软件测试面笔试 题库

日期：2022.04.22

《软件测试面笔试题》

提示：

1、 题目类型包括：单选题、多选题、简答题、填空题、编程题等

2、 题目前缀【类型】【知识点】

知识点目录			
LoadRunner 负载测试工具	测试类型	测试用例	测试配置
JMeter 压力测试工具	测试实战	测试概念	测试需求
测试计划	测试缺陷	测试文档	网络基础
JAVA	C 语言	Python	Linux

业务场景一：——讯飞语记 web 项目功能测试

1.【简答题】【讯飞语记】介绍下这个项目的测试流程

拿到这个项目之后，首先是根据需求文档，对需求进行分析，属性功能模块和逻辑，在需求文档上标准有疑问的地方，需求测试完成之后，和老师进行汇报需求问题，最终确定需求。打开讯飞语记项目，根据需求编写测试计划，明确测试策略、测试方法和测试进度，然后编写测试用例，采用黑盒测试方法，测试用例编写完成之后又老师进行评审，根据评审意见进行优化。接着进行测试用例的执行，并标注 bug，编写缺陷报告，最后，根据缺陷报告编写测试报告。

2.【简答题】【讯飞语记】你认为在这个项目测试过程中，哪一项工作最重要？

我认为测试工作中，每一项工作都比较重要，在需求分析阶段，要搞清楚需求，如果需求出现错误，会造成后续返工，导致整个项目延期；测试计划阶段要确定测试范围，明确测试方法，合理安排测试时间和进度，在测试设计阶段，要根据需求编写测试用例，并经过几轮评审通过后才能进行测试执行，在测试执行阶段，要能够快速定位 bug，对 bug 定位和描述要精准，在测试评估阶段，要正确对测试项目进行客观公正评价。

3.【简答题】【讯飞语记】你在这个项目中，编写了多少条用例，你编写用例采用了哪些方法。

我总共写了 400 条用例，主要包括冒烟测试、功能测试，总共写了 8 个模块的功能用例，我主要采用的黑盒测试方法，包括等价类边界值，因果图判定表，场景法，错误推断法四种黑盒设计方法。

4.【简答题】【讯飞语记】这个项目的冒烟测试是指什么，采用什么方法进行编写的？

冒烟测试也叫基本流程测试，主要测试登录、新建一条语记，能够保存成功，点击退出能够正常退出。冒烟测试必须成功后，开发才能发布版本，如果冒烟失败，需要开发重新构建版本。在编写冒烟测试用例的时候，我使用的是场景法，先确定基本流，选择登录、新建、退出的正向用例，验证是否成功。

5.【简答题】【讯飞语记】这个项目你发现几个 bug,举几个典型的 bug 讲一下？

我总共发现了 5 个 bug，有功能性的和易用性 bug，功能性的 bug，有一条，第三方微信扫描登录，没有设置时效性，二维码不会过期，安全性太差。还有一条易用性 bug,在设置分类界面，字体和其他页面偏大，整体不统一。

6.【简答题】【讯飞语记】你在这个项目中，除了编写测试用例外，还写了哪些文档，主要包含哪些内容？

除了测试用例，我还编写了测试报告，在我们小组测试用例执行完毕之后，我编写了测试报告，测试报告主要包含，缺陷分析，通过缺陷分析得出版本的质量，测试计划执行情况、评估测试进度的偏差，还有总体评价，通过图表数据得到整体的项目评价，并给出测试的专业建议。

7.【简答题】【讯飞语记】你测试用例写好后是直接进行执行吗？有没有进行用例评审，从哪些维度进行评审的？

我测试用例写好后，先在自己小组内评审，合并成整体测试用例，然后老师让我们组间进行交叉评审，其他小组给我们提了很多有效的建议，我们修改好之后，老师又进行的评审。评审的维度很多，主要包含：1、测试用例是否包含预期结果，预期结果是否来自需求 2、测试用例是否覆盖所有功能模块，是否有遗漏；3 测试用例包含的要素是否齐全，有没有优先级；4 优先级设定是否合理，有没有全是高或全是中的；5、测试用例是否可执行，操作步骤是否清晰 6、测试标题描述是否简洁明了，是否有二义性；

8.【简答题】【讯飞语记】你在编写测试用例的时候，怎么把黑盒测试方法融入这个项目中的。

首先先使用场景法，梳理每个功能模块的流程，确定基本流和备选流，备选流设置异常数据测试用例，基本流程是正向用例。这个项目中的功能模块遇到文本校验和表单校验的时候，采用等价类和边界方法；在查询模块，因为存在多条件组合情况，使用的判定表；在编写兼容性测试用例时，使用的正交法，生成兼容性正交矩阵表；在测试用例编写完成后，使用了错误推断法进行用例的补充，使用例的覆盖率更广。

9.【简答题】【讯飞语记】你发现的缺陷是在哪里进行记录的，怎么进行缺陷的管理。

首先在测试用例执行的过程中，根据预期结果填写实际结果，遇到实际结果和预期结果不一致的进行标红，此处就是一个 bug，在 bug 标注完成之后，开始编写缺陷报告，可以在缺陷管理平台上记录，我使用的是 delbug 平台，要对缺陷进行描述，包括标题、重现步骤、优先级、严重程度、所属模块等，缺陷记录之后可以直接制定开发进行修复，开发修复完成之后，需要再新版本进行验证，通过后关闭缺陷，不通过继续打开。

10.【简答题】【讯飞语记】这个项目的兼容性你是怎么设计测试用例的，使用的什么方法？

根据需求文档，讯飞语记这个项目，需要从主流浏览器、操作系统、场景输入法、常用杀毒软件等要素去考虑，如果全部覆盖测试用例较多，而且场景不容易搭建，所有采用正交设计法，选取主流的组合，在项目组内选取不同同学的电脑环境测试。

业务场景二：——PHP 论坛项目自动化测试

1.【简答题】【PHP 论坛】这个项目自动化测试的流程是什么？你使用的是哪个框架》

这个项目是搭建本地的项目，我们测试的流程是，第一步环境搭建，第二步属性项目功能，梳理出可以实现自动化测试的模块，编写自动化测试用例，第三步根据用例开发自动化脚本，第四步运行脚本，测试执行，记录缺陷，最后一步是编写测试报告。使用的是 pyhton+selenium+unittest 框架

2.【简答题】【PHP 论坛】这个项目使用了哪些元素定位的方法，举例子说明？

Selenium 元素定位有 8 大方法，可以通过，id 定位、name 定位、class_name 定位、tag 定位、link 定位、xpath 定位、CSS 定位。在这个项目中我用的最多的是 ID 定位方法，也是优先使用的，在登录定位界面，用户名、密码和登录按钮，本身都有 ID 属性，直接通过 id 属性直接定位。在发帖界面有个内嵌窗口，使用是 frame 切换窗体的

功能。

3.【简答题】【PHP 论坛】这个项目在编写自动化脚本的时候，使用了哪些函数？请举例子说明。

Selenium 内置函数很多，在登录界面，主要先进行元素定位，在用户名密码文本框使用 sendkey 函数进行文本输入，使用 clik 函数进行登录按钮点击。在发帖界面，使用 python 语言编写随机函数，保证发布的帖子内容不重复，在上传头像界面，使用 sendkey 函数后面添加路径的方式，进行图片的上传。

4.【简答题】【PHP 论坛】这个项目有没有使用 OP 模式，谈一下你对这个模式的理解？

页面对象模型(PO)是一种设计模式,用来管理维护一组页面元素的对象库.，在 PO 下,应用程序的每一个页面都有一个对应的 Page 类.，每一个 Page 类维护着该页面的元素集和操作这些元素的方法.。他的优势有：PO 提供了一种业务流程与页面元素操作分离的模式，这使得测试代码变得更加清晰（代码可读性强），可复用的页面方法代码会变得更加优化（可维护性高），页面对象与用例分离，使得我们更好的复用对象（复用性高）。

5.【简答题】【PHP 论坛】这个项目的注册模块自动化你是怎么设计脚本的，注册的数据怎么储存的？

注册功能用例设计，主要针对用户名、密码、邮箱等 5 个文本框进行格式校验，采用等价类边界值的方法设计用例，在进行自动化脚本设计的时候，使用 csv 文件储存，数据驱动的方式，导入 CSV 包，通过文件打开读取，按行读取，把数据按顺序提取出来，迭代运行，可以实现，每次注册从 csv 中读取一组数据进行测试。

业务场景三：——WebTours 飞机订票系统性能测试

1.【简答题】【WebTours 飞机订票系统】你拿到这个项目是怎么开展性能测试的？

这个项目是安装在本机的，安装项目后先运行熟悉订票的功能，再查看需求文档，找到需求文档里面的性能需求，其中有一条需求是 20 个用户登录并发，响应时间不超过 5 秒，成功率 99%，cup 使用不超过 80%。了解需求后，制定性能测试计划，编写性能测试方案，选择性能测试工具 loadrunner。截止设计脚本，使用 loadrunner 录制登录脚本，对脚本进行增强操作，然后到场景设计控件中，设置 20 个 vuser，设置集合点，运行监控数据，到第三个控件生成测试报告，查看测试结果。

2.【简答题】【WebTours 飞机订票系统】针对这个项目，你使用了 2 款测试工具，请对比下 2 款工具的区别。

Loadrunner 和 jmeter 都可以做性能测试，其中 Jmeter 是开源工具，比较小巧，loadrunner 包含三个安装组件，相比较大，他俩的工作原理不一样，laodruner 是通过第一个组件录制脚本，模拟浏览器发送请求，脚本完善后到第二个组件进行场景的设计和运行，运行结束后到第三个组件查看生成的测试报告。Jmeter 是通过 badboy 工具录制脚本，然后导入 jmeter,通过设置线程组和场景插件进行性能测试。

3.【简答题】【WebTours 飞机订票系统】针对这个项目，你使用了 2 款测试工具，请对比下 2 款工具的区别。

Loadrunner 和 jmeter 都可以做性能测试，其中 Jmeter 是开源工具，比较小巧，loadrunner 包含三个安装组件，相比较大，他俩的工作原理不一样，laodruner 是通过第一个组件录制脚本，模拟浏览器发送请求，脚本完善后到第二个组件进行场景的设计和运行，运行结束后到第三个组件查看生成的测试报告。Jmeter 是通过 badboy 工具录制脚本，然后导入 jmeter,通过设置线程组和场景插件进行性能测试。

4.【简答题】【WebTours 飞机订票系统】性能测试包括哪些，这个项目你做了那些性能测试，举例子说明

性能测试包含并发测试、稳定性测试、基准测试、大数据量测试、负载测试、压力测试等，其中在这个项目中我主要做了负载和并发测试。负载测试主要测试在订票业务流程中，每隔 30 秒增加 5 个用户订票操作，在场景设计中，通过中途添加 vuser 的方式实现。并发测试，主要通过设置集合点实现。

5.【简答题】【WebTours 飞机订票系统】你这个项目做的登录并发测试，性能指标有哪些？你测试的真实数据是多少，有没有达标？

性能指标包括：响应时间、业务成功率、cup 和内存的使用率，我跑了 100 个用户并发测试，其中去除思考时间，登录平均响应时间是 1.2 秒，业务成功率是 98%，cup 和内存使用率 70%左右，都满足需求中的指标。

业务场景四：——PhpWind 论坛接口测试

1.【简答题】【PhpWind 论坛】你以这个项目为案例，如何做接口测试，讲下你开展接口测试的流程。

这个项目在做接口测试的时候，先拿到的是接口文档，里面包含十大接口的地址方法和参数配置，需求文档熟悉后，我们使用接口测试工具 postman 进行接口测试，举登录为例子，设置 post 请求方法，输入登录地址，配置 boby 参数，点击发送，查看服务端返回数据是否满足需求，然后通过修改参数，进行错误的用例执行，检查响应结果是否正确。不正确的进行 bug 的记录，测试完成后，编写测试报告。

2.【简答题】【PhpWind 论坛】这个项目的接口文档有没有用抓包工具进行校验，怎么使用的？

我们拿到接口文档之后，先使用 fiddler 对论坛 10 个接口进行抓包，通过抓包查看请求头和请求体，请求参数和服务端返回的数据，拿到这些数据后和需求文档进行对比，查看需求文档是否有遗漏和错误的地方。Fiddler 是抓包工具，直接开启，使用过滤器只抓取论坛地址，操作论坛的登录发帖等功能，fiddler 会自动抓取。

3.【简答题】【PhpWind 论坛】这个项目的接口测试，除了 postman 工具，还用什么工具可以做接口测试？

接口测试除了用 postman 工具，也可以用 jmeter 进行接口测试，方法和 potman 类似，创建线程组，添加 HTTP 取样器，设置请求地址，请求方法，请求参数，添加查看结果树，运行后，查看服务端返回数据是否正确。

4.【简答题】【PhpWind 论坛】这个项目的接口测试，用 python 怎么进行测试的？，需要注意哪些？

需要通过第三方包 requests 库去使用，这个库里面支持 HTTP 相关操作，使用 get 和 post 方法可以实现，因为 postman 和 jmeter 在做接口测试的时候有 cookie 设置器，在用 requests 库时，需要自己配置 cookies，这块需要结合 fiddler 抓包，获取 cookies 的值。

5.【简答题】【PhpWind 论坛】Jmeter 这款工具，我看你在性能测试和接口测试都用到了，做性能和接口，有哪些不同？

Jemter 做性能测试主要偏重场景的设计，需要下载第三方插件，性能测试的时候，先通过 badboy 录制脚本，这个脚本导出 jemter 格式，导入 jmeter,然后设置性能场景，并发、稳定性等 sessionid 管理，查看测试结果数据是否满足性能指标。接口测试的时候，根据接口需求文档，手动输入 url ,选择请求方式（get、post），配置参数或者信息体，点击发送，查看服务端响应 是否符合需求

- 业务场景五-web 应用测试

1.【简答题】【web 应用测试】先介绍下你做的 web 项目

介绍 web 项目的构成模块，自己负责的模块，相应的功能点介绍，大概介绍使用到的工具、框架、语言、测试类型、测试方式等内容

2.【简答题】【web 应用测试】针对 web 项目做了什么类型的测试

1、功能测试 2、性能测试 3、压力测试 4、负载测试 5、易用性测试 6、安装测试 7、界面测试 8、配置测试 9、文档测试 10、兼容性测试 11、安全性测试 12、恢复测试

3.【简答题】【web 应用测试】简要说明 web 项目的功能测试

1、 需求评审
2、 分析需求，整理测试点，画出简要流程图
3、 按模块测试点编写测试用例
4、 用例评审
5、 执行用例，并根据实际情况优化用例
6、 回归 bug，进行后面的测试
7、 整理测试报告

4.【简答题】【web 应用测试】web 项目测试时间需要多久？

按开发工作量，初步估计测试时间，预估时间是开发实践的 50%~100%，具体根据开发质量实际情况，尽量在计划之内完成测试，如果 bug 太多，可能会延长发布时间

5.【简答题】【web 应用测试】web 项目需求评审内容

1、 明确整个项目的目标是什么
2、 项目的各个模块功能分工
3、 明确每个模块所需要实现的目标
4、 明确每个模块下所有的功能点

6.【简答题】【web 应用测试】web 项目测试用例评审内容

1、 用例模板是否符合规范
2、 用例是否包含了所有功能模块
3、 每个模块的用例范围是否包含完整
4、 与关联模块的相关测试是否涉及到

7.【简答题】【web 应用测试】web 项目测试 bug 的处理流程

1、 提出 bug，指派给对应开发
2、 实时跟踪 bug 状态，等待开发解决
3、 解决后，发布新包到测试环境，回归 bug
4、 如果问题仍然存在，重新打开 bug，指向开发
5、 如果理解有冲突，问题上升到测试经理或者项目经理
6、 如果已解决，关闭 bug

面试/笔试题

1.【简答题】【测试类型】说说你对集成测试中自顶向下集成和自底向上集成两个策略的理解，要谈出它们各自的优缺点和主要适应于哪种类型测试;

自顶向下集成

优点：较早地验证了主要控制和判断点;按深度优先可以首先实现和验证一个完整的软件功能;功能较早证实，带来信心;只需一个驱动，减少驱动器开发的费用;支持故障隔离。

缺点：柱的开发量大;底层验证被推迟;底层组件测试不充分。

适应于产品控制结构比较清晰和稳定;高层接口变化较小;底层接口未定义或经常可能被修改;产口控制组件具有较大的技术风险，需要尽早被验证;希望尽早能看到产品的系统功能行为。

2、自底向上集成

优点：对底层组件行为较早验证;工作最初可以并行集成，比自顶向下效率高;减少了桩的工作量；支持故障隔离。

缺点：驱动的开发工作量大;对高层的验证被推迟，设计上的错误不能被及时发现。

适应于底层接口比较稳定;高层接口变化比较频繁;底层组件较早被完成。

2.【简答题】【测试类型】您所熟悉的软件测试类型都有哪些?请试着分别比较这些不同的测试类型的区别与联系(如功能测试、性能测试……)

测试类型有：功能测试，性能测试，界面测试。

功能测试在测试工作中占的比例最大，功能测试也叫黑盒测试。是把测试对象看作一个黑盒子。利用黑盒测试法进行动态测试时，需要测试软件产品的功能，不需测试软件产品的内部结构和处理过程。采用黑盒技术设计测试用例的方法有：等价类划分、边界值分析、错误推测、因果图和综合策略。

性能测试是通过自动化的测试工具模拟多种正常、峰值以及异常负载条件来对系统的各项性能指标进行测试。负载测试和压力测试都属于性能测试，两者可以结合进行。通过负载测试，确定在各种工作负载下系统的性能，目标是测试当负载逐渐增加时，系统各项性能指标的变化情况。压力测试是通过确定一个系统的瓶颈或者不能接收的性能点，来获得系统能提供的最大服务级别的测试。

界面测试，界面是软件与用户交互的最直接的层，界面的好坏决定用户对软件的第一印象。而且设计良好的界面能够引导用户自己完成相应的操作，起到向导的作用。同时界面如同人的面孔，具有吸引用户的直接优势。设计合理的界面能给用户带来轻松愉悦的感受和成功的感觉，相反由于界面设计的失败，让用户有挫败感，再实用强大的功能都可能在用户的畏惧与放弃中付诸东流。

区别在于，功能测试关注产品的所有功能上，要考虑到每个细节功能，每个可能存在的功能问题。性能测试主要关注于产品整体的多用户并发下的稳定性和健壮性。界面测试更关注于用户体验上，用户使用该产品的时候是否易用，是否易懂，是否规范(快捷键之类的)，是否美观(能否吸引用户的注意力)，是否安全(尽量在前台避免用户无意输入无效的数据，当然考虑到体验性，不能太粗鲁的弹出警告)?做某个性能测试的时候，首先它可能是个功能点，首先要保证它的功能是没问题的，然后再考虑该功能点的性能测试

3.【简答题】【测试类型】请试着比较一下黑盒测试、白盒测试、单元测试、集成测试、系统测试、验收测试的区别与联系。

黑盒测试：已知产品的功能设计规格，可以进行测试证明每个实现了的功能是否符合要求。

白盒测试：已知产品的内部工作过程，可以通过测试证明每种内部操作是否符合设计规格要求，所有内部成分是否以经过检查。

软件的黑盒测试意味着测试要在软件的接口处进行。这种方法是把测试对象看做一个黑盒子，测试人员完全不考虑程序内部的逻辑结构和内部特性，只依据程序的需求规格说明书，检查程序的功能是否符合它的功能说明。因此黑盒测试又叫功能测试或数据驱动测试。黑盒测试主要是为了发现以下几类错误：

1、是否有不正确或遗漏的功能?2、在接口上，输入是否能正确的接受?能否输出正确的结果?3、是否有数据结

构错误或外部信息(例如数据文件)访问错误?4、性能上是否能够满足要求?5、是否有初始化或终止性错误?

