# 书籍信息

|  |  |  |  |
| --- | --- | --- | --- |
| 书名 | 版本 | 简写 | 偏移量\* |
| 《C++ Primer》 | 中文第5版 |  | 26 |
| 《深入理解计算机系统》 | 中文第3版 | CSAPP |  |
| 《现代操作系统》 | 中文第4版 |  |  |
| 《计算机网络：自顶向下方法》 | 中文第6版 |  |  |
| 《Unix网络编程 卷1：套接字联网API》 | 中文第3版 | UNP |  |
| 《数据库系统概念》 | 中文第6版 |  |  |
| 《超全面的后端C/C++面经整理》 |  |  |  |

\*偏移量：指本人所用电子版PDF的页码和书籍实际页码的差值，比如PDF中为第71页，书籍实际页码为45页，那么差值为71-45=26。本文出于阅读和跳转方便，标注的页码均为电子版PDF的页码。

# C++

## 指针和引用的区别

1. 《C++ Primer》第71~75页
2. 《超全面》第0问
3. 浅谈C++中指针和引用的区别：<https://www.cnblogs.com/dolphin0520/archive/2011/04/03/2004869.html>

|  |  |  |
| --- | --- | --- |
|  | 指针 | 引用 |
|  | 指针本身就是一个对象 | 引用本身并非对象，它只是为一个已经存在的对象所起的别名 |
|  | 允许对指针赋值和拷贝，在指针的生命周期内其可以先后指向不同的对象，给指针赋值就是令其存放一个新地址，指向一个新对象 | 一旦定义了引用，就无法令其再绑定到另外的对象，之后每次使用该引用都是访问其最初绑定的那个对象 |
|  | 指针无需在定义时赋初值，指针可以为空 | 引用必须初始化，且引用不能为空 |
|  | sizeof得到的是指针自身的大小，一般为4字 | sizeof得到的引用大小取决于被引用对象大小 |
|  | 指针可以有多级 | 引用只能有一级 |
|  | 可以用const指针 | 没有const引用 |
|  | 指针传参时需要解引用才能修改参数 | 引用传参时可以直接修改参数 |

## 在函数参数传递时，何时用指针，何时用引用，二者的区别

《超全面》第1问

1. **使用指针传递的场景**

需要返回函数内局部变量的内存的时候用指针（返回局部变量的引用是没有意义的）。使用指针传参需要开辟内存，用完要记得释放指针，不然会内存泄漏。

1. **使用引用传递的场景**

（1）对栈空间大小比较敏感（比如递归）时使用引用，使用引用传递无需创建临时变量，开销更小。

（2）类对象作为参数传递的时候使用引用，这是 C++ 类对象传递的标准方式。

C++ 值传递、指针传递、引用传递详解： https://www.cnblogs.com/yanlingyin/archive/2011/12/07/2278961.html

1. **值传递：**形参是实参的拷贝，改变形参的值并不会影响外部实参的值。从被调用函数的角度来说，值传递是单向的（实参->形参），参数的值只能传入，不能传出。当函数内部需要修改参数，并且不希望这个改变影响调用者时，采用值传递。
2. **指针传递：**形参为指向实参地址的指针，当对形参的指向操作时，就相当于对实参本身进行的操作。
3. **引用传递：**形参相当于是实参的“别名”，对形参的操作其实就是对实参的操作，在引用传递过程中，被调函数的形式参数虽然也作为局部变量在栈中开辟了内存空间，但是这时存放的是由主调函数放进来的实参变量的地址。被调函数对形参的任何操作都被处理成间接寻址，即通过栈中存放的地址访问主调函数中的实参变量。正因为如此，被调函数对形参做的任何操作都影响了主调函数中的实参变量。

