》** 这本书，这本书高级开发者一定要通读的，另外还有 2 本非常好的图书也推荐一下：一本是 **《MySQL 排错指南》，** 读完这本书以后，你会对整个“数据库世界”充满了清晰的认识；另外一本是 **《数据库索引设计与优化》，** 这本书读起来非常舒服，尤其是对于喜欢算法和数据结构的同学来说。

# 6. 网络编程

网络编程这一块，对于应届生或者初级岗位一般只会问一些基础网络通信原理（如三次握手和四次挥手）的 socket  基础 API 的使用，客户端与服务器端网络通信的流程（回答 【客户端创建 socket -> 连接 server ->收发数据；服务器端创建 socket -> 绑定 IP 和端口号 -> 启动侦听 ->接受客户端连接 ->与客户端通信收发数据】即可）、TCP 与 UDP 的区别等等。

对于工作经验三年以内的社会人士或者一些中级面试者一般会问一些稍微重难点问题，如 select 函数的用法，非阻塞 connect 函数的写法，epoll 的水平和边缘模式、阻塞socket与非阻塞 socket 的区别、send/recv 函数的返回值情形、REUSE\_ADDR 选项等等。Windows 平台可能还会问 WSAEventSelect 和 WSAAsyncSelect 函数的用法、完成端口（IOCP 模型）。

对于三年以上尤其是“号称”自己设计过服务器、看过开源网络通信库代码的面试者，面试官一般会深入问一些问题，这类问题要么是实际项目中常见的难题或者网络通信细节，根据我的经验，一般有这样一些问题：

1.nagle 算法；

2.keepalive 选项；

3.Linger 选项；

4.对于某一端出现大量 CLOSE\_WAIT 或者 TIME\_WAIT 如何解决；

5.通讯协议如何设计或如何解决数据包的粘包与分片问题；

6.心跳机制如何设计；（可能不会直接问问题本身，如问如何检查死链）

7.断线重连机制如何设计；

8.对 IO Multiplexing 技术的理解；

9.收发数据包正确的方式，收发缓冲区如何设计；

10.优雅关闭；

11.定时器如何设计；

12.epoll 的实现原理。

举两个例子，让读者感受一下：

B 站面试曾被问的一个问题：如果 A 机器与 B 机器网络 connect 成功后从未互发过数据，此时其中一机器突然断电，则另外一台机器与断电的机器之间的网络连接处于哪种状态？

# 7. 内存数据库&缓存技术

时下以 NoSql key-value 为思想的内存数据库大行其道，广泛地用于各种后台项目开发。所以**熟悉一种或几种内存数据库程序已经是面试后台开发的基本要求，** 而这当中以 Redis 为最典型代表，这里以 Redis 为例。

第一层面一般是对 Redis 的基础用法的考察 如考察 Redis 支持的基础数据类型、Redis的数据持久化、事务等。

第二层面不仅考察 Redis 的基础用法，还会深入到 Redis 源码层面上，如 Redis 的网络通信模型、Redis 各种数据结构的实现等等。

Redis 高可用技术、cluster、哨兵策略等。

笔者以为，无论是从找工作应付面试还是从提高技术的角度，Redis 是一个非常值得学习的开源软件，希望广大读者有意识地去了解、学习它。

另外一些像分布式、RPC、MQ 等技术和 C++ 本身关系不是很紧密，这里就不罗列了。

# 8. 项目经验

除了社会招聘和一些小型的企业，一般的大型互联网公司对应届生不会做过多的项目经验要求，而是希望他们算法与数据结构等基础扎实、动手实践能力强即可。对于一般的小公司，对于应届生会要求其至少熟练使用一门编程语言以及相应的开发工具，号称熟悉 Linux C++ 开发的面试者，不熟悉 GDB 调试基本上不是真正的熟悉 Linux C++ 开发；号称熟悉汇编或者反汇编，不熟悉 IDA 或者 OllyDbg，基本上也是名不符实的；号称熟悉 VC++ 开发，连 F8、F9、F10、F11、F12 等快捷键不熟悉也是难以经得住面试官的提问的。**受疫情影响，很多面试都改成了线上面试，当你写算法题时如果你对开发工具不熟悉，面试官基本一下子就能看出来。** 这点请大家注意。

这里给一些学历不算好，学校不是非常有名，尤其是二本以下的广大想进入 IT 行业的同学一个建议，在大学期间除了要学好计算机专业基础知识以外，一定要熟练使用一门编程语言以及相应的开发工具。

关于项目经验，许多面试者认为一定要是自己参与的项目，其实也可以来源于你学习和阅读他人源码或开源软件的源码，如果你能理解并掌握这些开源软件中的思想和技术，在面试的时候能够与面试官侃侃而谈，面试官也会非常满意的。

很多同学可能纠结大学或者研究生期间要不要跟着导师做一些项目。当然，如果这些项目是课程要求，那么你必须得参加；如果这些项目是可以选择性的，尤其是一些仅仅拿着第三方的库进行所谓的包装和加工，那么建议可以少参加一些。