软件的白盒测试是对软件的过程性细节做细致的检查。这种方法是把测试对象看做一个打开的盒子，它允许测试人员利用程序内部的逻辑结构及有关信息，设计或选择测试用例，对程序所有逻辑路径进行测试。通过在不同点检查程序状态，确定实际状态是否与预期的状态一致。因此白盒测试又称为结构测试或逻辑驱动测试。白盒测试主要是想对程序模块进行如下检查：

1、对程序模块的所有独立的执行路径至少测试一遍。

2、对所有的逻辑判定，取“真”与取“假”的两种情况都能至少测一遍。

3、在循环的边界和运行的界限内执行循环体。

4、测试内部数据结构的有效性，等等。

单元测试(模块测试)是开发者编写的一小段代码，用于检验被测代码的一个很小的、很明确的功能是否正确。通常而言，一个单元测试是用于判断某个特定条件(或者场景)下某个特定函数的行为。

单元测试是由程序员自己来完成，最终受益的也是程序员自己。可以这么说，程序员有责任编写功能代码，同时也就有责任为自己的代码编写单元测试。执行单元测试，就是为了证明这段代码的行为和我们期望的一致。

集成测试(也叫组装测试，联合测试)是单元测试的逻辑扩展。它的最简单的形式是：两个已经测试过的单元组合成一个组件，并且测试它们之间的接口。从这一层意义上讲，组件是指多个单元的集成聚合。在现实方案中，许多单元组合成组件，而这些组件又聚合成程序的更大部分。方法是测试片段的组合，并最终扩展进程，将您的模块与其他组的模块一起测试。最后，将构成进程的所有模块一起测试。

系统测试是将经过测试的子系统装配成一个完整系统来测试。它是检验系统是否确实能提供系统方案说明书中指定功能的有效方法。(常见的联调测试)

系统测试的目的是对最终软件系统进行全面的测试，确保最终软件系统满足产品需求并且遵循系统设计。

验收测试是部署软件之前的最后一个测试操作。验收测试的目的是确保软件准备就绪，并且可以让最终用户将其用于执行软件的既定功能和任务。

验收测试是向未来的用户表明系统能够像预定要求那样工作。经集成测试后，已经按照设计把所有的模块组装成一个完整的软件系统，接口错误也已经基本排除了，接着就应该进一步验证软件的有效性，这就是验收测试的任务，即软件的功能性能如同用户所合理期待的那样。

4.【简答题】【测试类型】请你分别介绍一下单元测试、集成测试、系统测试、验收测试、回归测试

1、单元测试：完成最小的软件设计单元（模块）的验证工作，目标是确保模块被正确的编码，使用过程设计描述作为指南，对重要的控制路径进行测试以发现模块内的错误，通常情况下是白盒的，对代码风格和规则、程序设计和结构、业务逻辑等进行静态测试，及早的发现和解决不易显现的错误。

2、集成测试：通过测试发现与模块接口有关的问题。目标是把通过了单元测试的模块拿来，构造一个在设计中所描述的程序结构，应当避免一次性的集成（除非软件规模很小），而采用增量集成。

自顶向下集成：模块集成的顺序是首先集成主模块，然后按照控制层次结构向下进行集成，隶属于主模块的模块按照深度优先或广度优先的方式集成到整个结构中去。

自底向上集成：从原子模块开始来进行构造和测试，因为模块是自底向上集成的，进行时要求所有隶属于某个给顶层次的模块总是存在的，也不再有使用稳定测试桩的必要。

3、系统测试：是基于系统整体需求说明书的黑盒类测试，应覆盖系统所有联合的部件。系统测试是针对整个产品系统进行的测试，目的是验证系统是否满足了需求规格的定义，找出与需求规格不相符合或与之矛盾的地方。系统测试的对象不仅仅包括需要测试的产品系统的软件，还要包含软件所依赖的硬件、外设甚至包括某些数据、某些支持软件及其接口等。因此，必须将系统中的软件与各种依赖的资源结合起来，在系统实际运行环境下来进行测试。

4、回归测试：回归测试是指在发生修改之后重新测试先前的测试用例以保证修改的正确性。理论上，软件产生新版本，都需要进行回归测试，验证以前发现和修复的错误是否在新软件版本上再次出现。根据修复好了的缺陷再重新进行测试。回归测试的目的在于验证以前出现过但已经修复好的缺陷不再重新出现。一般指对某已知修正的缺陷再次围绕它原来出现时的步骤重新测试。

5、验收测试：验收测试是指系统开发生命周期方法论的一个阶段，这时相关的用户或独立测试人员根据测试计划和结果对系统进行测试和接收。它让系统用户决定是否接收系统。它是一项确定产品是否能够满足合同或用户所规定需求的测试。验收测试包括 Alpha 测试和 Beta 测试。

Alpha 测试：是由用户在开发者的场所来进行的，在一个受控的环境中进行。

Beta 测试：由软件的最终用户在一个或多个用户场所来进行的，开发者通常不在现场，用户记录测试中遇到的问题并报告给开发者，开发者对系统进行最后的修改，并开始准备发布最终的软件。

5.【简答题】【测试类型】请你回答一下单元测试、集成测试、系统测试、验收测试、回归测试这几步中最重要的是哪一步

这些步骤都重要，看公司需求，从这几个步骤里面选一个最重要的，我认为是系统测试，因为此时单元测试和集成测试已完成，能够对软件所有功能进行功能测试，能够覆盖系统所有联合的部件，是针对整个产品系统进行的测试，能够验证系统是否满足了需求规格的定义，因此我认为系统测试很重要。

6.【简答题】【测试类型】请回答集成测试和系统测试的区别，以及它们的应用场景主要是什么？

区别：

1、计划和用例编制的先后顺序：从 V 模型来讲，在需求阶段就要制定系统测试计划和用例，HLD 的时候做集成测试计划和用例，有些公司的具体实践不一样，但是顺序肯定是先做系统测试计划用例，再做集成。

2、用例的粒度：系统测试用例相对很接近用户接受测试用例，集成测试用例比系统测试用例更详细，而且对于接口部分要重点写，毕竟要集成各个模块或者子系统。

3、执行测试的顺序：先执行集成测试，待集成测试出的问题修复之后，再做系统测试。

应用场景：

集成测试：完成单元测试后，各模块联调测试；集中在各模块的接口是否一致、各模块间的数据流和控制流是

否按照设计实现其功能、以及结果的正确性验证等等；可以是整个产品的集成测试，也可以是大模块的集成测试；集成测试主要是针对程序内部结构进行测试，特别是对程序之间的接口进行测试。集成测试对测试人员的编写脚本能力要求比较高。测试方法一般选用黑盒测试和白盒测试相结合。

系统测试：针对整个产品的全面测试，既包含各模块的验证性测试（验证前两个阶段测试的正确性）和功能性（产品提交个用户的功能）测试，又包括对整个产品的健壮性、安全性、可维护性及各种性能参数的测试。系统测试测试软件《需求规格说明书》中提到的功能是否有遗漏，是否正确的实现。做系统测试要严格按照《需求规格说明书》，以它为标准。测试方法一般都使用黑盒测试法。

7.【简答题】【测试类型】请说一说黑盒与白盒的测试方法

黑盒测试：

黑盒测试也称功能测试或数据驱动测试，它是在已知产品所应具有的功能，通过测试来检测每个功能是否都能正常使用，在测试时，把程序看作一个不能打开的黑盆子，在完全不考虑程序内部结构和内部特性的情况下，测试者在程序接口进行测试，它只检查程序功能是否按照需求规格说明书的规定正常使用，程序是否能适当地接收输入数锯而产生正确的输出信息，并且保持外部信息（如数据库或文件）的完整性。

“黑盒”法着眼于程序外部结构、不考虑内部逻辑结构、针对软件界面和软件功能进行测试。“黑盒”法是穷举输入测试，只有把所有可能的输入都作为测试情况使用，才能以这种方法查出程序中所有的错误。实际上测试情况有无穷多个，因此不仅要测试所有合法的输入，而且还要对那些不合法但是可能的输入进行测试。

常用的黑盒测试方法有：等价类划分法；边界值分析法；因果图法；场景法；正交实验设计法；判定表驱动分析法；错误推测法；功能图分析法。

白盒测试：

白盒测试也称为结构测试或逻辑驱动测试，是针对被测单元内部是如何进行工作的测试。它根据程序的控制结构设计测试用例，主要用于软件或程序验证。白盒测试法检查程序内部逻辑结构，对所有的逻辑路径进行测试，是一种穷举路径的测试方法，但即使每条路径都测试过了，但仍然有可能存在错误。因为：穷举路径测试无法检查出程序本身是否违反了设计规范，即程序是否是一个错误的程序；穷举路径测试不可能检查出程序因为遗漏路径而出错；穷举路径测试发现不了一些与数据相关的错误。

白盒测试需要遵循的原则有：1. 保证一个模块中的所有独立路径至少被测试一次；2. 所有逻辑值均需要测试真（true）和假（false）；两种情况；3. 检查程序的内部数据结构，保证其结构的有效性；4. 在上下边界及可操作范围内运行所有循环。

常用白盒测试方法：

静态测试：不用运行程序的测试，包括代码检查、静态结构分析、代码质量度量、文档测试等等，它可以由人工进行，充分发挥人的逻辑思维优势，也可以借助软件工具（Fxcop）自动进行。

动态测试：需要执行代码，通过运行程序找到问题，包括功能确认与接口测试、覆盖率分析、性能分析、内存分析等。

白盒测试中的逻辑覆盖包括语句覆盖、判定覆盖、条件覆盖、判定/条件覆盖、条件组合覆盖和路径覆盖。六种覆盖标准发现错误的能力呈由弱到强的变化：

1.语句覆盖每条语句至少执行一次。

2.判定覆盖每个判定的每个分支至少执行一次。

3.条件覆盖每个判定的每个条件应取到各种可能的值。

4.判定/条件覆盖同时满足判定覆盖条件覆盖。

5.条件组合覆盖每个判定中各条件的每一种组合至少出现一次。

6.路径覆盖使程序中每一条可能的路径至少执行一次。

8.【简答题】【测试类型】您所熟悉的软件测试类型都有哪些?请试着分别比较这些不同的测试类型的区别与联系(如功能测试、性能测试……)

易用性测试-界面的友好性，操作方便性等。

功能测试-系统中功能性需求的满足

安全性测试-系统是否存在安全隐患和漏洞

性能测试-系统在大并发下的响应速度和健壮性

9.【简答题】【测试类型】请说一下手动测试与自动化测试的优缺点

手工测试缺点：

1、重复的手工回归测试，代价昂贵、容易出错。

2、依赖于软件测试人员的能力。

手工测试优点：

1、测试人员具有经验和对错误的猜测能力。

2、测试人员具有审美能力和心理体验。

3、测试人员具有是非判断和逻辑推理能力。

自动化测试的优点：

1、对程序的回归测试更方便。这可能是自动化测试最主要的任务，特别是在程序修改比较频繁时，效果是非常明显的。由于回归测试的动作和用例是完全设计好的，测试期望的结果也是完全可以预料的，将回归测试自动运行，可以极大提高测试效率，缩短回归测试时间。

2、可以运行更多更繁琐的测试。自动化的一个明显的好处是可以在较少的时间内运行更多的测试。

3、可以执行一些手工测试困难或不可能进行的测试。比如，对于大量用户的测试，不可能同时让足够多的测试人员同时进行测试，但是却可以通过自动化测试模拟同时有许多用户，从而达到测试的目的。

4、更好地利用资源。将繁琐的任务自动化，可以提高准确性和测试人员的积极性，将测试技术人员解脱出来投入更多精力设计更好的测试用例。有些测试不适合于自动测试，仅适合于手工测试，将可自动测试的测试自动化后，可以让测试人员专注于手工测试部分，提高手工测试的效率。

5、测试具有一致性和可重复性。由于测试是自动执行的，每次测试的结果和执行的内容的一致性是可以得到保障的，从而达到测试的可重复的效果。

6、测试的复用性。由于自动测试通常采用脚本技术，这样就有可能只需要做少量的甚至不做修改，实现在不同的测试过程中使用相同的用例。

7、增加软件信任度。由于测试是自动执行的，所以不存在执行过程中的疏忽和错误，完全取决于测试的设计质量。一旦软件通过了强有力的自动测试后，软件的信任度自然会增加。

自动化测试的缺点：

1、不能取代手工测试

2、手工测试比自动测试发现的缺陷更多

3、对测试质量的依赖性极大

4、测试自动化不能提高有效性

5、测试自动化可能会制约软件开发。由于自动测试比手动测试更脆弱，所以维护会受到限制，从而制约软件的开发。

6、工具本身并无想像力

10.【简答题】【测试类型】软件验收测试包括

正式验收测试、alpha 测试、beta 测试三种测试。

11.【简答题】【测试类型】请说出这些测试最好由那些人员完成，测试的是什么？

代码、函数级测试一般由白盒测试人员完成，他们针对每段代码或函数进行正确性检验，检查其是否正确的实现了规定的功能。

模块、组件级测试主要依据是程序结构设计测试模块间的集成和调用关系，一般由测试人员完成。

系统测试在于模块测试与单元测试的基础上进行测试。了解系统功能与性能，根据测试用例进行全面的测试。

12.【简答题】【测试类型】什么是回归测试?

回归测试: (regression testing): 回归测试有两类：用例回归和错误回归;用例回归是过一段时间以后再回头对以前使用过的用例在重新进行测试，看看会重新发现问题。错误回归，就是在新版本中，对以前版本中出现并修复的缺陷进行再次验证，并以缺陷为核心，对相关修改的部分进行测试的方法。

13.【简答题】【测试类型】单元测试、集成测试、系统测试的侧重点是什么？

单元测试针对的是软件设计的最小单元--程序模块(面向过程中是函数、过程;面向对象中是类。),进行正确性检验的测试工作,在于发现每个程序模块内部可能存在的差错.一般有两个步骤:人工静态检查\动态执行跟踪

集成测试针对的是通过了单元测试的各个模块所集成起来的组件进行检验,其主要内容是各个单元模块之间的接口,以及各个模块集成后所实现的功能.

系统测试针对的是集成好的软件系统，作为整个计算机系统的一个元素,与计算机硬件\外设\某些支持软件\数据

和人员等其他系统元素结合在一起,要在实际的运行环境中,对计算机系统进行一系列的集成测试和确认测试.

14.【简答题】【测试类型】你所了解的的软件测试类型都有哪些，简单介绍一下。

按测试策略分类：1、静态与动态测试 2、黑盒与白盒测试 3、手工和自动测试 4、冒烟测试 5、回归测试;

按测试阶段分类：单元测试、集成测试、系统测试;

其他常见测试方法：1、功能测试 2、性能测试 3、压力测试 4、负载测试 5、易用性测试 6、安装测试 7、界面测试 8、配置测试 9、文档测试 10、兼容性测试 11、安全性测试 12、恢复测试

15.【简答题】【测试类型】什么是兼容性测试？

兼容性测试是检查软件在不同软件平台，硬件平台上是否可以正常运行的测试。主要查看软件在不同操作系统、浏览器、数据库中是否运行正常。

16.【简答题】【测试概念】你觉得软件测试的核心竞争力是什么

测试人员的核心竞争力在于提早发现问题，并能够发现别人无法发现的问题。

1、早发现问题：问题发现的越早，解决的成本越低。如果一个需求在还未实现的时候就能发现需求的漏洞，那么这种问题的价值是最高的。

2、发现别人无法发现的问题：所有人都能发现的问题，你发现了，那就证明你是可以被替代的。别人发现不了，而你可以发现，那么你就是无法被替代。

17.【简答题】【测试概念】请问你在项目中关于功能测试和接口测试是怎么做的

功能测试：

首先制定测试计划，然后进行测试设计，将在测试计划阶段指定的测试活动分解，进而细化，为若干个可执行程序的子测试过程，然后执行测试，按照测试计划使用测试用例对待测项目进行逐一的，详细的排查分析评估，最后对测试结果进行统计和分析，

接口测试：

什么是接口（API）

API 全称 Application Programming Interface，这里面我们其实不用去关注 AP，只需要 I 上就可以。一个 API 就是一个 Interface。我们无时不刻不在使用 interfaces。我们乘坐电梯里面的按钮是一个 interface。我们开车一个踩油门它也是一个 interface。我们计算机操作系统也是有很多的接口。(这是目前个人找到比较好理解的一段解释)

接口就是一个位于复杂系统之上并且能简化你的任务，它就像一个中间人让你不需要了解详细的所有细节。那我们今天要讲的 Web API 就是这么一类东西。像谷歌搜索系统，它提供了搜索接口，简化了你的搜索任务。再像用户登录页面，我们只需要调用我们的登录接口，我们就可以达到登录系统的目的。

现在市面上有非常多种风格的 Web API，目前最流行的是也容易访问的一种风格是 REST 或者叫 RESTful 风格

的 API。从现在开始，以下我提到的所有 API 都是指 RESTful 风格的 API。

什么是接口测试和为什么要做接口测试

接口测试是测试系统组件间接口的一种测试。接口测试主要用于检测外部系统与系统之间以及内部各个子系统之间的交互点。测试的重点是要检查数据的交换，传递和控制管理过程，以及系统间的相互逻辑依赖关系等。

现在很多系统前后端架构是分离的，从安全层面来说，只依赖前端进行限制已经完全不能满足系统的安全要求（绕过前端太容易了），需要后端同样进行控制，在这种情况下就需要从接口层面进行验证。

如今系统越来越复杂，传统的靠前端测试已经大大降低了效率，而且现在我们都推崇测试前移，希望测试能更早的介入测试，那接口测试就是一种及早介入的方式。例如传统测试，你是不是得等前后端都完成你才能进行测试，才能进行自动化代码编写。而如果是接口测试，只需要前后端定义好接口，那这时自动化就可以介入编写接口自动化测试代码，手工测试只需要后端代码完成就可以介入测试后端逻辑而不用等待前端工作完成。

接口测试的策略

接口测试也是属于功能测试，所以跟我们以往的功能测试流程并没有太大区别，测试流程依旧是：1.测试接口文档（需求文档） 2.根据接口文档编写测试用例（用例编写完全可以按照以往规则来编写，例如等价类划分，边界值等设计方法）3. 执行测试，查看不同的参数请求，接口的返回的数据是否达到预期。

18.【简答题】【测试概念】您认为做好测试计划工作的关键是什么？

了解项目或系统的业务需求
和项目经理协调好，了解项目的进度计划安排情况

19.【简答题】【测试概念】为什么选择测试这行？

它是一个新兴的行业，有发展潜力，而且很锻炼人，需要掌握更多的技能，比做开发要更全面。

20.【简答题】【测试概念】完全测试程序是可能的吗？

不可能

测试人员对程序进行测试，只能找出程序中的 bug，但是并不能保证程序是没有 bug 的。

完全的测试要花费很多的人力财力，并且测试的数据量过大，很浪费时间。测试的结果还很多，有的都是类似的，没有必要进行相同的测试。所以完全测试是不可能的。

21.【简答题】【测试概念】你为什么能够做测试这一行？

虽然我的测试技术还不是很成熟，但是我觉得我还是可以胜任软件测试这个工作的，因为做软件测试不仅是要求技术好，还有有一定的沟通能力，耐心、细心等外在因素。综合起来看我认为我是胜任这个工作的。

22.【简答题】【测试概念】你对测试最大的兴趣在哪里?为什么?