## 堆和栈有什么区别

## 堆和栈哪个更快，为什么

## new和delete是如何实现的，new和malloc的异同

## 既然有了malloc/free，C++中为什么还需要new/delete

## C和C++的区别

## C++和Java的联系与区别，包括语言特性、垃圾回收、应用场景等

## Java的垃圾回收机制

## C++和Python的区别

## struct和class的区别

## define和const的联系与区别（编译阶段、安全性、内存占用等）

## C++中const的用法（定义，用途）

## C++ 中的 static 用法和意义

## 计算下面几个类的大小：

1. class A {};
2. int main(){
3. cout<<sizeof(A)<<endl;// 输出 1;
4. A a;
5. cout<<sizeof(a)<<endl;// 输出 1;
6. return 0;
7. }

空类的大小是 1， 在 C++ 中空类会占一个字节，这是为了让对象的实例能够相互区别。具体来说，空类同样可以被实例化，并且每个实例在内存中都有独一无二的地址，因此，编译器会给空类隐含加上一个字节，这样空类实例化之后就会拥有独一无二的内存地址。当该空白类作为基类时，该类的大小就优化为 0 了，子类的大小就是子类本身的大小。这就是所谓的空白基类最优化.

空类的实例大小就是类的大小，所以 sizeof(a)=1 字节, 如果 a 是指针，则 sizeof(a) 就是指针的大小，即 4 字节

1. class A { virtual Fun(){} };
2. int main(){
3. cout<<sizeof(A)<<endl; // 输出 4(32位机器)/8(64位机器);
4. A a;
5. cout<<sizeof(a)<<endl; // 输出 4(32位机器)/8(64位机器);
6. return 0;
7. }

因为有虚函数的类对象中都有一个虚函数表指针 \_\_vptr，其大小是 4 字节

1. class A { static int a; };
2. int main(){
3. cout<<sizeof(A)<<endl;// 输出 1;
4. A a;
5. cout<<sizeof(a)<<endl;// 输出 1;
6. return 0;
7. }

静态成员存放在静态存储区，不占用类的大小, 普通函数也不占用类大小

1. class A { int a; };
2. int main(){
3. cout<<sizeof(A)<<endl;// 输出 4;
4. A a;
5. cout<<sizeof(a)<<endl;// 输出 4;
6. return 0;
7. }
8. class A { static int a; int b; };;
9. int main(){
10. cout<<sizeof(A)<<endl;// 输出 4;
11. A a;
12. cout<<sizeof(a)<<endl;// 输出 4;
13. return 0;
14. }