回答这个面试题，没有固定统一的答案，但可能是许多企业都会问到的。提供以下答案供考：

最大的兴趣，感觉这是一个有挑战性的工作;

测试是一个经验行业，工作越久越能感觉到做好测试的难度和乐趣

通过自己的工作，能使软件产品越来越完善，从中体会到乐趣

回答此类问题注意以下几个方面：

尽可能的切合招聘企业的技术路线来表达你的兴趣，例如该企业是数据库应用的企业，那么表示你的兴趣在数据库的测试，并且希望通过测试提升自己的数据库掌握能力。

表明你做测试的目的是为了提升能力，也是为了更好的做好测试;提升能力不是为了以后转开发或其他的，除非用人企业有这样的安排。

不要过多的表达你的兴趣在招聘企业的范畴这外。比如招聘企业是做财务软件的，可是你表现出来的是对游戏软件的兴趣;或招聘是做 JAVA 开发的，而你的兴趣是在 C 类语言程序的开发。

23.【简答题】【测试概念】你自认为测试的优势在哪里?

该面试也没有固定不变的答案，但可参考以下几点，并结合自身特点：

有韧性、有耐心、做事有条理性、喜欢面对挑战、有信心做好每一件事情、较强的沟通能力、从以前的经理处都得到了很好的评价表明我做的很好

24.【简答题】【测试基础】试述软件的概念和特点?软件复用的含义?构件包括哪些?

软件是计算机系统中与硬件相互依存的另一部分，与计算机系统操作有关的计算机程序、规程、规则，以及可能有的文件、文档及数据。

软件复用(SoftWare Reuse)是将已有软件的各种有关知识用于建立新的软件，以缩减软件开发和维护的花费。软件复用是提高软件生产力和质量的一种重要技术。早期的软件复用主要是代码级复用，被复用的知识专指程序，后来扩大到包括领域知识、开发经验、设计决定、体系结构、需求、设计、代码和文档等一切有关方面。

可以被复用的软件成分一般称作可复用构件

25.【简答题】【测试基础】什么是软件测试，软件测试的目的?

软件测试是通过人工或者自动化的操作进行还没有商业化用途的程序，查看他们的功能是否满足客户需求。

目的：在最短时间内找出尽可能多的软件缺陷。

26.【简答题】【测试基础】什么是软件测试?

为了发现程序中的错误而执行程序的过程

27.【简答题】【测试基础】测试活动中统计了哪些数据？

用例个数 工作量 bug 数量

28.【简答题】【测试基础】什么是功能测试？

功能测试是在规定的一段时间内运行软件系统的所有功能，以验证这个软件系统有无严重错误。

29.【简答题】【测试基础】请问功能测试和性能测试的区别是什么？

1）测试目的：

功能测试：检测实际软件的功能是否符合用户需求，测功能是不是全部实现，某个实现是不是有 BUG。主要为了发现以下几类错误：A、是否有不正确或遗漏的功能？B、功能实现是否满足用户需求和系统设计的隐藏需求？C、能否正确接收输入？能否正确输出结果？

性能测试：验证软件质量的三个质量特性，可靠性，正确性和效率。主要是测试产品的健壮性

2）测试方式：

功能测试按照系用例，按照系统需求说明书和测试用例，对产品的功能一步步进行测试。找出产品功能是否全部实现

性能测试：一般都使用性能工具对产品的健壮性进行评估。通过创建场景和虚拟用户模拟真实环境，进行压力测试和负载测试。

30.【简答题】【测试基础】为什么要进行交叉测试？

因为自己执行自己设计的用例，会按照设计用例的思路来执行用例，可能会忽略一些偶然或异常的情况，交叉执行可能会发现新的 BUG，当然如果用例已经写得很细，颗粒度很小吗，输入输出写得很全面交叉执行的结果都会差不多，无论谁来执行结果都是一样的。

31.【简答题】【测试基础】什么是软件测试?软件测试的目的与原则

在规定的条件下对程序进行操作，以发现程序错误，衡量软件质量，并对其是否能满足设计要求进行评估的过程。

软件测试的目的：

测试是程序的执行过程，目的在于发现错误

一个成功的测试用例在于发现至今未发现的错误

一个成功的测试是发现了至今未发现的错误的测试

确保产品完成了它所承诺或公布的功能，并且用户可以访问到的功能都有明确的书面说明。

确保产品满足性能和效率的要求

确保产品是健壮的和适应用户环境的

软件测试的原则：

测试用例中一个必须部分是对预期输出或接过进行定义

程序员应避免测试自己编写的程序

编写软件的组织不应当测试自己编写的软件

应当彻底检查每个测试的执行结果

测试用例的编写不仅应当根据有效和预料到的输入情况，而且也应当根据无效和未预料到的输入情况

检擦程序是否“未做其应该做的”仅是测试的一半，测试的另一半是检查程序是否“做了其不应该做的”

应避免测试用例用后即弃，除非软件本身就是个一次性的软件

计划测试工作时不应默许假定不会发现错误

程序某部分存在更多错误的可能性，与该部分已经发现错误的数量成正比

软件测试是一项极富创造性，极具智力的挑战性的工作

32.【简答题】【测试基础】为什么尽量不要让时间富裕的员工去做一些测试？

首先，专业的测试人员是有一定的技能和耐心对软件一步一步进行测试。如果让时间充裕的员工去测试的话，他可能心思并不在测试上面。会很随意的、没有目标的进行测试。这样子的话测试并不完整，有的时候甚至很重要的 bug 都没法找出。所以还是需要专业的测试人员来进行测试的。

33.【简答题】【测试基础】软件测试项目从什么时候开始为什么？

一般软件测试越早展开越好，一般是从需要阶段就要进行软件测试。软件测试不仅是测试功能，对于需求文档一类的也要进行测试。越早的找出 bug，就会减少后续开发人员修改程序的次数，并且可以降低成本，如果等整个软件开发的差不多了发现一个致命的错误的话，是需要花费很多时间和人力来重新修改的。如果在一开始就发现的话就不会出现这种情况了。

34.【简答题】【测试基础】简述你在以前的工作中做过哪些事情，比较熟悉什么。参考答案如下。

我过去的主要工作是系统测试和自动化测试。在系统测试中，主要是对 BOSS 系统的业务逻辑功能，以及软交换系统的 Class 5 特性进行测试。性能测试中，主要是进行的压力测试，在各个不同数量请求的情况下，获取系统响应时间以及系统资源消耗情况。自动化测试主要是通过自己写脚本以及一些第三方工具的结合来测试软交换的特性测试。

在测试中，我感觉对用户需求的完全准确的理解非常重要。另外，就是对 BUG 的管理，要以需求为依据，并不是所有 BUG 均需要修改。

测试工作需要耐心和细致，因为在新版本中，虽然多数原来发现的 BUG 得到了修复，但原来正确的功能也可

能变得不正确。因此要注重迭代测试和回归测试。

35.【简答题】【测试缺陷】简述 bug 的生命周期?

1， 有效地记录 BUG 2， 使用 BUG 模板 3， 评价 BUG 优先级和严重性 4， BUG 的生命 5， 维护 BUG 数据库

36.【简答题】【测试缺陷】缺陷记录应包含的内容?

缺陷标识、缺陷类型、缺陷严重程度、缺陷产生可能性、缺陷优先级、缺陷状态、缺陷起源、缺陷来源、缺陷原因。

37.【简答题】【测试缺陷】如果广东用户头条 app 刷不出东西了，你应该怎么排查问题

1、检查网络连接是否稳定，更换网络尝试

2、更新头条版本尝试

3、清除 app 缓存，应用数据

38.【简答题】【测试缺陷】在您以往的工作中，一条软件缺陷(或者叫 Bug)记录都包含了哪些内容?如何提交高质量的软件缺陷(Bug)记录?

一条 Bug 记录最基本应包含：

bug 编号;bug 严重级别，优先级;bug 产生的模块;首先要有 bug 摘要，阐述 bug 大体的内容;bug 对应的版本;bug 详细现象描述，包括一些截图、录像....等等;bug 出现时的测试环境，产生的条件即对应操作步骤;高质量的 Bug 记录：

1) 通用 UI 要统一、准确缺陷报告的 UI 要与测试的软件 UI 保持一致，便于查找定位。

2) 尽量使用业界惯用的表达术语和表达方法使用业界惯用的表达术语和表达方法，保证表达准确，体现专业化。

3) 每条缺陷报告只包括一个缺陷每条缺陷报告只包括一个缺陷，可以使缺陷修正者迅速定位一个缺陷，集中精力每次只修正一个缺陷。校验者每次只校验一个缺陷是否已经正确修正。

4) 不可重现的缺陷也要报告首先缺陷报告必须展示重现缺陷的能力。不可重现的缺陷要尽力重现，若尽力之后仍不能重现，仍然要报告此缺陷，但在报告中要注明无法再现，缺陷出现的频率。

5) 明确指明缺陷类型根据缺陷的现象，总结判断缺陷的类型。例如，即功能缺陷、界面缺陷、数据缺陷，合理化建议这是最常见的缺陷或缺陷类型，其他形式的缺陷或缺陷也从属于其中某种形式。

6) 明确指明缺陷严重等级和优先等级时刻明确严重等级和优先等级之间的差别。高严重问题可能不值得解决，小装饰性问题可能被当作高优先级。

7) 描述 (Description) ，简洁、准确，完整，揭示缺陷实质，记录缺陷或缺陷出现的位置描述要准确反映缺陷的本质内容，简短明了。为了便于在软件缺陷管理数据库中寻找制定的测试缺陷，包含缺陷发生时的用户界面(UI)是个良好的习惯。例如记录对话框的标题、菜单、按钮等控件的名称。

8) 短行之间使用自动数字序号，使用相同的字体、字号、行间距短行之间使用自动数字序号，使用相同的字体、字号、行间距，可以保证各条记录格式一致，做到规范专业。

9) 每一个步骤尽量只记录一个操作保证简洁、条理井然，容易重复操作步骤。

10) 确认步骤完整，准确，简短保证快速准确的重复缺陷，“完整”即没有缺漏，“准确”即步骤正确，“简短”即没有多余的步骤。

11) 根据缺陷，可选择是否进行图象捕捉为了直观的观察缺陷或缺陷现象，通常需要附加缺陷或缺陷出现的界面，以图片的形式作为附件附着在记录的“附件”部分。为了节省空间，又能真实反映缺陷或缺陷本质，可以捕捉缺陷或缺陷产生时的全屏幕，活动窗口和局部区域。为了迅速定位、修正缺陷或缺陷位置，通常要求附加中文对照图。 附加必要的特殊文档和个人建议和注解如果打开某个特殊的文档而产生的缺陷或缺陷，则必须附加该文档，从而可以迅速再现缺陷或缺陷。有时，为了使缺陷或缺陷修正者进一步明确缺陷或缺陷的表现，可以附加个人的修改建议或注解。

12) 检查拼写和语法缺陷在提交每条缺陷或缺陷之前，检查拼写和语法，确保内容正确，正确的描述缺陷。

13) 尽量使用短语和短句，避免复杂句型句式软件缺陷管理数据库的目的是便于定位缺陷，因此，要求客观的描述操作步骤，不需要修饰性的词汇和复杂的句型，增强可读性。以上概括了报告测试缺陷的规范要求，随着软件的测试要求不同，测试者经过长期测试，积累了相应的测试经验，将会逐渐养成良好的专业习惯，不断补充新的规范书写要求。此外，经常阅读、学习其他测试工程师的测试缺陷报告，结合自己以前的测试缺陷报告进行对比和思考，可以不断提高技巧。

14) 缺陷描述内容缺陷描述的内容可以包含缺陷操作步骤，实际结果和期望结果。操作步骤可以方便开发人员再现缺陷进行修正，有些开发的再现缺陷能力很差，虽然他明白你所指的缺陷，但就是无法再现特别是对系统不熟悉的新加入开发人员，介绍步骤可以方便他们再现。实际结果可以让开发明白错误是什么，期望结果可以让开发了解正确的结果应该是如何。

39.【简答题】【测试缺陷】BUG 管理工具的跟踪过程(用 BugZilla 为例子)

测试人员发现了 BUG，提交到 Bugzilla 中，状态为 new，BUG 的接受者为开发接口人员

开发接口将 BUG 分配给相关的模块的开发人员，状态修改为已分配，开发人员和测试确认 BUG，如果是本人的 BUG，则设置为接收;如果是别的开发人员的问题，则转发出去，由下一个开发人员来进行此行为;如果认为不是问题，则需要大家讨论并确认后，拒绝这个 BUG，然后测试人员关闭此问题。

如果开发人员接受了 BUG，并修改好以后，将 BUG 状态修改为已修复，并告知测试在哪个版本中可以测试。

测试人员在新版本中测试，如果发现问题依然存在，则拒绝验证;如果已经修复，则关闭 BUG。

40.【简答题】【测试缺陷】您以往所从事的软件测试工作中，是否使用了一些工具来进行软件缺陷(Bug)的管理?如果有，请结合该工具描述软件缺陷(Bug)跟踪管理的流程。

CQ，也可以使用 BugFree、禅道等免费工具。

41.【简答题】【测试缺陷】请问如果用户点击微博的关注图标但是 app 上面没有反应，应该怎么排查这个问题

首先 1.在 Eclipse Devices 窗口，选中 app 对应的包名，然后点击 debug 图标（绿色的小虫子），然后切换到 Debug 视图。

2.切换视图之后，可以看到 debug 下，app 的线程列表。

3.对于 main 线程（第一个线程），选中，并将其挂起 Suspend。

4.然后我们就可以看到，Suspend 之后，main 线程卡住的位置。

42.【简答题】【测试缺陷】请问如果想进行 bug 的测评，怎么去评测 bug？

Bug 的 priority（）和 severity（）是两个重要属性，通常人员在提交 bug 的时候，只定义 severity，而将 priority 交给 leader 定义，通常 bug 管理中，severity 分为四个等级 blocker、critical、major、minor/trivial，而 priority 分为五个等级 immediate、urgent、high、normal、low。

Severity：

1、blocker：即系统无法执行，崩溃，或严重资源不足，应用模块无法启动或异常退出，无法测试，造成系统不稳定。常见的有严重花屏、内存泄漏、用户数据丢失或破坏、系统崩溃/死机/冻结、模块无法启动或异常退出、严重的数值计算错误、功能设计与需求严重不符、其它导致无法测试的错误， 如服务器 500 错误。

2、critical：即映像系统功能或操作，主要功能存在严重缺陷，但不会映像到系统稳定性。常见的有：功能未实现，功能错误、系统刷新错误、数据通讯错误、轻微的数值计算错误、影响功能及界面的错误字或拼写错误。

3、major：即界面、性能缺陷、兼容性，常见的有：操作界面错误，边界条件错误，提示信息错误，长时间操作无进度提示，系统未优化，兼容性问题。

4、minor/trivial：即易用性及建议性问题。

Priority

1、immediate：即马上解决，

2、urgent：急需解决

3、high：高度重视，有时间要马上解决

4、low：在系统发布前解决，或确认可以不用解决。

43.【简答题】【测试缺陷】请问你遇到过哪些印象深刻的 bug，接口测试出现 bug 的原因有哪些？

面试官询问遇到过哪些印象深刻的 bug，其实它并不关心你描述的这个 bug 是否真的有价值，或有多曲折离奇？他只是：了解你平时工作中的测试能力

所以，这就要求的你平时工作中遇到 bug 时试着自己去定位，定位 bug 的过程远比你的单纯的执行测试用例有“价值”（自我技能提高的价值），在定位 bug 的过程中你需要掌握和运用更多知识。

另外，建议你平时养成总结的好习惯，发现的 bug，开发解决了，最好问问他原因以及解决的方法，这样再遇到类似问题时，自己也可以试着定位解决。遇到难解决的 bug，也可以把最终的解决过程记录下来。（这不是就有素材了）

所以，建议你平时可以主动要求去分享一些自己工作中用到或学习的技术。或者多去参加集体活动，加强自己的表达能力。From:虫师

接口测试常见的 bug 有以下几个:

特殊值处理不当导致程序异常退出或者崩溃

类型边界溢出，导致数据独处和写入不一致

取值边界外未返回正确的错误信息

权限未处理，可以访问其他用户的信息

逻辑校验不完善，可以利用漏洞获取非正当利益

状态处理不当，导致逻辑出现错误

数组类型 item 个数为 0 或者 item 重复时程序异常退出

44.【简答题】【测试缺陷】简单概述缺陷报告，并说明包括哪些项？

现在缺陷报告一般不再使用纸质档文档编写，而是专用测试管理工具（如 TestDirector），这样便于缺陷管理。在这些工具中，每个缺陷作为一条记录输入指定的缺陷管理系统中。

缺陷报告包括：软件名称、版本号、功能模板、缺陷编号、对应的用例编号、编写时间、编写人、测试员、预期结果、实际结果、缺陷描述、严重级别、优先级别

45.【简答题】【测试缺陷】什么是 bug？

软件的 bug 指的是软件中（包括程序和文档）不符合用户需求的问题。

常见的软件 bug 分为以下三类：

没有实现的功能

完成了用户需求的功能，但是运行时会出现一些功能或性能上的问题

实现了用户不需求的多余功能

46.【简答题】【测试缺陷】所有的软件缺陷都能修复吗？所有的软件缺陷都要修复吗？

从理论上来说所有的缺陷都是可以修复的，但是并不是所有的缺陷都要修复。

一些对于软件没有影响的、不影响使用的缺陷我们可以不用修复。因为修复些细小的缺陷也是需要花费很多时间。项目上面可能会因为时间问题而先忽略这些小缺陷

47.【简答题】【测试缺陷】你在测试中发现了一个 bug，但是开发经理认为这不是一个 bug，你应该怎样解决?

首先，将问题提交到缺陷管理库里面进行备案。

然后，要获取判断的依据和标准：

根据需求说明书、产品说明、设计文档等，确认实际结果是否与计划有不一致的地方，提供缺陷是否确认的直接依据;

如果没有文档依据，可以根据类似软件的一般特性来说明是否存在不一致的地方，来确认是否是缺陷;

根据用户的一般使用习惯，来确认是否是缺陷;

与设计人员、开发人员和客户代表等相关人员探讨，确认是否是缺陷;

合理的论述，向测试经理说明自己的判断的理由，注意客观、严谨，不参杂个人情绪。

等待测试经理做出最终决定，如果仍然存在争议，可以通过公司政策所提供的渠道，向上级反映，并有上级做出决定。

48.【简答题】【缺陷报告】一个缺陷测试报告的组成？

缺陷编号、缺陷标题、缺陷描述、缺陷的优先级、缺陷的重要程度、缺陷所述的模块、缺陷所属的版本、缺陷所属的开发人员、输入数据、输出结果、缺陷分析等。

49.【简答题】【测试流程】你认为做好测试计划工作的关键是什么？

首先，要有一个明确的目标，详细的阅读需求文档说明。

其次，要对整个测试人员、测试时间、测试进度进行一个预估，并预先进行管理。

最后，要对整个测试流程设定一个规范，所有测试人员都按着规范做事，不能随心所欲的测试。

50.【简答题】【测试流程】开发人员老是犯一些低级错误怎么解决？

要在开发的前期就制定好一些编码规范，这样子可以减少很多因为个人习惯引起的错误。同时，测试人员在发现开发人员犯一些低级错误的时候不可以指责他们，要耐心的给他们指出错误所在。然后可以有开发人员自己进行测试，找出一些一眼看得出来是错误的地方。

51.【简答题】【测试流程】您在以往的测试工作中都曾经具体从事过哪些工作？其中最擅长哪部分工作？

我一般都是做的 Web 测试，搭建测试环境，对于一个程序进行集成测试，系统测试，回归测试等。还要编写测试用例以及一些文档，用户使用手册，功能测试文档等等。最擅长的是功能测试。

52.【简答题】【测试流程】开发人员说不是 bug 时，你如何应付？

首先把自己的理由告诉开发人员。在同开发人员沟通到底是不是 bug，但是如果开发人员还是认为不是 bug 的话，就把这个问题提到项目经理处，同时附上自己的理由。有项目经理决定是否为 bug。

53.【简答题】【测试流程】当测试过程发生错误时，有哪几种解决办法？

1）跳转到别的测试过程

2）调用一个能够清除错误的过程

3）退出过程，启用另一个

4）退出过程和应用程序，重新启动 Windows，在失败的地方重新开始测试

54.【简答题】【测试流程】怎么才能够全面的测试到每一个点？

测试的全面性主要需要在设计测试计划的时候考虑，从测试策略，产品需求等等多个角度考虑从而定义全部的测试点。

55.【简答题】【测试流程】开发与测试的关系？

开发和测试是一个有机的整体。在产品发布之前，开发和测试是循环进行的，测出的缺陷要经开发人员修改后继续测试。在开发的同时测试经理开始编写测试用例，测试文档要参考开发文档，所以开发和测试是不可分割的，少了任何一个都不能开发出产品。

56.【简答题】【测试流程】开发人员修复缺陷后，如何保证不影响其他功能？

重新执行用例、看是否出现错误结果。并对周围的一些相关功能点追加新的测试用例。

57.【简答题】【测试流程】项目中相关需求问题，测试可以直接和客户沟通吗？

A1：可以，最初与客户沟通需求时，测试人员直接参与，所以我们可以直接和客户方的代表开会进行沟通。

A2：不可以，一般情况下我们需要将问题整理到一起，由项目经理和测试经理作为接口人和客户进行沟通。

A3：不可以，我们的需求是产品线提的，产品线与客户直接沟通，所以关于需求问题我们直接找产品线。

58.【简答题】【测试流程】什么叫预测试，预测试是怎么进行的，预测试一般为多长时间？

预测试就是开放刚刚开发完成，测试环境刚搭建起来，这时我们要对系统的各种功能能不能跑通，业务流程能不能完成进行测试，就是冒烟测试，这就是转测试，我们转测试大概需要一天的时间。

59.【简答题】【测试流程】详细的描述一个测试活动完整的过程。(供参考，本答案主要是瀑布模型的做法)

项目经理通过和客户的交流，完成需求文档，由开发人员和测试人员共同完成需求文档的评审，评审的内容包括：需求描述不清楚的地方和可能有明显冲突或者无法实现的功能的地方。项目经理通过综合开发人员，测试人员以及客户的意见，完成项目计划。然后 SQA 进入项目，开始进行统计和跟踪

开发人员根据需求文档完成需求分析文档，测试人员进行评审，评审的主要内容包括是否有遗漏或双方理解不同的地方。测试人员完成测试计划文档，测试计划包括的内容上面有描述。

测试人员根据修改好的需求分析文档开始写测试用例，同时开发人员完成概要设计文档，详细设计文档。此两份文档成为测试人员撰写测试用例的补充材料。

测试用例完成后，测试和开发需要进行评审。

测试人员搭建环境

开发人员提交第一个版本，可能存在未完成功能，需要说明。测试人员进行测试，发现 BUG 后提交给 BugZilla。

开发提交第二个版本，包括 Bug Fix 以及增加了部分功能，测试人员进行测试。

重复上面的工作，一般是 3-4 个版本后 BUG 数量减少，达到出货的要求。

如果有客户反馈的问题，需要测试人员协助重现并重新测试。

60.【简答题】【测试流程】您认为在测试人员同开发人员的沟通过程中，如何提高沟通的效率和改善沟通的效果?维持测试人员同开发团队中其他成员良好的人际关系的关键是什么？

尽量面对面的沟通，其次是能直接通过电话沟通，如果只能通过 Email 等非及时沟通工具的话，强调必须对特性的理解深刻以及能表达清楚。

运用一些测试管理工具如 TestDirector 进行管理也是较有效的方法，同时要注意在 TestDirector 中对 BUG 有准确的描述。

在团队中建立测试人员与开发人员良好沟通中注意以下几点：

一真诚、二是团队精神、三是在专业上有共同语言、四是要对事不对人，工作至上

当然也可以通过直接指出一些小问题，而不是进入 BUG Tracking System 来增加对方的好感。

61.【简答题】【测试流程】通过画因果图来写测试用例的步骤为___、___、___、___及把因果图转换为状态图共五个步骤。 利用因果图生成测试用例的基本步骤是：

分析软件规格说明描述中，哪些是原因(即输入条件或输入条件的等价类)，哪些是结果(即输出条件)，并给每个原因和结果赋予一个标识符。

分析软件规格说明描述中的语义，找出原因与结果之间，原因与原因之间对应的是什么关系？根据这些关系，画出因果图。

由于语法或环境限制，有些原因与原因之间，原因与结果之间的组合情况不可能出现。为表明这些特殊情况，在因果图上用一些记号标明约束或限制条件。

把因果图转换成判定表。

把判定表的每一列拿出来作为依据，设计测试用例。

62.【简答题】【测试流程】软件测试项目从什么时候开始?为什么？

软件测试应该在需求分析阶段就介入,因为测试的对象不仅仅是程序编码,应该对软件开发过程中产生的所有产品都测试,并且软件缺陷存在放大趋势.缺陷发现的越晚,修复它所花费的成本就越大.

63.【简答题】【测试流程】一套完整的测试应该由哪些阶段组成?

可行性分析、需求分析、概要设计、详细设计、编码、单元测试、集成测试、系统测试、验收测试

64.【简答题】【测试流程】为什么要在一个团队中开展软件测试工作?

因为没有经过测试的软件很难在发布之前知道该软件的质量，就好比 ISO 质量认证一样，测试同样也需要质量的保证，这个时候就需要在团队中开展软件测试的工作。在测试的过程发现软件中存在的问题，及时让开发人员得知并修改问题，在即将发布时，从测试报告中得出软件的质量情况。

65.【简答题】【测试流程】在做测试的过程中，假如前端和后端吵起来了都在踢皮球觉得对方该改代码，你怎么办？

此时就应该找技术领导拍板或 leader 们基于安全性、性能、可测试性、可维护性讨论敲定一个解决方案，做到开发环境方便开发，线上环境少配置

少依赖、少出错机会。

66.【简答题】【测试流程】请问你觉得测试项目具体工作是什么？

搭建测试环境

撰写测试用例

执行测试用例

写测试计划，测试报告

测试，并提交 BUG 表单

跟踪 bug 修改情况

执行自动化测试，编写脚本，执行，分析，报告

进行性能测试，压力测试等其他测试，执行，分析，调优，报告

67.【简答题】【测试流程】请你回答一下测试的相关流程是什么？

测试最规范的过程如下

需求测试->概要设计测试->详细设计测试->单元测试->集成测试->系统测试->验收测试

68.【简答题】【测试流程】请你说一说当前工作中涉及的测试问题(测试流程和测试性能)

在测试性能中，时常会出现脚本回访卡住的问题，原因有以下几种：

1、 runtimesetting 中的 continue error 没有勾选

2、录制的脚本中存在冗余的代码部分，需要对脚本进行优化，去除冗余的部分（优化脚本）

例如：在用 FireFox 录制脚本时，脚本中会产生一个叫

”Url=http://download.cdn.mozilla.net/pub/firefox/releases/43.0.1/update/win32/zh-CN/firefox-43.0.1.complete.mar","Referer=", ENDITEM,”这样的代码（该代码出现的问题不止一处，在查找时一定要注意。），这是因为采用 firefox 浏览器录制时产生的压缩文件，在脚本回放时卡住的原因正是因为这个（建议：能采用 IE 录制尽量用 IE 浏览器）

解决办法：注释掉或者删除掉该段代码即可， 关联问题：在用 loadrunner 自带对比工具对比脚本后 找到需要关联的动态值。在关联后回放脚本时报错 HTTP-status code 417（exception failed）错误时，产生的原因如下：

1、脚本中还存在没有关联或者关联失败的动态值，利用 lr 自带对比工具仔细对比

2、脚本中的动态值被做了加密策略，仔细查看脚本中动态值的部分，看看动态值是否被做了安全策略（随机生成或者打乱动态值顺序、在动态值中加入了特殊符号），由于在 tree-response 中的动态值是未被加密的状态，在 client 向 server 发送请求时，client 的动态值发给服务器，这时服务器的动态值已经被做了参数化，所以服务器不认准 client 向服务器发送的动态值。

解决办法：去掉动态值的安全策略即可（JVM 参数）

69.【简答题】【测试流程】软件测试分为几个阶段 各阶段的测试策略和要求是什么？

和开发过程相对应，测试过程会依次经历单元测试、集成测试、系统测试、验收测试四个主要阶段：

单元测试：单元测试是针对软件设计的最小单位—程序模块甚至代码段进行正确性检验的测试工作，通常由开发人员进行。

集成测试：集成测试是将模块按照设计要求组装起来进行测试，主要目的是发现与接口有关的问题。由于在产品提交到测试部门前，产品开发小组都要进行联合调试，因此在大部分企业中集成测试是由开发人员来完成的。

系统测试：系统测试是在集成测试通过后进行的，目的是充分运行系统，验证各子系统是否都能正常工作并完成设计的要求。它主要由测试部门进行，是测试部门最大最重要的一个测试，对产品的质量有重大的影响。

验收测试：验收测试以需求阶段的《需求规格说明书》为验收标准，测试时要求模拟实际用户的运行环境。对于实际项目可以和客户共同进行，对于产品来说就是最后一次的系统测试。测试内容为对功能模块的全面测试，尤其要进行文档测试。

单元测试测试策略：

自顶向下的单元测试策略：比孤立单元测试的成本高很多，不是单元测试的一个好的选择。

自底向上的单元测试策略：比较合理的单元测试策略，但测试周期较长。

孤立单元测试策略：最好的单元测试策略。

集成测试的测试策略：

大爆炸集成：适应于一个维护型项目或被测试系统较小

自顶向下集成：适应于产品控制结构比较清晰和稳定;高层接口变化较小;底层接口未定义或经常可能被修改;产口控制组件具有较大的技术风险，需要尽早被验证;希望尽早能看到产品的系统功能行为。

自底向上集成：适应于底层接口比较稳定;高层接口变化比较频繁;底层组件较早被完成。

基于进度的集成

优点：具有较高的并行度;能够有效缩短项目的开发进度。

缺点：桩和驱动工作量较大;有些接口测试不充分;有些测试重复和浪费。

系统测试的测试策略：

数据和数据库完整性测试;功能测试;用户界面测试;性能评测;负载测试;强度测试;容量测试;安全性和访问控制测试;故障转移和恢复测试;配置测试;安装测试;加密测试;可用性测试;版本验证测试;文档测试

70.【简答题】【测试用例】功能测试用例需要详细到什么程度才是合格的？

测试用例覆盖到所有的测试点。

71.【简答题】【测试用例】测试用例通常包括哪些内容？

用例编号、测试环境、用例标题、输入数据、预期结果等

72.【简答题】【测试用例】一个项目需要写多少测试用例怎么估算？

这个在需求分析之后根据测试点来评估的，我们的测试点写的很细，所以测试用例的数目几乎等于测试点的数目。

73.【简答题】【测试用例】不能发现 BUG 的测试用例不是好的测试用例吗？

我不这样认为，我觉得在执行之前，每个用例都可能发现缺陷，好的测试用例是一套完整的不遗漏的测试用例，是能够被其他的测试人员执行的测试用例。不能因为是否找到 BUG 来说明用例是否好。

74.【简答题】【测试用例】什么是测试用例 什么是测试脚本 两者的关系是什么？

为实施测试而向被测试系统提供的输入数据、操作或各种环境设置以及期望结果的一个特定的集合。

测试脚本是为了进行自动化测试而编写的脚本。

测试脚本的编写必须对应相应的测试用例

75.【简答题】【测试用例】您所熟悉的测试用例设计方法都有哪些？请分别以具体的例子来说明这些方法在测试用例设计工作中的应用。

有黑盒和白盒两种测试种类，黑盒有等价类划分法，边界分析法，因果图法和错误猜测法。白盒有逻辑覆盖法，循环测试路径选择，基本路径测试。

例子：在一次输入多个条件的完整性查询中。利用等价类划分法则和边界分析法则，首先利用等价划分法，可以一个或多个结果是 OK 的测试用例，然后确认多个 NG 的测试用例，然后利用边界值分析法，可以对结果分别是 OK 和 NG 的测试用例进行扩展和补充。

76.【简答题】【测试用例】您认为做好测试用例设计工作的关键是什么？

测试用例设计工作的关键是对可行的和不可行的都要考虑。

1，输入 2，详细的操作步骤 3，预期输出 4，实际输出。

77.【简答题】【测试用例】测试用例如何设计的？

在测试用例的设计之前首先要仔细阅读开发的详细设计文档，充分了解产品的详细功能，不清楚的地方与开发人员进行沟通，搞懂每个功能，尽量详细到输入框、按钮等小功能，功能点清楚之后按照功能模块分类进行用例编写。在具体的用例设计中会运用到等价类边界值等黑盒测试方法

78.【简答题】【测试用例】测试用例设计方法有哪些？

等价类、边界值、错误推测法、场景法、因果图、判定表。

79.【简答题】【测试用例】测试用例内容有哪些？

ID 、标题、 优先级、 预置条件 、操作步骤 、预期结果、 实际结果、测试人、测试时间。

80.【简答题】【测试用例】测试用例为什么需要有优先级，有哪一些优先级？

因为在不同阶段执行的用例数目是不同的，用例对应的功能的重要程度也是不同的，我们用的是高中低三级。

81.【简答题】【测试用例】你们项目一共有多少条测试用例？

500-------到 2000，具体项目具体分析，和项目大小颗粒度大小都有关系。

82.【简答题】【测试用例】黑盒测试的测试用例常见设计方法都有哪些?请分别以具体的例子来说明这些方法在测试用例设计工作中的应用。

1)等价类划分： 等价类是指某个输入域的子集合.在该子集合中,各个输入数据对于揭露程序中的错误都是等效的.并合理地假定:测试某等价类的代表值就等于对这一类其它值的测试.因此,可以把全部输入数据合理划分为若干等价类,在每一个等价类中取一个数据作为测试的输入条件,就可以用少量代表性的测试数据.取得较好的测试结果.等价类划分可有两种不同的情况:有效等价类和无效等价类.

2)边界值分析法：是对等价类划分方法的补充。测试工作经验告诉我,大量的错误是发生在输入或输出范围的边界上,而不是发生在输入输出范围的内部.因此针对各种边界情况设计测试用例,可以查出更多的错误.

使用边界值分析方法设计测试用例,首先应确定边界情况.通常输入和输出等价类的边界,就是应着重测试的边界情况.应当选取正好等于,刚刚大于或刚刚小于边界的值作为测试数据,而不是选取等价类中的典型值或任意值作为测试数据.

3)错误猜测法：基于经验和直觉推测程序中所有可能存在的各种错误, 从而有针对性的设计测试用例的方法.

错误推测方法的基本思想: 列举出程序中所有可能有的错误和容易发生错误的特殊情况,根据他们选择测试用例. 例如, 在单元测试时曾列出的许多在模块中常见的错误. 以前产品测试中曾经发现的错误等, 这些就是经验的总结. 还有, 输入数据和输出数据为 0 的情况. 输入表格为空格或输入表格只有一行. 这些都是容易发生错误的情况. 可选择这些情况下的例子作为测试用例.

4)因果图方法：前面介绍的等价类划分方法和边界值分析方法,都是着重考虑输入条件,但未考虑输入条件之间的联系, 相互组合等. 考虑输入条件之间的相互组合,可能会产生一些新的情况. 但要检查输入条件的组合不是一件容易的事情, 即使把所有输入条件划分成等价类,他们之间的组合情况也相当多. 因此必须考虑采用一种适合于描述对

于多种条件的组合,相应产生多个动作的形式来考虑设计测试用例. 这就需要利用因果图(逻辑模型). 因果图方法最终生成的就是判定表. 它适合于检查程序输入条件的各种组合情况.

5)正交表分析法：可能因为大量的参数的组合而引起测试用例数量上的激增，同时，这些测试用例并没有明显的优先级上的差距，而测试人员又无法完成这么多数量的测试，就可以通过正交表来进行缩减一些用例，从而达到尽量少的用例覆盖尽量大的范围的可能性。

6)场景分析方法：指根据用户场景来模拟用户的操作步骤，这个比较类似因果图，但是可能执行的深度和可行性更好。

7)状态图法：通过输入条件和系统需求说明得到被测系统的所有状态，通过输入条件和状态得出输出条件;通过输入条件、输出条件和状态得出被测系统的测试用例。

8)大纲法：大纲法是一种着眼于需求的方法，为了列出各种测试条件，就将需求转换为大纲的形式。大纲表示为树状结构，在根和每个叶子结点之间存在唯一的路径。大纲中的每条路径定义了一个特定的输入条件集合，用于定义测试用例。树中叶子的数目或大纲中的路径给出了测试所有功能所需测试用例的大致数量。

83.【简答题】【测试用例】请你说一下如何写测试用例

1、测试人员尽早介入，彻底理解清楚需求，这个是写好测试用例的基础

2、如果以前有类似的需求，可以参考类似需求的测试用例，然后还需要看类似需求的 bug 情况

3、清楚输入、输出的各种可能性，以及各种输入的之间的关联关系，理解清楚需求的执行逻辑，通过等价类、边界值、判定表等方法找出大部分用例

4、找到需求相关的一些特性，补充测试用例

5、根据自己的经验分析遗漏的测试场景

6、多总结类似功能点的测试点，才能够写出质量越来越高的测试用例

7、书写格式一定要清晰

84.【简答题】【测试用例】您认为做好测试用例设计工作的关键是什么？

白盒测试用例设计的关键是以较少的用例覆盖尽可能多的内部程序逻辑结果

黑盒法用例设计的关键同样也是以较少的用例覆盖模块输出和输入接口。不可能做到完全测试，以最少的用例在合理的时间内发现最多的问题

85.【简答题】【测试用例】您认为做好测试用例设计工作的关键是什么？

对业务和软件需求非常清楚，可以根据需求不同选择不同的测试用例设计

86.【简答题】【测试用例】请你设计一个微信朋友圈点赞的测试用例

功能测试：

点赞某条朋友圈，验证是否成功

接口测试：

点赞朋友圈，验证朋友能否收到提示信息

性能测试

点赞朋友圈，是否在规定时间显示结果，是否在规定时间在朋友手机上进行提示

兼容性测试

在不同的终端比如 ipad,手机上点赞朋友圈，验证是否成功

87.【简答题】【测试用例】请你说一说洗牌问题的思路并手写代码，并设计测试用例

洗牌问题：有个长度为 2n 的数组{a1,a2,a3,…,an,b1,b2,b3,…,bn}，希望排序后{a1,b1,a2,b2,….,an,bn}，请考虑有无时间复杂度 o(n)，空间复杂度 0(1)的解法。

```
void PerfectShuffle(int *A,int n){
if(n <= 1){
return;
}//if
//
int size = 2*n;
int index,count;
for(int i = n;i < size;++i){
// 交换个数

count = n - (i - n) - 1;
```

// 待交换

```
index = i;
for(int j = 1;j <= count;++j){
swap(A[index],A[i-j]);
index = i - j;
}//for
}//for
}
};
```

可以就数组的类型，可以是 int 型的，浮点型的，还可以是大数类型，负数，进行测试。

88.【简答题】【测试用例】设计测试用例时应该考虑哪些方面，即不同的测试用例针对那些方面进行测试?