静态成员 a 不占用类的大小，所以类的大小就是 b 变量的大小 即 4 个字节

## C++ 的 STL 介绍（这个系列很重要，建议侯捷的书籍和视频），包括内存管理 allocator，函数，实现机理，多线程实现等

## STL 源码中的 hash 表的实现

## 解决哈希冲突的方式

## STL 中 unordered\_map 和 map 的区别

## STL 中 vector 的实现

## vector 使用的注意点及其原因，频繁对 vector 调用 push\_back() 对性能的影响和原因

## C++ 中 vector 和 list 的区别

## C++ 中的重载和重写的区别：

## C ++ 内存管理（热门）

## 介绍面向对象的三大特性，并且举例说明每一个

## 多态的实现（和下个问题一起回答）

## C++ 虚函数相关（虚函数表，虚函数指针），虚函数的实现原理（热门，重要）

## 实现编译器处理虚函数表应该如何处理

## 基类的析构函数一般写成虚函数的原因

## 构造函数为什么一般不定义为虚函数

## 构造函数或者析构函数中调用虚函数会怎样

## 纯虚函数

## 静态绑定和动态绑定的介绍

## 深拷贝和浅拷贝的区别（举例说明深拷贝的安全性）

## 对象复用的了解，零拷贝的了解

## 介绍 C++ 所有的构造函数

## 什么情况下会调用拷贝构造函数（三种情况）

## 结构体内存对齐方式和为什么要进行内存对齐？

## 内存泄露的定义，如何检测与避免？

## C++ 的智能指针有哪些

## 调试程序的方法

## 遇到 coredump 要怎么调试

## inline 关键字说一下 和宏定义有什么区别

## 模板的用法与适用场景，实现原理

## 成员初始化列表的概念，为什么用成员初始化列表会快一些（性能优势）

## 用过 C11 吗，知道 C11 新特性吗？（有面试官建议熟悉 C11）

## C++ 的调用惯例（简单一点 C++ 函数调用的压栈过程）

## C++ 的四种强制转换

## string 的底层实现

## 一个函数或者可执行文件的生成过程或者编译过程是怎样的

## set，map 和 vector 的插入复杂度

## 定义和声明的区别

## typdef 和 define 区别

## 被 free 回收的内存是立即返还给操作系统吗？为什么？

## 引用作为函数参数以及返回值的好处

## 友元函数和友元类

## 说一下 volatile 关键字的作用

## STL 中的 sort() 算法是用什么实现的，stable\_sort() 呢

## vector 会迭代器失效吗？什么情况下会迭代器失效？

## 为什么 C++ 没有实现垃圾回收？

# 计算机网络

## 建立 TCP 服务器的各个系统调用

## 继上一题，说明 socket 网络编程有哪些系统调用？其中 close 是一次就能直接关闭的吗，半关闭状态是怎么产生的？

## 对路由协议的了解与介绍

## UDP 如何实现可靠传输

## TCP 和 UDP 的区别

## TCP 和 UDP 相关的协议与端口号

## TCP（UDP，IP）等首部的认识（http 请求报文构成）

## 网页解析的过程与实现方法

## 在浏览器中输入 URL 后执行的全部过程

## 网络层分片的原因与具体实现

## TCP 的三次握手与四次挥手的详细介绍（热门问题）

## TCP 握手以及每一次握手客户端和服务器端处于哪个状态

## 为什么使用三次握手，两次握手可不可以？

## TIME\_WAIT 的意义（为什么要等于 2MSL）

## 超时重传机制（不太高频）

## TCP 怎么保证可靠性？

## 流量控制的介绍，采用滑动窗口会有什么问题（死锁可能，糊涂窗口综合征）？

## TCP 滑动窗口协议

## 拥塞控制和流量控制的区别

## TCP 拥塞控制，算法名字？（极其重要）

## HTTP协议与 TCP 的区别与联系

## HTTP/1.0和HTTP/1.1 的区别

## http 的请求方法有哪些？get 和 post 的区别

## http 的常见状态码和含义

## http 和 https 的区别，由 http 升级为 https 需要做哪些操作

## https 的具体实现，怎么确保安全性

## TCP 三次握手时的第一次的 seq 序号是怎样产生的

## 一个机器能够使用的端口号上限是多少，为什么？可以改变吗？那如果想要用的端口超过这个限制怎么办？

## 对称密码和非对称密码体系

## 数字证书的了解（高频）

## 服务器出现大量 close\_wait 的连接的原因以及解决方法

## 消息摘要算法列举一下，介绍 MD5 算法，为什么 MD5 是不可逆的，有什么办法可以加强消息摘要算法的安全性让它不那么容易被破解呢？

## 单条记录高并发访问的优化

## 介绍一下 ping 的过程，分别用到了哪些协议

## TCP/IP 的粘包与避免介绍一下

## 说一下 TCP 的封包和拆包

## 一个 ip 配置多个域名，靠什么识别？

## 服务器攻击（DDos 攻击）

## DNS 的工作过程和原理

## OSA 七层协议和五层协议，分别有哪些

## IP 寻址和 MAC 寻址有什么不同，怎么实现的

# 数据库

## 关系型和非关系型数据库的区别（低频）

## 什么是非关系型数据库（低频）

## 说一下 MySQL 执行一条查询语句的内部执行过程？

## 数据库的索引类型

## 说一下事务是怎么实现的

## MySQL 怎么建立索引，怎么建立主键索引，怎么删除索引

## 索引的优缺点，什么时候使用索引，什么时候不能使用索引（重点）

## 索引的底层实现（重点）

## B 树和 B + 树的区别（重点）

## 索引最左前缀 / 最左匹配

## Mysql 的优化（索引优化，性能优化，高频）

## MYSQL 数据库引擎介绍，innodb 和 myisam 的特点与区别

## 数据库中事务的 ACID（四大特性都要能够举例说明，理解透彻，比如原子性和一致性的关联，隔离性不好会出现的问题）

## 什么是脏读，不可重复读和幻读？

## 数据库的隔离级别，mysql 和 Oracle 的隔离级别分别是什么（重点）

## 数据库连接池的作用

## Mysql 的表空间方式，各自特点

## 分布式事务

## 数据库的范式

## 数据的锁的种类，加锁的方式

## 什么是共享锁和排他锁

## 分库分表的理解和简介

## 数据库高并发的解决方案

## 乐观锁与悲观锁解释一下

## 乐观锁与悲观锁是怎么实现的

## 对数据库目前最新技术有什么了解吗

# Linux

## Linux 的 I/O 模型介绍以及同步异步阻塞非阻塞的区别（超级重要）

## 文件系统的理解（EXT4，XFS，BTRFS）

## EPOLL 的介绍和了解

## IO 复用的三种方法（select,poll,epoll）深入理解，包括三者区别，内部原理实现？

## Epoll 的 ET 模式和 LT 模式（ET 的非阻塞）

## 查询进程占用 CPU 的命令（注意要了解到 used，buf，代表意义）

## linux 的其他常见命令（kill，find，cp 等等）

## shell 脚本用法

## 硬连接和软连接的区别

## 文件权限怎么看（rwx）

## 文件的三种时间（mtime, atime，ctime），分别在什么时候会改变

## Linux 监控网络带宽的命令，查看特定进程的占用网络资源情况命令

## Linux 中线程的同步方式有哪些？

## 怎么修改一个文件的权限

## 查看文件内容常用命令

## 怎么找出含有关键字的前后 4 行

## Linux 的 GDB 调试

## coredump 是什么，怎么才能 coredump

## tcpdump 常用命令

## crontab 命令

## 查看后台进程

# 操作系统

## 进程与线程的区别和联系（重点）

## Linux 理论上最多可以创建多少个进程？一个进程可以创建多少线程，和什么有关

## 冯诺依曼结构有哪几个模块？分别对应现代计算机的哪几个部分？

## 进程之间的通信方法有哪几种（重点）

## 进程调度方法详细介绍

## 进程的执行过程是什么样的，执行一个进程需要做哪些工作

## 操作系统的内存管理说一下

## 实现一个 LRU 算法

## 死锁产生的必要条件（怎么检测死锁，解决死锁问题）

## 死锁的恢复

## 什么是饥饿

## 如果要你实现一个 mutex 互斥锁你要怎么实现？

## 线程之间的通信方式有哪些？ 进程之间的同步方式又哪些？

## 什么时候用多进程，什么时候用多线程

## 文件读写使用的系统调用

## 孤儿进程和僵尸进程分别是什么，怎么形成的？

## 说一下 PCB / 说一下进程地址空间 /

## 内核空间和用户空间是怎样区分的

## 线程是如何同步的（尤其是如果项目中用到了多线程，很大可能会结合讨论）

## 同一个进程内的线程会共享什么资源？

## 异常和中断的区别

## 一般情况下在 Linux/windows 平台下栈空间的大小

## 虚拟内存的了解

## 服务器高并发的解决方案

## 协程了解吗（高频）

## 协程的底层是怎么实现的，怎么使用协程？

## 进程的状态以及转换图

## 在执行 malloc 申请内存的时候，操作系统是怎么做的？