设计测试用例时需要注意的是，除了对整体流程及功能注意外，还要注意强度测试、性能测试、压力测试、边界值测试、稳定性测试、安全性测试等多方面。(测试用例需要考虑的四个基本要素是输入、输出、操作和测试环境;另外，测试用例需要考虑的是测试类型(功能、性能、安全……)，这部分可以参照 TP 做答。此外，还需要考虑用例的重要性和优先级)

89.【简答题】【测试用例】在 windows 下保存一个文本文件时会弹出保存对话框，如果为文件名建立测试用例，等价类应该怎样划分?

单字节，如 A;双字节， AA、我我;特殊字符 /‘。‘;、=-等;保留字，如 com;文件格式为 8.3 格式的;文件名格式为非 8.3 格式的;/,\,*等九个特殊字符。

90.【简答题】【测试用例】假设有一个文本框要求输入 10 个字符的邮政编码，对于该文本框应该怎样划分等价类?

特殊字符，如 10 个*或￥;英文字母，如 ABCDefghik;小于十个字符，如 123;大于十个字符，如 11111111111;数字和其他混合，如 123AAAAAAA;空字符;保留字符

91.【单选题】【测试用例】使用白盒测试方法时，设计测试用例应根据（）？

A、程序的内部逻辑
B、程序的复杂结构
C、程序的功能
D、使用说明书

答案：A。

解析：白盒测试又称为结构测试或逻辑驱动测试，它允许测试人员利用程序内部的逻辑结构及有关信息来设计或选择测试用例，对程序所有的逻辑路径进行测试，故 A 选项正确。

92.【简答题】【测试用例】测试用例设计的原则是什么?目前主要的测试用例设计方法有哪些?

代表性：能够代表并覆盖各种合理的和不合理、合法的和非法的、边界的和越界的、以及极限的输入数据、操作和环境设置等.

可判定性：即测试执行结果的正确性是可判定的，每一个测试用例都应有相应的期望结果.

可再现性：即对同样的测试用例，系统的执行结果应当是相同的。

方法有等价类、边界值、因果图、状态图、正交法、大纲法

93.【简答题】【测试用例】面向对象的测试用例设计有几种方法?如何实现?

给类中的每个构造函数设计一组测试用例

组合类中的类变量、实例变量

组合类中的各种方法

根据前置条件和后置条件设计测试用例

根据代码设计测试用例

94.【简答题】【测试实战】请你回答一下如何测试手机开机键？

功能测试：

按下开机键，屏幕能否亮起

性能测试：

按下开机键，屏幕能否在规定时间内亮起

压力测试

连续多次按下开机键，观察屏幕是否能一直亮起，到多久时间失灵

健壮性测试

给定一个中了病毒的手机或者是淘汰许久的老机子，安歇开机键观察屏幕能否亮起

可靠性测试

连续按下开机键有限次数，比如 1 万次，记录屏幕未亮起的次数

可用性测试

开机键按下费不费力，开机键的形状设计是否贴合手指，开机键的位置设计是否方便

95.【简答题】【测试实战】请你测试一下游戏中英雄的技能

测试的设计都是通用的，首先功能测试看功能有没有实现，然后再对性能、压力、容量、健壮性、安全性、可靠性、恢复性、备份、协议、兼容性、可用性、配置、GUI 这些非功能测试去思考。具体答案这里不再赘述

96.【简答题】【测试实战】请问测试路由器怎么测，用命令行还是界面？

可以采用 lperf 这个命令，

Lperf 是一个网络性能测试工具，可以测量最大 tcp 和 udp 带宽，具有多种参数和特性，可以记录带宽，延迟抖动，数据包丢失，通过这些信息可以发现网络问题，检查网络质量，定位网络瓶颈。

iperf 的使用非常简单，测试的原理是在 wan 口连接一台 PC 机，在 LAN 口连接一台 PC，两边分别运行 iperf 服务端和客户端模式，用来测量 LAN->WAN 和 WAN->LAN 性能。具体命令如下：

服务端：iperf -s -w 1m

客户端：iperf -c <server ip> -w 1m -t 20 -P 10

含义是 TCP wndowsize 为 1MByte，测试时间是 20s，线程是 10。

97.【简答题】【测试实战】请问如何对登录界面进行测试

黑盒测试方法

输入正确用户名和密码，验证是否登陆成功

输入正确的用户名和错误的密码，验证是否登陆失败并且提示信息正确

输入未注册的用户名和任意的密码，验证是否登陆失败并且提示信息正确

用户名和密码都为空，验证是否登陆失败并且提示信息正确

用户名和密码两者之一为空

若启用了验证码，输入正确的用户名密码验证码是否能登陆成功

输入正确用户名和密码，错误的验证码，能否登陆成功并且提示信息正确

用户名和密码是否大小写敏感

页面上的密码框是否加密显示

后台系统第一次创建的用户重新登录时是否提示修改密码

忘记用户名和忘记密码的功能是否可用

前段功能是否根据要求限制用户名和密码的长度

点击验证码图片是否可以更换验证码，更换后的验证码是否可用

刷新页面是否会刷新验证码

如果验证码具有时效性，分别验证时效内和时效外验证码的有效性

用户登录成功但是会话超时后是否重定向到用户登录界面

不同级别的用户登录系统后的权限是否正确

页面默认定位焦点是否定位到用户名输入框中

快捷键 tab 和回车键是否可以正常使用

非功能性需求，从安全，性能，兼容三个方面

安全：

用户密码后台存储是否加密

用户密码在网络传输过程中是否加密

密码是否具有有效期，密码有效期到期后是否提示修改密码

不登陆的时候直接在浏览框中输入登录界面后的 url 地址，是否会重新定位到登陆界面

密码输入框是否不支持复制粘贴

页面密码输入框中输入的密码是否可以在页面源码模式下被查看

用户名和密码输入框中输入 xss 跨站脚本攻击字符串验证系统的行为是否被篡改

连续多次登陆失败后系统是否会阻止用户后续的尝试

统一用户在同一终端的多种不同浏览器上登陆，验证登录功能的互斥性是否符合设计预期

同一用户先后在不同终端的浏览器上登陆用户名和密码输入框中输入典型的 sql 注入攻击字符串验证系统的返回页面，验证登陆是否有互斥性

性能测试：

单用户登陆的响应界面是否符合预期

单用户登陆时后台请求数量是否过多

高并发场景下用户登录的响应界面是否符合预期

高并发场景下服务端的监控指标是否符合预期

高集合点并发场景下是否存在资源死锁和不合理的资源等待

长时间大量用户连续登录和登出，服务器端是否存在内存泄漏

兼容性测试：

不同浏览器下验证登陆功能的页面显示和功能正确性

相同浏览器的不同版本下验证登陆功能的页面显示和功能正确性

不同终端的不同浏览器下验证登陆功能的页面显示和功能正确性

不同分辨率下……

补充：弱网测试

网络切换和网络延迟时登陆界面是否正常

是否支持第三方登陆

是否可记住密码，记住的密码是否加密

98.【简答题】【测试实战】给你一个网站，你如何测试?

首先，查找需求说明、网站设计等相关文档，分析测试需求。

制定测试计划，确定测试范围和测试策略，一般包括以下几个部分：功能性测试;界面测试;性能测试;数据库测试;安全性测试;兼容性测试

设计测试用例：

功能性测试可以包括，但不限于以下几个方面：

链接测试。链接是否正确跳转，是否存在空页面和无效页面，是否有不正确的出错信息返回。

提交功能的测试。

多媒体元素是否可以正确加载和显示。

多语言支持是否能够正确显示选择的语言等。

界面测试可以包括但不限于一下几个方面：

页面是否风格统一，美观

页面布局是否合理，重点内容和热点内容是否突出

控件是否正常使用

对于必须但未安装的控件，是否提供自动下载并安装的功能

文字检查

性能测试一般从以下两个方面考虑：

压力测试;负载测试;强度测试

数据库测试要具体决定是否需要开展。数据库一般需要考虑连结性，对数据的存取操作，数据内容的验证等方面。

安全性测试：

基本的登录功能的检查

是否存在溢出错误，导致系统崩溃或者权限泄露

相关开发语言的常见安全性问题检查，例如 SQL 注入等

如果需要高级的安全性测试，确定获得专业安全公司的帮助，外包测试，或者获取支持

兼容性测试，根据需求说明的内容，确定支持的平台组合：

浏览器的兼容性;

操作系统的兼容性;

软件平台的兼容性;

数据库的兼容性

开展测试，并记录缺陷。合理的安排调整测试进度，提前获取测试所需的资源，建立管理体系(例如，需求变更、风险、配置、测试文档、缺陷报告、人力资源等内容)。

定期评审，对测试进行评估和总结，调整测试的内容。

99.【简答题】【测试实战】如何测试一个纸杯?

功能度：用水杯装水看漏不漏;水能不能被喝到

安全性：杯子有没有毒或细菌

可靠性：杯子从不同高度落下的损坏程度

可移植性：杯子在不同的地方、温度等环境下是否都可以正常使用

兼容性：杯子是否能够容纳果汁、白水、酒精、汽油等

易用性：杯子是否烫手、是否有防滑措施、是否方便饮用

用户文档：使用手册是否对杯子的用法、限制、使用条件等有详细描述

疲劳测试：将杯子盛上水(案例一)放 24 小时检查泄漏时间和情况;盛上汽油(案例二)放 24 小时检查泄漏时间和情况等

压力测试：用根针并在针上面不断加重量，看压强多大时会穿透

100.【简答题】【性能测试】在搜索引擎中输入汉字就可以解析到对应的域名，请问如何用 LoadRunner 进行测试。

建立测试计划，确定测试标准和测试范围

设计典型场景的测试用例，覆盖常用业务流程和不常用的业务流程等

根据测试用例，开发自动测试脚本和场景：

录制测试脚本：新建一个脚本(Web/HTML 协议);点击录制按钮，在弹出的对话框的 URL 中输入”about:blank”;在打开的浏览器中进行正常操作流程后，结束录制;调试脚本并保存，可能要注意到字符集的关联。

设置测试场景：针对性能设置测试场景，主要判断在正常情况下，系统的平均事务响应时间是否达标;针对压力负载设置测试场景，主要判断在长时间处于满负荷或者超出系统承载能力的条件下，系统是否会崩溃;执行测试，获取测试结果，分析测试结果

101.【简答题】【性能测试】什么是系统瓶颈？

系统瓶颈就是软件在一定的并发量、访问量下无法达到用户的需求。

比如说用户需要在 10s 内完成一个访问，但是每一次都要 12s 才能完成，这个就是性能瓶颈，有可能是程序本身的问题，也有可能和操作系统、软件相关。

102.【简答题】【性能测试】LoadRunner 分为哪三个模块?请简述各模块的主要功能。

Virtual User Generator：用于录制脚步

Mercury LoadRunner Controller：用于创建、运行和监控场景

Mercury LoadRunner Analysis：用于分析测试结果

103.【简答题】【性能测试】性能测试的流程？

1.测试需求分析 2.测试计划制定与评审 3.测试用例设计与开发 4.测试执行与监控 5.分析测试结果 6.编写性能测试报告 7.测试经验总结

104.【简答题】【性能测试】一台客户端有三百个客户与三百个客户端有三百个客户对服务器施压，有什么区别?

300 个用户在一个客户端上，会占用客户机更多的资源，而影响测试的结果。线程之间可能发生干扰，而产生一些异常。

300 个用户在一个客户端上，需要更大的带宽。

IP 地址的问题，可能需要使用 IP Spoof 来绕过服务器对于单一 IP 地址最大连接数的限制。

所有用户在一个客户端上，不必考虑分布式管理的问题;而用户分布在不同的客户端上，需要考虑使用控制器来整体调配不同客户机上的用户。同时，还需要给予相应的权限配置和防火墙设置。

105.【简答题】【性能测试】请你回答一下性能测试有哪些指标，对一个登录功能做性能测试，有哪些指标，怎么测出可同时处理的最大请求数量

性能测试常用指标：

从外部看，主要有

1、吞吐量：每秒钟系统能够处理的请求数，任务数

2、响应时间：服务处理一个请求或一个任务的耗时

3、错误率：一批请求中结果出错的请求所占比例

从服务器的角度看，性能测试关注 CPU，内存，服务器负载，网络，磁盘ＩＯ

对登录功能做性能测试

单用户登陆的响应界面是否符合预期

单用户登陆时后台请求数量是否过多

高并发场景下用户登录的响应界面是否符合预期

高并发场景下服务端的监控指标是否符合预期

高集合点并发场景下是否存在资源死锁和不合理的资源等待

长时间大量用户连续登录和登出，服务器端是否存在内存泄漏

怎么测出可同时处理的最大请求数量

可以采用性能测试工具(WeTest 服务器性能)，该工具是腾讯 wetest 团队出品，使用起来很简单方便，但测试功能相当强大，能提供 10w+以上的并发量，定位性能拐点，测出服务器模型最大并发

106.【简答题】【性能测试】你在做项目中有做过压力测试吗，怎么做

1、首先对要测试的系统进行分析，明确需要对那一部分做压力测试，比如秒杀，支付

2、如何对这些测试点进行施压

第一种方式可以通过写脚本产生压力机器人对服务器进行发包收报操作

第二点借助一些压力测试工具比如 Jmeter,LoadRunner

3、如何对这些测试点进行正确的施压

需要用压力测试工具或者其他方法录制脚本，模拟用户的操作

4、对测试点设计多大的压力比较合适？

需要明确压力测试限制的数量，即用户并发量

5、测试结束后如何通过这些数据来定位性能问题

通过测试可以得到吞吐量，平均响应时间等数据，这个数据的背后是整个后台处理逻辑综合作用的结果，这时候就可以先关注系统的 CPU，内存，然后对比吞吐量，平均响应时间达到瓶颈时这些数据的情况，然后就能确认性

能问题是系统的哪一块造成的

107.【简答题】【性能测试】对于有系统大量并发访问，你会如何做测试，有什么建议

如何做高并发系统的测试，一般而言，整体的测试策略是：先针对部分系统进行性能测试及压力测试，得到各部分的峰值处理性能，再模拟整体流程测试，重点测试整体业务流程以及业务预期负荷，着重测试以下几点：

1、不同省份，不同运营商 CDN 节点性能，可采用典型压力测试方案

2、核心机房 BGP 网络带宽，此部分重点在于测试各运行商的 BGP 网络可靠性，实际速率，一般采用 smokeping,lxChariot 等工具

3、各类硬件设备性能，一般采用专业的网络设备测试工具

4、各类服务器并发性能，分布式处理能力，可采用压力测试方案工具

5、业务系统性能，采用业务系统压力测试方案

6、数据库处理性能，这部分需要结合业务系统进行测试，以获取核心业务场景下的数据库的 TPS/QPS，

7、如果有支付功能，需要进行支付渠道接口及分流测试，此部分相对而言可能是最大的瓶颈所在，此外还涉及备份方案，容灾方案，业务降级方案的测试。

108.【简答题】【自动化测试】请你说一下能不能用机器学习去进行自动化测试，如何监控异常流量，如果是脉冲呢，如何和正常流量作区分

如何用机器学习去进行自动化测试，就是让自动化测试变得更加智能，在自动化测试过程中，当测试功能模块越来越多，没有太多的时间去全部覆盖，我们可以采用归纳学习的方式，基于自动化测试的执行结果或者手工测试执行的结果为数据输入，然后基于一定的模型（例如：以通过率和模块的重要率计算的平均值）得出测试推荐模块，或者当要执行一个功能模块时，基于历史数据和模型（bug 出现的错误相关性，功能相关性等）计算出与该功能模块相关性最大模块，并推荐测试。

如何监控异常流量

1、抓包

```
tcpdump -i eth0 -w server.cap
```

对包文件使用第三方工具如：wireshark 做分析

2、iftop

```
yum install iftop
```

3、iptraf

```
yum install iptraf –y  或  yum install iptraf-ng    -y
```

启动命令 ifptraf-ng

109.【简答题】【单元测试】请问你有没有做过什么单元测试，怎么进行单元测试，对一个没有参数没有返回值但可能对全局变量有影响的怎么进行单元测试

如何进行单元测试：

１、创建单元测试，该工具可以对任何类、接口、结构等实体中的字段、属性、构造函数、方法等进行单元测试。创建单元测试大致可以分为两类：

整体测试，整体测试是在类名称上右击鼠标，在下拉菜单中点击创建单元测试选项。这样就可以为整个类创建单元测试了，这时他会为整个类可以被测试的内容全部添加测试方法。开发人员直接在这些自动生成的测试方法中添加单元测试代码就可以了。

单独测试，如果只想单独对某个方法、属性、字段进行测试，则可以将鼠标焦点放在这个待测试的项目名称之上，然后点击鼠标右键，在右键菜单中选择创建单元测试选项。这样就可以单独为某个方法创建单元测试了。

运行单元测试

查看测试结果

编写单元测试代码

测试没有参数的函数，它可能还有别的输入，例如全局变量，成员变量，或调用子函数获得的输入（这个要使用工具才能做到），只要函数需读取的，都应该设定初始值，如果完全没有，没有输入也是一种输入，照样测试就是了。同样道理，输出也不仅仅是返回值，没有返回值还可能修改了全局变量什么的，这些也是要判断的输出。

110.【简答题】【压力测试】请问你有没有做过压力测试

在软件工程中，压力测试是对系统不断施加压力的测试，是通过确定一个系统的瓶颈或者不能接收的性能点，来获得系统能提供的最大服务级别的测试。例如测试一个 Web 站点在大量的负荷下，何时系考察公司：网易

统的响应会退化或失败。网络游戏中也常用到这个词汇。

111.【简答题】【测试需求】怎么熟悉需求

首先要熟悉需求，这是测试阶段必要的过程。

测试人员对需求不熟悉，是无法完成好测试的。业务越复杂的系统，要想做好测试，对业务就必须熟悉。

在需求过程中，一般产品经理会下发需求，让团队成员熟悉需求，再召集全员进行需求评审会议。

在收到需求第一时间就通读需求，不懂不明白的地方都做好记录，等评审会议上一并解决。

112.【简答题】【测试需求】如何测试需求

需求是由产品经理收集用户需求后转化为软件需求。在转化过程中，会因为产品经理个人思维的局限（技术层面的实现、测试的考量等）而造成需求不完善、描述有歧义、逻辑不够清晰。

人对自己的成果天然的包容性，一些显而易见的问题容易被忽略掉。

那么通过需求评审，借助团队的力量弥补产品经理个人思维的不足，从而达到完善需求的目的。

1.1 解决需求歧义

比如给你一个需求：

这是一个活动，满 300-50， 用户只要购买活动商品，自动扣减订单金额，每张订单只扣减一次。

这个需求未描述同一个账户是否可重复多次参与，那么就容易产生歧义了。作为一个用户，是否可以拆成多个订单来购买，达到多次参加活动的目的呢？

1.2 需求描述不完整

在需求中，某些重要的功能，只有输入没有对输出结果的描述。

1.3 不可测

主要针对数据来源不明确的情况。

比如某模块数据来源第三方，但是第三方没有提供测试环境等。

1.4 站在用户和业务角度，觉得不合常理

比如某电商系统中，未登录不能添加购物车。

1.5 不符合行业规范和法律法规相关规定

需要测试人员要有意识了解行业业务规范。

如增值税票开票信息中，纳税人识别号非必填。

1.6 业务知识

每个行业都是该行业的专业知识，通过对行业知识的学习，可以提高对软件隐性需求的理解。可以提高测试人员在需求过程中的参与度。

113.【简答题】【测试需求】需求文档不完善或不准确如何处理？

产品经理在解说需求文档时不可能完全没有问题，我经常会采用如下处理方式：

1. 如果遇到需求文档不完善或不准确，我会当场提出，与产品经理、开发人员等相关人员进行沟通交流，确保需求明确，然后做记录，提醒产品对需求文档进行补充更新之后再发送给相关人员；

2. 如果产品不能当场确认的需求，那么要记录到会议纪要中，后续在排期之前产品需要明确确认后发送邮件给相关人员；

3. 如果排期前还是无法确认，又需要预估时间的，那么排期比需求明确要略长

4. 在测试时，需求还没确认，那么也要开始测试。此时可以根据代码以及实际数据跑出的结果，查看系统当前实现功能，然后与 prd 要求进行比较，然后向产品和研发说明，确认系统实现功能。

114.【简答题】【测试需求】什么是测试需求？

确切地讲，所谓的测试需求就是在项目中要测试什么。我们在测试活动中，首先需要明确测试需求（What），才能决定怎么测（How），测试时间（When），需要多少人（Who），测试的环境是什么（Where），测试中需要的技能、工具以及相应的背景知识，测试中可能遇到的风险等等，以上所有的内容结合起来就构成了测试计划的基本要素。而测试需求是测试计划的基础与重点。

就像软件的需求一样，测试需求根据不同的公司环境，不同的专业水平，不同的要求，详细程度也是不同的。但是，对于一个全新的项目或者产品，测试需求力求详细明确，以避免测试遗漏与误解。

115.【简答题】【测试需求】为什么要做测试需求？

如果要成功的做一个测试项目，首先必须了解测试规模、复杂程度与可能存在的风险，这些都需要通过详细的测试需求来了解。所谓知己知彼，百战不殆。测试需求不明确，只会造成获取的信息不正确，无法对所测软件有一个清晰全面的认识，测试计划就毫无根据可言。活在自己世界里的人是可悲的，只凭感觉不做详细了解就下定论的项目是失败的。

测试需求越详细精准，表明对所测软件的了解越深，对所要进行的任务内容就越清晰，就更有把握保证测试的质量与进度。

如果把测试活动比作软件生命周期，测试需求就相当于软件的需求规格，测试策略相当于软件的架构设计，测试用例相当于软件的详细设计，测试执行相当于软件的编码过程。只是在测试过程中，我们把“软件”两个字全部替换成了“测试”。这样，我们就明白了整个测试活动的依据来源于测试需求。

116.【简答题】【测试需求】测试需求分析依据

通常是以被测产品的需求为原型进行分析转变而来，测试需求主要通过以下途径来进行收集：

与待测软件相关的各种文档资料。如软件需求规格、Use case、界面设计、项目会议或与客户沟通时有关于需求信息的会议记录、其他技术文档等。

与客户或系统分析员的沟通。

业务背景资料。如待测软件业务领域的知识等。

正式与非正式的培训。

其他。如果以旧系统为原型，以全新的架构方式来设计或完善软件，那么旧系统的原有功能跟特性就成为了最有效的测试需求收集途径。

117.【简答题】【测试需求】测试需求架构划分

测试需求分析应首先进行测试需求架构划分并先进行评审，通过后才进行后续的测试需求展开分析，从产品整体上考虑有哪些功能、测试类型需要进行分析，列出测试特性列表，也方便下一步展开具体分析。

首先，这里需要对功能进行一下定义以达成共识，功能是指能独立实现一个基本业务处理要求，为了降低测试需求设计的复杂性及依赖性，测试需求架构罗列的功能是指最小功能点，即不可再继续分解。

(1)应用程序：

A.一般是最底层的菜单项为最小功能点，若最底层的菜单项不能体现一个独立的业务流程时，可采用上一层的菜单项为最小功能点。

B. 还有某些比较特殊没有体现在菜单项的功能也需要作为最小功能点考虑，如 POS 应用程序中交易的冲正功能等。

(2)驱动：一般是以一个 API 为最小功能点。

然后，再考虑产品实际用户使用的场合及用户特点考虑哪些测试类型，如故障及恢复、功能集成、性能要求、安装测试、软硬件兼容性等，此处需要从产品层面考虑，而不是从功能点层面考虑。

118.【简答题】【测试需求】测试需求收集

测试需求的收集主要通过对测试依据进行分析整理，最后生成一个以测试的观点出发的 checklist（检查表），用来作为测试该软件的主要工作内容。检查表的检查要点包括需求的正确性、必要性、优先级、明确性、可测性、完整性、一致性、可修改性：

在整个信息收集过程中，务必确保软件的功能与特性被正确理解。因此，测试需求分析人员必须具备优秀的沟通能力与表达能力。

119.【简答题】【测试需求】需求确定中不确定的需求怎么解决？

一般情况下先由项目组内讨论解决，如果依旧得不到解决，则直接与需求方确认。

120.【简答题】【测试需求】需求测试的注意事项有哪些?

是否使用了公司的模板、文档内容是否符合规范、所有的需求是分级是否清析适当、所有的需求是否具有一致性、需求是否可行(即，该需求组合有解决方案)、需求可否用已知的约束来实现、需求是否足够(即，可以把它送到一个规范的开发组织，并有一个生产出所需要产品的合理的可能性)、所有的其它需求是交叉引用是否正确、用户描述是否清楚、是否用客户的语言来描述需求、每个需求描述是否清楚没有岐义，可以移交给一个独立的组去实现时也能理解、是否所有的需求都是可验证的、是否每条需求都具有独立性，即使发生了变化也不会影响其它需求、性能指标是否明确、非功能性需求是否得到充分表现、是否完整列出适用的标准或协议、标准和协议之间是否存在冲突。

121.【简答题】【测试计划】什么是测试计划？

1. 测试计划就像写论文一样，列出提纲，才能一步步完善，有了测试计划就会掌握整个项目的进度和方向，在工作中可以有个指导的作用，不会偏离工作方向。

2. 测试计划规定预期的目标，以什么样的程度完成和在预期多久内完成，这样的规定能够使工作人员做好心理准备，合理的期限和目标能够使工作人员不松懈，有效率的完成测试任务。

3. 计划作为对未来工作的规划，肯定会受到突发或者不可预知因素而导致整个项目出现延期甚至无法进行的结果。因此计划中对于风险评估的必要性就在于罗列出影响整个项目进行的因素，并制定相应紧急方案，将损失降至最小化。

4. 测试计划的制定是在需求分析完成之后进行的的，所以测试计划的执行在一定程度上也是对需求分析的进一步检验，如果在制定过程中，发现有不合理因素存在，还能及时反馈，进行调整，不至于使众多的人力做了无用

功。

5. 测试计划的安排也是一个项目中多个部门间合作的工作指导，一环扣一环，工作上的交接在时间上做好详细的备注，才能让部门的合作显得默契。

122.【简答题】【测试计划】为什么要制定测试计划？

1. 制定测试计划能够把自身知识和经验直接转化为执行任务的具体方法，在方法中更能体现出制定者的自身素质以及能力。

2. 制定测试计划可以促进团队间关于测试任务和过程的交流，增强团队之间的默契，可以高效率、高配合、高质量的完成测试任务。

3. 制定测试计划可以为组织、安排和管理测试项目提供一个整体框架，对项目执行过程中的风险进行分析，并制定相关的应对策略。

4. 制定测试计划中对人员的合理安排化，通过对每个员工的专长来对应分配难易任务，整个项目的进行就会显得合理化、层次化、条理化。同时将职责清晰地具体划分到个人身上，也有利于日后的纠错，即使发现哪个环节出现问题，及时弥补。

123.【简答题】【测试计划】什么时候制定测试计划？

一般情况下，在产品需求确认，做过测试需求分析之后我们就要开始编写测试计划。当然测试计划编写的工作需要根据工作实际情况来决定，也就是具体情况具体分析。下面是测试计划需要的内容：

1. 测试范围：明确测什么？测试的目的与项目的简介（目的、背景以及范围）。

2. 测试策略：明确怎么测？对于不同的业务需求，具体有哪些测试类型、测试场景以及测试方法。

3. 资源安排：包括测试人员的安排，测试环境、以及测试工具等。

4. 进度安排：在明确测试范围、方法和人员之后，我们要考虑什么时候开始测试，预计要测试多久？以便和产品上线计划衔接。

5. 达到上线条件的确认性：是达到测试完成和产品需要满足的条件，以便项目内所有角色都有一致认可的目标。

6. 风险备注：最后，我们需要对整个测试流程中可能存在的风险，以及当这些风险发生时的应对措施提前及逆行一些考虑和准备，并在测试计划中体现出来。

124.【简答题】【测试计划】怎样做好测试计划？

1）理解系统。从整个系统的高度了解被测系统必须满足的功能和非功能性需求。利用涉及整个系统的文档，形成对系统的整体了解。

2）及早介入。为了深入了解项目，测试人员应该在系统的开始阶段介入，可以增加对客户需求，客户问题，潜在风险以及最重要的功能方面的理解

3）测试期望。程序员的期望是什么？客户的期望是什么？销售对测试的期望又是什么？测试目标必须是绝对

的，以免说不清是否达到目标。

4）吸取教训。把以前工作中学习到的经验教训运用过来，对确定测试策略很有作用。

5）工作量太小。完成测试需要多少工作量？需要多少人员？

6）技术选择。系统会采取什么技术？系统会采用什么架构？这些信息有助于确定测试策略和测试工具。

7）时间表。系统开发和测试分配的时间有多长？截止日期是什么时候？

125.【简答题】【测试计划】设计系统测试计划需要参考的项目文档有？

软件测试计划、软件需求工件、和迭代计划

126.【简答题】【测试计划】测试计划工作的目的是什么?测试计划文档的内容应该包括什么?其中哪些是最重要的?

软件测试计划是指导测试过程的纲领性文件：

领导能够根据测试计划进行宏观调控，进行相应资源配置等

测试人员能够了解整个项目测试情况以及项目测试不同阶段的所要进行的工作等

便于其他人员了解测试人员的工作内容，进行有关配合工作

包含了产品概述、测试策略、测试方法、测试区域、测试配置、测试周期、测试资源、测试交流、风险分析等内容。借助软件测试计划，参与测试的项目成员，尤其是测试管理人员，可以明确测试任务和测试方法，保持测试实施过程的顺畅沟通，跟踪和控制测试进度，应对测试过程中的各种变更。

测试计划编写 6 要素(5W1H)：

why——为什么要进行这些测试;

what—测试哪些方面，不同阶段的工作内容;

when—测试不同阶段的起止时间;

where—相应文档，缺陷的存放位置，测试环境等;

who—项目有关人员组成，安排哪些测试人员进行测试;

how—如何去做，使用哪些测试工具以及测试方法进行测试

测试计划和测试详细规格、测试用例之间是战略和战术的关系，测试计划主要从宏观上规划测试活动的范围、方法和资源配置，而测试详细规格、测试用例是完成测试任务的具体战术。所以其中最重要的是测试测试策略和测试方法(最好是能先评审)。

127.【简答题】【测试计划】你认为做好测试计划工作的关键是什么？

明确测试的目标，增强测试计划的实用性

编写软件测试计划得重要目的就是使测试过程能够发现更多的软件缺陷，因此软件测试计划的价值取决于它对帮助管理测试项目，并且找出软件潜在的缺陷。因此，软件测试计划中的测试范围必须高度覆盖功能需求，测试方法必须切实可行，测试工具并且具有较高的实用性，便于使用，生成的测试结果直观、准确

坚持“5W”规则，明确内容与过程

“5W”规则指的是“What(做什么)”、“Why(为什么做)”、“When(何时做)”、“Where(在哪里)”、“How(如何做)”。利用“5W”规则创建软件测试计划，可以帮助测试团队理解测试的目的(Why)，明确测试的范围和内容(What)，确定测试的开始和结束日期(When)，指出测试的方法和工具(How)，给出测试文档和软件的存放位置(Where)。

采用评审和更新机制，保证测试计划满足实际需求

测试计划写作完成后，如果没有经过评审，直接发送给测试团队，测试计划内容的可能不准确或遗漏测试内容，或者软件需求变更引起测试范围的增减，而测试计划的内容没有及时更新，误导测试执行人员。

分别创建测试计划与测试详细规格、测试用例

应把详细的测试技术指标包含到独立创建的测试详细规格文档，把用于指导测试小组执行测试过程的测试用例放到独立创建的测试用例文档或测试用例管理数据库中。测试计划和测试详细规格、测试用例之间是战略和战术的关系，测试计划主要从宏观上规划测试活动的范围、方法和资源配置，而测试详细规格、测试用例是完成测试任务的具体战术。

128.【简答题】【测试计划】一份测试计划应该包括哪些内容?

背景、项目简介、目的、测试范围、测试策略、人员分工、资源要求、进度计划、参考文档、常用术语、提交文档、风险分析。

129.【简答题】【测试计划】产品认为测试排期长，需要缩短测试排期怎么办？

如果产品看到测试计划之后反映需求功能简单（从产品层面看需求功能简单），测试时间排期过长，需要缩短测试时间，那就要将测试的整个流程详细的说出，同时将如果不进行充分测试可能带来的后果是什么。最终如果还要决定缩短测试排期的话，可以叫上产品、开发、测试进行会议讨论，将会议记录以及风险写明，抄送三方，以防止出现因缩短测试时间未覆盖到的测试点导致线上问题出现归咎于测试。

130.【简答题】【测试计划】如果遇到开发未提测全部功能，或者部分功能未给排期的情况下，该怎么写测试计划？

如果在写测试计划的时候，全部功能或者部分功能未提测或者未给排期的情况下，需要进行说明，同时将具体的时间写为待定，如果开发有提测或者给与排期之后，进行邮件的更新。

131.【简答题】【测试计划】如果遇到一些功能模块是依靠别的模块上线之后才能进行测试的情况怎么办？

这个时候你可以进行备注说明待定以及待评估的原因。（用不同颜色字体标注效果会更好哦~）

132.【简答题】【测试用例】为什么要编写测试用例

1）理清思路，避免遗漏

复杂的项目需要我们把功能细分，根据每一个功能来编写测试用例，

整理我们的测试系统思路，避免遗漏要测试的功能点

2）跟踪测试进度

通过测试用例执行后的统计结果，方便我们跟踪项目进度

3）回归测试

在不同的测试环境，不同的测试人员在不同的阶段执行相同的测试用例，用回归测试来规范我们的测试行为

4）历史参考

在做项目的各个版本中，有很多功能是相近的，对于这类功能设计的测试用例，

以后遇到类似的功能可作为参考，也是分析缺陷的参考依据

133.【简答题】【测试用例】怎样编写测试用例

1）分析需求文档，得到测试点

需求分析，了解需求的实现背景，需求的合理性，需求的范围，

挖掘需求文档中隐藏的需求，通过需求交底的过程，确定开发的初步实现思路和方法，

列出需求的框架，包含测试范围及各个功能点，测试的场景等，对于需求中的遗漏和

疑问要找产品确认

2）分析测试用例优先级

让测试更有侧重点，一般分为高中低三个优先级

3）细化测试点，变成可执行测试用例

根据测试需求得到的需求框架（这个不应该有遗漏），梳理细化测试点，

根据测试点，细化具体的测试用例，注意各个点的组合情况和反向测试情况，

参考公共测试用例，但不照搬，要思考本次需求自己特有的测试点，把握好测试点细化的度

4）及时更新测试用例

需求会有变动要维护，测试用例也一样需要维护，

在测试评审时，评审人员对你的测试用例是否有测试点，场景遗漏，

测试用例描述模糊，测试结果输出模糊等提出的问题，我们要及时更改

134.【简答题】【测试用例】怎么提高测试用例的编写能力

1）熟悉业务，了解系统

熟悉业务才能更有效的使用系统，对系统越熟悉，越容易发现系统和业务的问题

2）站在用户的角度

客户需要什么，不需要什么，即客户的使用场景，有利于我们更好的挖掘和思考隐含的需求，

至于这个需求该不该做，那是需求人员的职责，这个需求做起来复杂不复杂，那是开发人员的事，

作为测试人员，我们首先要考虑的就是我们所设计的测试用例是不是包含用户常用到的场景以及基本不会用到的场景有哪些

3）多思考，不要被惯性思维和经验束缚

4）多总结

把常用的用例设计误区和一些好的测试用例设计总结并抽象出来以便以后的复用

135.【简答题】【测试缺陷】当开发人员说不是 BUG 时，你如何应付?