/ 内存分配的原理说一下 / malloc 函数底层是怎么实现的？/ 进程是怎么分配内存的？

## 什么是字节序？怎么判断是大端还是小端？有什么用？

# 场景题、算法和数据结构

## leetcode hot100 至少刷两遍，剑指 offer 至少刷两遍 重中之重！！

## 介绍熟悉的设计模式（单例，简单工厂模式）

## 写单例模式，线程安全版本

## 写三个线程交替打印 ABC

## 二维码登录的实现过程

## 不使用临时变量实现 swap 函数

## 实现一个 strcpy 函数（或者 memcpy），如果内存可能重叠呢

## 实现快排

## 实现一个堆排序

## 实现一个插入排序

## 快排存在的问题，如何优化

## 反转一个链表

## Top K 问题（可以采取的方法有哪些，各自优点？）（重点）

## 8G 的 int 型数据，计算机的内存只有 2G，怎么对它进行排序？（外部排序）

## 自己构建一棵二叉树，使用带有 null 标记的前序遍历序列

## 介绍一下 b 树和它的应用场景有哪些

## 介绍一下 b+ 树和它的应用场景有哪些

## 介绍一下红黑树和它的应用场景有哪些

## 怎么写 sql 取表的前 1000 行数据

## N 个骰子出现和为 m 的概率

## 海量数据问题（可参考左神的书）

## 一致性哈希

## 希尔排序，手撕

## Dijkstra 算法

## 如何实现一个动态数组

## 最小生成树算法说一下

## 海量数据的 bitmap 使用原理

## 布隆过滤器原理与优点

## 布隆过滤器处理大规模问题时的持久化，包括内存大小受限、磁盘换入换出问题

## 实现一个队列，并且使它支持多线程，队列有什么应用场景

# 智力题

## 100 层楼，只有 2 个鸡蛋，想要判断出那一层刚好让鸡蛋碎掉，给出策略

## 毒药问题，1000 瓶水，其中有一瓶可以无限稀释的毒药，要快速找出哪一瓶有毒，需要几只小白鼠

## 先手必胜策略问题：100 本书，每次能够拿 1-5 本，怎么拿能保证最后一次是你拿

## 放 n 只蚂蚁在一条树枝上，蚂蚁与蚂蚁之间碰到就各自往反方向走，问总距离或者时间

## 瓶子换饮料问题：1000 瓶饮料，3 个空瓶子能够换 1 瓶饮料，问最多能喝几瓶

## 在 24 小时里面时针分针秒针可以重合几次

## 有一个天平，九个砝码，一个轻一些，用天平至少几次能找到轻的？

## 有十组砝码每组十个，每个砝码重 10g，其中一组每个只有 9g，有能显示克数的秤最少几次能找到轻的那一组砝码？

## 生成随机数问题：给定生成 1 到 5 的随机数 Rand5()，如何得到生成 1 到 7 的随机数函数 Rand7()

## 赛马：有 25 匹马，每场比赛只能赛 5 匹，至少要赛多少场才能找到最快的 3 匹马？

## 烧 香 / 绳子 / 其他 确定时间问题：有两根不均匀的香，燃烧完都需要一个小时，问怎么确定 15 分钟的时长？

## 掰巧克力问题 N\_M 块巧克力，每次掰一块的一行或一列，掰成 1\_1 的巧克力需要多少次？（1000 个人参加辩论赛，1V1，输了就退出，需要安排多少场比赛）

# 大数据

## 介绍一下 Hadoop

## 说一下 MapReduce 的运行机制

## 介绍一下 kafka

## 为什么 kafka 吞吐量高？介绍一下零拷贝

## 介绍一下 spark

## 介绍一下 spark-streaming

## spark 的 transformation 和 action 有什么区别

## spark 常用的算子说几个

## 如何保证 kafka 的消息不丢失

## kafka 如何选举 leader

## 说下 spark 中的宽依赖和窄依赖

## 说下 spark 中 stage 是依照什么划分的

## spark 的内存管理是怎样的

## spark 的容错机制是什么样的

# HR面

## 自我介绍

## 项目中遇到的最大难点

## 项目中的收获

## 可以实习的时间，实习时长

## 哪里人

## 说一下自己的性格

## 你的优缺点是什么

## 有什么兴趣爱好，画的怎么样 / 球打的如何 / 游戏打的怎么样

## 看过最好的一本书是什么

## 学习技术中有什么难点

## 怎么看待加班

## 觉得深圳怎么样（或者其他地点）

## 遇见过最大的挫折是什么，怎么解决的

## 职业规划

## 目前的 offer 情况

## 你最大的优势和劣势是什么

## 介绍在项目里面充当的角色

## 介绍一下本科获得的全国赛奖项的情况

## 最有成就感的事情 / 最骄傲的一件事情

## 在实验室中担任什么角色，参加的 XXX 能聊聊吗

## 用两个词来形容自己