开发人员说不是 bug，有 2 种情况，一是需求没有确定，所以我可以这么做，这个时候可以找来产品经理进行确认，需不需要改动，3 方商量确定好后再看要不要改。二是这种情况不可能发生，所以不需要修改，这个时候，我可以先尽可能的说出是 BUG 的依据是什么?如果被用户发现或出了问题，会有什么不良结果?程序员可能会给你很多理由，你可以对他的解释进行反驳。如果还是不行，那我可以给这个问题提出来,跟开发经理和测试经理进行确认,如果要修改就改,如果不要修改就不改。其实有些真的不是 bug，我也只是建议的方式写进 TD 中，如果开发人员不修改也没有大问题。如果确定是 bug 的话，一定要坚持自己的立场，让问题得到最后的确认。

136.【简答题】【测试缺陷】测试缺陷是什么？

软件缺陷（Defect），常常又被叫做 Bug。所谓软件缺陷，即为计算机软件或程序中存在的某种破坏正常运行能力的问题、错误，或者隐藏的功能缺陷。

137.【简答题】【测试缺陷】缺陷的属性有哪些？

缺陷标识(Identifier)是标记某个缺陷的一组符号，每个缺陷必须有一个唯一的标识、缺陷基本信息、缺陷的标题、缺陷严重程度（Severity）是指因缺陷引起的故障对软件产品的影响程度、缺陷优先级（Priority）指缺陷必须被修复的紧急程度；缺陷状态（Status）指缺陷通过一个跟踪修复过程的进展情况、缺陷起源（Origin）指缺陷引起的故障或事件第一次被检测到的状态、缺陷提交人、缺陷提交时间、缺陷所属项目和模块、缺陷处理人、缺陷处理时间、对缺陷处理结果的描述、缺陷验证人、缺陷验证结果描述、缺陷验证时间、缺陷的详细描述、必要的附件等。

138.【简答题】【测试缺陷】缺陷严重等级划分是什么？

A 类【紧急（critical）】导致系统崩溃、死机；出现不可挽救的数据丢失或损坏、内存泄露。

B 类【高（very high）】导致程序模块丢失；软件错误导致数据丢失；用户需求未实现。

C 类【中（high）】影响系统正常运行的缺陷，主要功能出现错误，影响到产品的使用；

D 类【低（medium）】一般性错误或者一些建议性的错误。

139.【简答题】【测试缺陷】A 类 bug 描述

1、 系统崩溃，如应用程序死掉、应用程序异常退出、通讯意外中断或系统进入死循环。

2、 基本功能无法实现或遗漏，如某一应用程序启动不了或关键活动无法运行，关键数据丢失较多。

3、 性能问题，如操作实时失败、数据库读写效率低。

4、 无法正常安装

5、 升级脚本错误，使升级失败。

6、 内存使用错误，如内存泄露、内存溢出、数组越界等。

7、 进程资源不能释放。

140.【简答题】【测试缺陷】B 类 bug 描述

1、 基本功能存在部分问题或次要功能无法实现或遗漏

2、 程序抛出异常信息没有处理，如空指针、通讯异常等。

3、 安装后文件不全、文件错误造成基本功能无法实现。

4、 前后台版本不兼容。

141.【简答题】【测试缺陷】C 类 bug 描述

1、 次要功能存在部分问题

2、 界面存在明显缺陷，设计不友好、不完善

3、 安装时的小问题，或者安装后文件不全、文件错误造成次要功能无法实现。

142.【简答题】【测试缺陷】D 类 bug 描述

1、界面不规范

2、辅助说明描述不清楚

3、输入输出不规范

4、长操作未给用户提示

5、提示窗口文字未采用行业术语

6、建议性的改进要求

143.【简答题】【测试报告】测试报告的编写目的

本测试报告的具体编写目的，指出预期的读者范围。

实例：本测试报告为 XXX 项目的测试报告，目的在于总结测试阶段的测试以及分析测试结果，描述系统是否符合需求（或达到 XXX 功能目标）。预期参考人员包括用户、测试人员、、开发人员、项目管理者、其他质量管理

人员和需要阅读本报告的高层经理。

144.【简答题】【测试报告】测试报告的术语和缩略语

列出设计本系统/项目的专用术语和缩写语约定。对于技术相关的名词和与多义词一定要注明清楚，以便阅读时不会产生歧义。

145.【简答题】【测试报告】测试报告的测试概要

测试的概要介绍，包括测试的一些声明、测试范围、测试目的等等，主要是测试情况简介。（其他测试经理和质量人员关注部分）

146.【简答题】【测试报告】测试报告的测试方法(和工具)

简要介绍测试中采用的方法(和工具)。

提示：主要是黑盒测试，测试方法可以写上测试的重点和采用的测试模式，这样可以一目了然的知道是否遗漏了重要的测试点和关键块。工具为可选项，当使用到测试工具和相关工具时，要说明。注意要注明是自产还是厂商，版本号多少，在测试报告发布后要避免大多工具的版权问题。

测试报告的测试范围（测试用例设计）

简要介绍测试用例的设计方法。例如：等价类划分、边界值、因果图，以及用这类方法(3-4 句)。

提示：如果能够具体对设计进行说明，在其他开发人员、测试经理阅读的时候就容易对你的用例设计有个整体的概念，顺便说一句，在这里写上一些非常规的设计方法也是有利的，至少在没有看到测试结论之前就可以了解到测试经理的设计技术，重点测试部分一定要保证有两种以上不同的用例设计方法。

147.【简答题】【测试报告】测试报告的测试环境与配置

简要介绍测试环境及其配置。

提示：清单如下，如果系统/项目比较大，则用表格方式列出

数据库服务器配置

CPU：

内存：

硬盘：可用空间大小

操作系统：

应用软件：

机器网络名：

局域网地址：

应用服务器配置

…….

客户端配置

…….

对于网络设备和要求也可以使用相应的表格，对于三层架构的，可以根据网络拓扑图列出相关配置。

148.【简答题】【测试报告】测试结果与缺陷分析

整个测试报告中这是最激动人心的部分，这部分主要汇总各种数据并进行度量，度量包括对测试过程的度量和能力评估、对软件产品的质量度量和产品评估。对于不需要过程度量或者相对较小的项目，例如用于验收时提交用户的测试报告、小型项目的测试报告，可省略过程方面的度量部分；而采用了 CMM/ISO 或者其他工程标准过程的，需要提供过程改进建议和参考的测试报告－主要用于公司内部测试改进和缺陷预防机制－则过程度量需要列出。

149.【简答题】【测试报告】测试报告的测试时间

列出测试的跨度和工作量，最好区分测试文档和活动的时间。数据可供过程度量使用。

例如 XXX 子系统/子功能

实际开始时间－实际结束时间

总工时/总工作日

任务 开始时间 结束时间 总计

合计

对于大系统/项目来说最终要统计资源的总投入，必要时要增加成本一栏，以便管理者清楚的知道究竟花费了多少人力去完成测试。

测试类型 人员成本 工具设备 其他费用

总计

在数据汇总时可以统计个人的平均投入时间和总体时间、整体投入平均时间和总体时间，还可以算出每一个功能点所花费的时/人。

用时人员 编写用例 执行测试 总计

合计

这部分用于过程度量的数据包括文档生产率和测试执行率。

生产率人员 用例/编写时间 用例/执行时间 平均

合计

150.【简答题】【测试报告】测试报告的测试版本

给出测试的版本，如果是最终报告，可能要报告测试次数回归测试多少次。列出表格清单则便于知道那个子系统/子模块的测试频度，对于多次回归的子系统/子模块将引起开发者关注。

151.【简答题】【测试报告】测试报告的需求覆盖

需求覆盖率是指经过测试的需求/功能和需求规格说明书中所有需求/功能的比值，通常情况下要达到 100%的目标。

需求/功能（或编号） 测试类型 是否通过 备注

[Y][P][N][N/A]

根据测试结果 ，按编号给出每一测试需求的通过与否结论。P 表示部分通过，N/A 表示不可测试或者用例不适用。实际上，需求跟踪矩阵列出了一一对应的用例情况以避免遗漏，此表作用为传达需求的测试信息以供检查和审核。

需求覆盖率计算 Y 项/需求总数 ×100%

152.【简答题】【测试报告】测试报告的测试覆盖

需求/功能（或编号） 用例个数 执行总数 未执行 未/漏测分析和原因

实际上，测试用例已经记载了预期结果数据，测试缺陷上说明了实测结果数据和与预期结果数据的偏差；因此没有必要对每个编号在此包含更详细的说明的缺陷记录与偏差，列表的目的仅在于更好的查看测试结果。

测试覆盖率计算 执行数/用例总数 ×100%

153.【简答题】【测试报告】测试报告的缺陷汇总

被测系统 系统测试 回归测试 总计

合计

按严重程度

严重 一般 微小

按缺陷类型

用户界面 一致性 功能 算法 接口 文档 用户界面 其他

按功能分布

功能一 功能二 功能三 功能四 功能五 功能六 功能七

最好给出缺陷的饼状图和柱状图以便直观查看。俗话说一图胜千言，图标能够使阅读者迅速获得信息，尤其是各层面管理人员没有时间去逐项阅读文章。

图例

154.【简答题】【测试报告】测试报告的缺陷分析

本部分对上述缺陷和其他收集数据进行综合分析

缺陷综合分析

缺陷发现效率 = 缺陷总数/执行测试用时

可到具体人员得出平均指标

用例质量 = 缺陷总数/测试用例总数 ×100%

缺陷密度 = 缺陷总数/功能点总数

缺陷密度可以得出系统各功能或各需求的缺陷分布情况，开发人员可以在此分析基础上得出那部分功能/需求缺陷最多，从而在今后开发注意避免并注意在实施时予与关注，测试经验表明，测试缺陷越多的部分，其隐藏的缺陷也越多。

测试曲线图

描绘被测系统每工作日/周缺陷数情况，得出缺陷走势和趋向

重要缺陷摘要

缺陷编号 简要描述 分析结果 备注

155.【简答题】【测试报告】测试报告的残留缺陷与未解决问题

残留缺陷

编号：BUG 号

缺陷概要：该缺陷描述的事实

原因分析：如何引起缺陷，缺陷的后果，描述造成软件局限性和其他限制性的原因

预防和改进措施：弥补手段和长期策略

未解决问题

功能/测试类型：

测试结果：与预期结果的偏差

缺陷：具体描述

评价：对这些问题的看法，也就是这些问题如果发出去了会造成什么样的影响

156.【简答题】【测试报告】测试报告的测试结论

1. 测试执行是否充分（可以增加对安全性、可靠性、可维护性和功能性描述）
2. 对测试风险的控制措施和成效
3. 测试目标是否完成
4. 测试是否通过
5. 是否可以进入下一阶段项目目标

157.【简答题】【测试报告】测试报告的建议

1．对系统存在问题的说明，描述测试所揭露的软件缺陷和不足，以及可能给软件实施和运行带来的影响

2．可能存在的潜在缺陷和后续工作

3．对缺陷修改和产品设计的建议

4．对过程改进方面的建议

158.【简答题】【测试报告】测试报告的附录

- 缺陷列表
- 缺陷等级定义标准
- 测试通过标准

159.【简答题】【LoadRunner 负载测试工具、JMeter 压力测试工具】比较负载测试，容量测试和强度测试的区别?

负载测试：在一定的工作负荷下，系统的负荷及响应时间。

强度测试：在一定的负荷条件下，在较长时间跨度内的系统连续运行给系统性能所造成的影响。

容量测试：容量测试目的是通过测试预先分 析出反映软件 系统应用特征的某项指标的极限值（如最大并发用户数、数据库记录数等），系统在其极限值状 态下没有出现任何软件故障或还能保持主要功能正常运行。容量测试 还将确定测试对象在给定时间内能够持续处理的最大负载或工作量。容量测试的目的是使系统承受超额的数据容量来发现它是否能够正确处理。容量测试是面向数据的，并且它的目的是显示系统可以处理目标内确定的数据容量。

160.【简答题】【python 基础】python sorted ，sort 啥区别？

sorted 是个函数，sort 是方法。

sorted(b)的排序是临时的，如果不用变量接受就会被系统回收掉。而 a.sort()则是永久的，此时的 sort()是对象 a 的一个方法。

另外，sorted(b)有个默认的参数，reverse=False，如果使用 reverse=True 则会实现倒序输出。

161.【简答题】【数据库】什么是索引？创建一个

数据库索引其实就是为了使查询数据效率快。

分为了：聚集索引（主键索引），非聚集索引，联合索引；

1.添加 PRIMARY KEY（主键索引）

```
mysql>ALTER TABLE `table_name` ADD PRIMARY KEY ( `column` )
```

2.添加 UNIQUE(唯一索引)

```
mysql>ALTER TABLE `table_name` ADD UNIQUE (
`column`
```

```
)
```

3.添加 INDEX(普通索引)

```
mysql>ALTER TABLE `table_name` ADD INDEX index_name ( `column` )
```

4.添加 FULLTEXT(全文索引)

```
mysql>ALTER TABLE `table_name` ADD FULLTEXT ( `column`)
```

5.添加多列索引

```
mysql>ALTER TABLE `table_name` ADD INDEX index_name ( `column1`, `column2`, `column3` )
```

162.【简答题】【数据库】MySQL 用的引擎

InnoDB MyISAM Memory/Heap BDB Merge Example CSV MaxDB Archive 等

163.【简答题】【数据库】主键、外键的作用，索引的优点与不足?

主键：是表中的唯一标示键。作用：保证实体的完整性;加快数据库的操作速度;增加新的表记录时，数据库会自动检索新记录的主键值，不允许该值与其他表中记录的主键重复;数据库会按主键值的顺序显示记录，如果没有设定主键，则按输入的顺序显示记录。

外键：是主键的从属，表示了两个表之间的联系。作用：使用外键可以避免冗余。

索引的优点： 1、通过创建唯一性的索引，可以保证表中数据的唯一性; 2、加速数据的检索速度; 3、加快表与表之间的连接; 4、在使用分组与排序数据检索时，可以显著检索分组与排序的时间; 5、在查询的过程中使用优化隐藏器，提供系统性能。

缺点： 1、创建索引需要时间，且随着数据量的增加而增加; 2、索引需要占用物理空间;

3、当对表中数据进行修改时，索引也要动态维护，降低了数据的维护速度。

164.【简答题】【安全测试】WEB 安全测试的类型有哪些?

1.跨站脚本（XSS）

2.反射型跨站（Reflected XSS）

3.存储型跨站（Stored XSS）

4.DOM 跨站(DOM-Based XSS)

5.跨站请求伪造(CSRF)

6.SQL 注入

7.XML 注入

8.URL 跳转

9.文件系统跨越

10.系统命令

11.文件上传

12.任意文件下载

13.权限控制

14.访问控制

15.Session Expires

165.【简答题】【HTTP 请求】get/post，除此之外还有什么请求？

1.HEAD

2.PUT

3.PUT

4.CONNECT

5.OPTIONS

6.TRACE

7.PATCH

166.【简答题】【HTTP 请求】https 比较安全，如何实现的

HTTPS 是通过 SSL 协议外壳来确保它的安全性的。那么他主要体现在三个方面：

第一：数据是加密的

SSL 协议是通过非对称秘钥分发的形式来完成必要的协商，然后再通过对称秘钥的加密方式对数据完成加密。

第二：会去验证双方的身份信息

我们的客户端和服务端双方都需要向 CA 机构去申请证书的。在 SSL 握手阶段的时候，我们就会去验证双方证书的是否可信，从而去验证双方的身份。这样就可以防止第三方的冒充了。

第三：保证数据的完整性

每次数据都需要加上 MAC 的摘要并签名，接收的数据和发送的数据的这个摘要是要一样的，这样就表示数据并没有被篡改过。

167.【简答题】【LR】您在从事性能测试工作时，是否使用过一些测试工具？如果有，请试述该工具的工作原理，并以一个具体的工作中的例子描述该工具是如何在实际工作中应用的。

使用过 LoadRunner，该工具能够录制测试人员的操作步骤，然后对这个操作步骤模拟出多个用户来播放出来。

1、Visural User Genertor 创建脚本，选择协议，录制操作，编辑操作。

2、中央控制器（Controller）调度虚拟用户，创建场景，选择脚本，建立虚拟用户，设计 shedual，设置 ip spoofer。

3、运行脚本。分析 shedual。

4、分析测试结果。

您认为性能测试工作的目的是什么？做好性能测试工作的关键是什么？

性能测试工作的目的是检查系统是否满足在需求说明书中规定的性能，性能测试常常需要和强度测试结合起来，并常常要求同时进行软件和硬件的检测。

性能测试主要的关注对象是响应时间，吞吐量，占用内存大小（辅助存储区），处理精度等。

168.【简答题】【bug 管理】在您以往的工作中，一条软件缺陷（或者叫 Bug）记录都包含了哪些内容？如何提交高质量的软件缺陷（Bug）记录？

检测时间，系统环境，硬件环境，严重程度，程式版本，确认人，功能模板，问题描述，详细操作步骤，是否会重现。

问题描述和详细操作步骤要尽可能详细。Bug 应该尽量用书面语，对于严重程度比较高的缺陷要在相同环境下测试一遍。

169.【简答题】【测试文档】测试活动中，如果发现需要文档不完善或者不准确，怎么处理？

要及时的与项目经理进行沟通协调。要在邮件中详细的把不完善不准确的地方描述出来，并提出自己的意见。

170.【简答题】【配置管理】软件配置管理工作开展的情况和认识？

拿到一台裸机过后要安装客户需要的操作系统，并且安装一些所必须的软件。

171.【简答题】【生命周期】软件测试的文档测试应当贯穿于软件生命周期的全过程，其中用户文档是文档测试的重点。那么软件系统的用户文档包括哪些？

用户安装文档、用户配置文档、用户使用手册、联机指导等。

172.【简答题】【文档测试要点】简述软件系统中用户文档的测试要点？

完整性：用户文档中功能的描述要完整的。不能让用户产生疑问。

一致性：用户文档中的功能描述要与实际软件中的功能一致。不能描述过盛。

易使用性：用户文档描述的内容要方便用户阅读并且能够让用户很清楚的知道如何操作。

图表：有的时候用图表描述会很明了。

173.【简答题】【需求】没有产品说明书和需求文档地情况下能够进行黑盒测试吗？

可以。

这个情况下我们就要进行探索性测试，把软件当成用户需求，一步步进行测试。凭借经验判断功能正确与否，有的时候还可以与项目经理、开发人员一起进行交流沟通，从而进行更好的测试。

174.【简答题】【测试风险】软件测试的风险主要体现在哪里？

主要体现在没法完全测试。有些问题可能隐藏在没有测到的地方。这样子就被忽略了。客户使用的时候并不熟悉软件是如何操作的。可能有的时候会误点点出问题。这样子的话我们就要承担很大的风险了。

发现的缺陷越多，说明软件缺陷越多吗？

是的，通常如果发现一个缺陷的话，有的时候会发现很多类似的缺陷，因为由于开发人员的习惯，可能一个地方有错误，另外一个地方就会有相同的错误。

175.【简答题】【测试方法】目前主要的测试用例设计方法是什么？

白盒测试：逻辑覆盖、循环覆盖、基本路径覆盖

黑盒测试：边界值分析法、等价类划分、错误猜测法、因果图法、状态图法、测试大纲法、随机测试、场景法

176.【简答题】【测试方法】黑盒测试和白盒测试是软件测试的两种基本方法，请分别说明各自的优点和缺点!

黑盒测试的优点有：比较简单，不需要了解程序内部的代码及实现;与软件的内部实现无关; 从用户角度出发，能很容易的知道用户会用到哪些功能，会遇到哪些问题;基于软件开发文档，所以也能知道软件实现了文档中的哪些功能;在做软件自动化测试时较为方便。

黑盒测试的缺点有：不可能覆盖所有的代码，覆盖率较低，大概只能达到总代码量的 30%;自动化测试的复用性较低。

白盒测试的优点有：帮助软件测试人员增大代码的覆盖率，提高代码的质量，发现代码中隐 藏的问题。

白盒测试的缺点有：程序运行会有很多不同的路径，不可能测试所有的运行路径;测试基于代码，只能测试开发人员做的对不对，而不能知道设计的正确与否，可能会漏掉一些功能需求;系统庞大时，测试开销会非常大。

177.【简答题】【测试评审】软件的评审一般由哪些人员参加？其目的是什么？

参加人员：客户、项目经理、开发人员、测试人员

目的：查看软件在未正式投入运行前是否还存在问题。对于不同软硬件平台能否正常运行，是否有与客户理解

不一致的地方，同时可以对一些可以改进的地方再多加改进。

178.【简答题】【测试评审】测试用例需要哪些人来评审？

测试组内评审的，因为我们的方案是全体项目组成员(PM/SE 开发和测试)来评审的并且方案里的测试点写到了测试用例标题的程度。我们是项目组全体来评审的额，毕竟测试是保证软件质量的最后一个环节，测试用例是测试执行的依据，所以测试用例十分重要，项目组非常重视测试用例的评审，希望把漏测的降到最低，所以我们的测试用例是项目组全体成员来评审的。

179.【简答题】【测试评审】您以往的工作中是否曾开展过测试用例的评审工作?如果有，请描述测试用例评审的过程和评审的内容。

评审计划->预审->评审;

评审内容主要是测试用例对软件需求的覆盖程度，对于相关边界是否考虑，是否针对复杂流程准备多套测试数据，是否有专门针对非功能性需求的测试。

180.【简答题】【测试对象】软件测试的对象有哪些？

软件测试并不等于程序测试。软件测试应贯穿于软件定义与开发的整个期间。

需求分析、概要设计、详细设计以及程序编码等各阶段所得到的文档，包括需求规格说明、概要设计规格说明、详细设计规格说明以及源程序，都应成为软件测试的对象

181.【简答题】【测试设计】进行测试时产生了哪些文档或记录？

测试的整个过程有系统测试计划、系统测试用例、系统测试报告、缺陷报告、产品发布说明

在执行测试的过程中只有缺陷报告，这个还是用在缺陷管理工具中进行的，最后在工具中导出缺陷报告

182.【简答题】【测试框架】什么是测试方案，什么是测试策略？

测试方案是指导我们怎么测的问题，里面的主要内容是测试点。策略是指导我们要测什么方面，比如要进行功能测试，性能测试，兼容性测试等等，并指出需要依赖与什么工具。

183.【简答题】【测试方案】测试方案包含哪些内容？

业务功能的描述，对需求功能的理解，业务流程图，业务表，测试点等。

184.【简答题】【职业规划】你能不能说下你的 3-5 年的职业规划？

首先，要巩固自己的测试基础知识，在基本知识扎实的情况下提高理解需求文档地能力。

其次，学习自动化测试工具，并将它运用到测试中。

然后，在测试技术达到一定程度后，要学会如何带领一个测试团队。

最后，争取在最快的时间内达到测试经理的水平

185.【简答题】【职业规划】你的测试职业发展是什么？

测试经验越多，测试能力越高。所以我的职业发展是需要时间积累的，一步步向着高级测试工程师奔去。而且我也有初步的职业规划，前 3 年积累测试经验，按如何做好测试工程师的要点去要求自己，不断更新自己改正自己，做好测试任务。

186.【简答题】【职业规划】你的测试职业发展目标是什么？

测试经验越多，测试能力越高。所以我的职业发展是需要时间累积的，一步步向着高级测试工程师奔去。而且我也有初步的职业规划，前 3 年累积测试经验，不断的更新自己改正自己，做好测试任务。

187.【简答题】【职业素养】一个测试工程师应具备那些素质？

1、责任心 2、沟通能力 3、团队合作精神 4、耐心、细心、信心 5、时时保持怀疑态度，并且有缺陷预防的意识 6、具备一定的编程经验

188.【简答题】【测试技能】你认为测试人员需要具备哪些素质？

做测试应该要有一定的协调能力，因为测试人员经常要与开发接触处理一些问题，如果处理不好的话会引起一些冲突，这样的话工作上就会不好做。还有测试人员要有一定的耐心，有的时候做测试很枯燥乏味。除了耐心，测试人员不能放过每一个可能的错误。

189.【简答题】【测试目的】测试的目的是什么？

测试的目的是找出软件产品中的错误，软件尽可能的符合用户的要求。当然软件测试是不可能找出全部错误。

190.【简答题】【生命周期】软件生存周期及其模型是什么？

软件生存周期(Software life cycle)又称为软件生命期，生存期。是指从形成开发软件概念起，所开发的软件使用以后，知道失去使用价值消亡为止的整个过程。一般来说，整个生存周期包括计划(定义)、开发、运行(维护)三个时期，每个时期又划分为若干个阶段。每个阶段有明确的任务。

周期模型(典型的几种)：

瀑布模型

快速原型模型：快速原型模型允许在需求分析阶段对软件的需求进行初步而非完全的分析和定义，快速设计开发出软件系统的原型，该原型向用户展示待开发软件的全部或部分功能和性能;用户对该原型进行测试评定，给出具体改进意见以丰富细化软件需求;开发人员据此对软件进行修改完善，直至用户满意认可之后，进行软件的完整实现及测试、维护。

迭代模型：迭代包括产生产品发布(稳定、可执行的产品版本)的全部开发活动和要使用该发布必需的所有其他外围元素。在某种程度上，开发迭代是一次 完整地经过所有工作流程的过程：需求分析、设计、实施和测试工作流程。实质上，它类似小型的瀑布式项目。RUP 认为，所有的阶段都可以细分为迭代。每一次 的迭代都会产生一个可以发布的产品，这个产品是最终产品的一个子集。

生命周期阶段：

软件计划与可行性分析

需求分析

软件设计

编码

软件测试

运行与维护

191.【简答题】【测试配置】软件配置管理的作用?软件配置包括什么?

软件配置管理(Software Configuration Management，SCM)是一种标识、组织和控制修改的技术。软件配置管理应用于整个软件工程过程。在软件建立时变更是不可避免的，而变更加剧了项目中软件开发者之间的混乱。SCM 活动的目标就是为了标识变更、控制变更、确保变更正确实现并向其他有关人员报告变更。从某种角度讲，SCM 是一种标识、组织和控制修改的技术，目的是使错误降为最小并最有效地提高生产效率。

软件配置包括如下内容：配置项识别、工作空间管理、版本控制、变更控制、状态报告、配置审计

192.【简答题】【测试安全】软件的安全性应从哪几个方面去测试?

软件安全性测试包括程序、数据库安全性测试。根据系统安全指标不同测试策略也不同。

用户认证安全的测试要考虑问题： 明确区分系统中不同用户权限 、系统中会不会出现用户冲突 、系统会不会因用户的权限的改变造成混乱 、用户登陆密码是否是可见、可复制 、是否可以通过绝对途径登陆系统(拷贝用户登陆后的链接直接进入系统)、用户退出系统后是否删除了所有鉴权标记，是否可以使用后退键而不通过输入口令进入 系统 、系统网络安全的测试要考虑问题 、测试采取的防护措施是否正确装配好，有关系统的补丁是否打上 、模拟非授权攻击，看防护系统是否坚固 、采用成熟的网络漏洞检查工具检查系统相关漏洞(即用最专业的黑客攻击工具攻击试一下，现在最常用的是 NBSI 系列和 IPhacker IP) 、采用各种木马检查工具检查系统木马情况 、采用各种防外挂工具检查系统各组程序的外挂漏洞

数据库安全考虑问题： 系统数据是否机密(比如对银行系统，这一点就特别重要，一般的网站就没有太高要求)、系统数据的完整性(我刚刚结束的企业实名核查服务系统中就曾存在数据 的不完整，对于这个系统的功能实现

有了障碍) 、系统数据可管理性 、系统数据的独立性 、系统数据可备份和恢复能力(数据备份是否完整，可否恢复，恢复是否可以完整)

193.【简答题】【测试质量】什么是软件质量?

概括地说，软件质量就是“软件与明确的和隐含的定义的需求相一致的程度”。具体地说，软件质量是软件符合明确叙述的功能和性能需求、文档中明确描述 的开发标准、以及所有专业开发的软件都应具有的隐含特征的程度。 影响软件质量的主要因素，这些因素是从管理角度对软件质量的度量。可划分为三组，分别反应用户在使用软件产品时的三种观点。正确性、健壮性、效率、完整性、可用性、风险(产品运行);可理解性、可维修性、灵活性、可测试性(产品修改);可移植性、可再用性、互运行性(产品转移)。

194.【简答题】【测试质量】软件质量保证体系是什么 国家标准中与质量保证管理相关的几个标准是什么?他们的编号和全称是什么?

SQA 由一套软件工程过程和方法组成，以保证(软件的)质量。SQA 贯穿整个软件开发过程，(它)应包括需求文档评审、代码控制、代码评审、变更管理、配置管理、版本管理和软件测试。

软件质量保证(SQA-Software Quality Assurance)是建立一套有计划，有系统的方法，来向管理层保证拟定出的标准、步骤、实践和方法能够正确地被所有项目所采用。软件质量保证的目的是使软件过程对于管理人员来说是可见的。它通过对软件产品和活动进行评审和审计来验证软件是合乎标准的。软件质量保证组在项目开始时就一起参与建立计划、标准和过程。这些将使软件项目满足机构方针的要求。

195.【简答题】【测试质量】你觉得测试和开发需要怎么结合才能使软件的质量得到更好的保障

测试和开发应该按照 W 模型的方式进行结合，测试和开发同步进行，能够尽早发现软件缺陷，降低软件开发的成本。

在 V 模型中，测试过程被加在开发过程的后半部分，单元测试所检测代码的开发是否符合详细设计的要求。集成测试所检测此前测试过的各组成部分是否能完好地结合到一起。系统测试所检测已集成在一起的产品是否符合系统规格说明书的要求。而验收测试则检测产品是否符合最终用户的需求。V 模型的缺陷在于仅仅把测试过程作为在需求分析、系统设计及编码之后的一个阶段，忽视了测试对需求分析、系统设计的验证，因此需求阶段的缺陷很可能一直到后期的验收测试才被发现，此时进行弥补将耗费大量人力物力资源。

相对于 V 模型，W 模型增加了软件各开发阶段中应同步进行的验证和确认活动。W 模型由两个 V 字型模型组成，分别代表测试与开发过程，图中明确表示出了测试与开发的并行关系。

W 模型强调：测试伴随着整个软件开发周期，而且测试的对象不仅仅是程序，需求、设计等同样要测试，也就是说，测试与开发是同步进行的。W 模型有利于尽早地全面的发现问题。例如，需求分析完成后，测试人员就应该参与到对需求的验证和确认活动中，以尽早地找出缺陷所在。同时，对需求的测试也有利于及时了解项目难度和测

试风险，及早制定应对措施，这将显著减少总体测试时间，加快项目进度。

W 模型中测试的活动与软件开发同步进行，测试的对象不仅仅是程序，还包括需求和设计，因此能够尽早发现软件缺陷，降低软件开发的成本。

196.【简答题】【测试特性】软件产品质量特性是什么？

功能性：适应性、准确性、互操作性、依从性、安全性。

可靠性：成熟性、容错性、易恢复性。

可使用性：易理解性、易学习性、易操作性。

效率：时间特性、资源特性。

可维护性：易分析性、易变更性、稳定性、易测试性。

可移植性： 适应性、易安装性、遵循性、易替换性

197.【简答题】【测试阶段】软件测试各个阶段通常完成什么工作?各个阶段的结果文件是什么?包括什么内容?

单元测试阶段：各独立单元模块在与系统地其他部分相隔离的情况下进行测试，单元测试针对每一个程序模块进行正确性校验，检查各个程序模块是否正确地实现了规定的功能。生成单元测试报告，提交缺陷报告。

集成测试阶段：集成测试是在单元测试的基础上，测试在将所有的软件单元按照概要设计规格说明的要求组装成模块、子系统或系统的过程中各部分工作是否达到或实现相应技术指标及要求的活动。该阶段生成集成测试报告，提交缺陷报告。

系统测试阶段：将通过确认测试的软件，作为整个给予计算机系统的一个元素，与计算机硬件、外设、某些支持软件、数据和人员等其他系统元素结合在一起，在实际运行环境下，对计算机系统进行全面的功能覆盖。该阶段需要提交测试总结和缺陷报告。

198.【简答题】【测试阶段】测试分为哪几个阶段？

一般来说分为 5 个阶段：单元测试、集成测试、确认测试、系统测试、验收测试。

199.【简答题】【测试任务】测试人员在软件开发过程中的任务是什么？

1、尽可能早的找出系统中的 Bug;

2、避免软件开发过程中缺陷的出现;

3、衡量软件的品质，保证系统的质量;

4、关注用户的需求，并保证系统符合用户需求。

总的目标是：确保软件的质量。

200.【简答题】【测试策略】软件测试的策略是什么？

软件测试策略：在一定的软件测试标准、测试规范的指导下，依据测试项目的特定环境约束而规定的软件测试的原则、方式、方法的集合。

201.【简答题】【测试策略】系统测试的策略有哪些种类

有性能测试、负载测试、强度测试、易用性测试、安全测试、配置测试、安装测试、文档测试、故障恢复测试、用户界面测试、恢复测试、分布测试、可用性测试。

202.【简答题】【测试类型】简述什么是静态测试、动态测试、黑盒测试、白盒测试、α测试 β 测试

静态测试是不运行程序本身而寻找程序代码中可能存在的错误或评估程序代码的过程。

动态测试是实际运行被测程序，输入相应的测试实例，检查运行结果与预期结果的差异，判定执行结果是否符合要求，从而检验程序的正确性、可靠性和有效性，并分析系统运行效率和健壮性等性能。

黑盒测试一般用来确认软件功能的正确性和可操作性,目的是检测软件的各个功能是否能得以实现,把被测试的程序当作一个黑盒,不考虑其内部结构,在知道该程序的输入和输出之间的关系或程序功能的情况下,依靠软件规格说明书来确定测试用例和推断测试结果的正确性。

白盒测试根据软件内部的逻辑结构分析来进行测试,是基于代码的测试，测试人员通过阅读程序代码或者通过使用开发工具中的单步调试来判断软件的质量，一般黑盒测试由项目经理在程序员开发中来实现。

α 测试是由一个用户在开发环境下进行的测试，也可以是公司内部的用户在模拟实际操作环境下进行的受控测试，Alpha 测试不能由程序员或测试员完成。

β 测试是软件的多个用户在一个或多个用户的实际使用环境下进行的测试。开发者通常不在测试现场，Beta 测试不能由程序员或测试员完成。

203.【简答题】【测试标准】测试结束的标准是什么？

从微观上来说，在测试计划中定义，比如系统在一定性能下平稳运行 72 小时，目前 Bug Tracking System 中，本版本中没有一般严重的 BUG，普通 BUG 的数量在 3 以下，BUG 修复率 90%以上等等参数，然后由开发经理，测试经理，项目经理共同签字认同版本 Release。

如果说宏观的，则是当这个软件彻底的消失以后，测试就结束了。

204.【简答题】【测试标准】你觉得软件测试通过的标准应该是什么样的？

测试用例完全执行，测试用例覆盖到所有的测试点，并且缺陷的密度达到客户的需求。

205.【简答题】【测试兴趣】你对测试最大的兴趣在哪里?为什么?

最大的兴趣就是测试有难度，有挑战性!做测试越久越能感觉到做好测试有多难。曾经在无忧测试网上看到一篇文章，是关于如何做好一名测试工程师。一共罗列了 11，12 点，有部分是和人的性格有关，有部分需要后天的努力。但除了性格有关的 1，2 点我没有把握，其他点我都很有信心做好它。

刚开始进入测试行业时，对测试的认识是从无忧测试网上了解到的一些资料，当时是冲着做测试需要很多技能才能做的好，虽然入门容易，但做好很难，比开发更难，虽然当时我很想做开发(学校专业课我基本上不缺席，因为我喜欢我的专业)，但看到测试比开发更难更有挑战性，想做好测试的意志就更坚定了。

我觉得做测试整个过程中有 2 点让我觉得很有难度(对我来说，有难度的东西我就非常感兴趣)，第一是测试用例的设计，因为测试的精华就在测试用例的设计上了，要在版本出来之前，把用例写好，用什么测试方法写?(也就是测试计划或测试策略)，如果你刚测试一个新任务时，你得花一定的时间去消化业务需求和技术基础，业务需求很好理解(多和产品经理和开发人员沟通就能达到目的)，而技术基础可就没那么简单了，这需要你自觉的学习能力，比如说网站吧，最基本的技术知识你要知道网站内部是怎么运作的的，后台是怎么响应用户请求的?测试环境如何搭建?这些都需要最早的学好。至少在开始测试之前能做好基本的准备，可能会遇到什么难题?需求细节是不是没有确定好?这些问题都能在设计用例的时候发现。

第二是发现 BUG 的时候了，这应该是测试人员最基本的任务了，一般按测试用例开始测试就能发现大部分的 bug，还有一部分 bug 需要测试的过程中更了解所测版本的情况获得更多信息，补充测试用例，测试出 bug。还有如何发现 bug?这就需要在测试用例有效的情况下，通过细心和耐心去发现 bug 了，每个用例都有可能发现 bug，每个地方都有可能出错，所以测试过程中思维要清晰(测试过程数据流及结果都得看仔细了，bug 都在里面发现的)。如何描述 bug 也很有讲究，bug 在什么情况下会产生，如果条件变化一点点，就不会有这个 bug，以哪些最少的操作步骤就能重现这个 bug，这个 bug 产生的规律是什么?如果你够厉害的话，可以帮开发人员初步定位问题。

206.【简答题】【测试兴趣】你对测试最大的兴趣在哪里？为什么？

最大的兴趣就是具有挑战性。

因为我并不知道哪里会出现 bug，在找到一个 bug 后会很高兴。并且测试需要很强的耐心和细心。我可以很容易的找到一些细节问题。

207.【简答题】【测试兼容】什么是兼容性测试?请举例说明如何利用兼容性测试列表进行测试。

主要验证软件产品在不同版本之间的兼容性。包括向下兼容和交错兼容，向下兼容是测试软件新版本保留它早期版本功能的情况，交错兼容是验证共同存在的两个相关但不相同的产品之间的兼容性。

208.【简答题】【测试兼容】对某软件进行测试，发现在 WIN98 上运行得很慢，怎么判别是该软件存在问题还是其软硬件运行环境存在问题？

看软件的运行环境要求。如果符合要求则是程序存在问题，若不符合要求则是硬件系统存在问题

209.【简答题】【测试开发】请问测试开发需要哪些知识？需要具备什么能力？

需要的知识：

软件测试基础理论知识，如黑盒测试、白盒测试等；

考编程语言基础，如 C/C++、java、python 等；

自动化测试工具，如 Selenium、Appium、Robotium 等；

计算机基础知识，如数据库、Linux、计算机网络等；

测试框架，如 JUnit 等。

需要具备的能力：

业务分析能力，分析整体业务流程、分析被测业务数据、分析被测系统架构、分析被测业务模块、分析测试所需资源、分析测试完成目标；

缺陷洞察能力，一般缺陷的发现能力、隐性问题的发现能力、发现连带问题的能力、发现问题隐患的能力、尽早发现问题的能力、发现问题根源的能力；

团队协作能力，合理进行人员分工、协助组员解决问题、配合完成测试任务、配合开发重现缺陷、督促项目整体进度、出现问题勇于承担；

专业技术能力，掌握测试基础知识、掌握计算机知识、熟练运用测试工具；

逻辑思考能力，判断逻辑的正确性、对可行性逻辑分析、站在客观角度思考；

问题解决能力，技术上的问题、工作中的问题、沟通问题；

沟通表达能力，和技术人员、产品人员、上下级的沟通；

宏观把控能力，有效控制测试时间、有效控制测试成本、有效制定测试计划、有效进行风险评估、有效控制测试方向。

210.【简答题】【测试开发】你觉得自动化测试有什么意义，都需要做些什么

自动化测试的意义在于

1、可以对程序的新版本自动执行回归测试

2、可以执行手工测试困难或者不可能实现的测试，如压力测试，并发测试，

3、能够更好的利用资源，节省时间和人力

执行自动化测试之前首先判断这个项目是不是和推广自动化测试，然后对项目做需求分析，指定测试计划，搭建自动化测试框架，设计测试用例，执行测试，评估

211.【简答题】【测试背景】针对于软件的行业背景，你如何理解软件的业务?

阅读用户手册了解软件的功能和操作流程;看一些业务的专业书籍补充业务知识;如果有用户实际的数据，可以拿实际的数据进行参考;参考以前的用例和 BUG 报告;在使用软件的过程中多思考;多与产品经理交流。

212.【简答题】【测试目的】您认为性能测试工作的目的是什么?做好性能测试工作的关键是什么?

关键是测试脚本的录制，测试时候测试环境的干净。

213.【简答题】【测试环境】您如何看待软件过程改进?在您曾经工作过的企业中，是否有一些需要改进的东西呢?您期望的理想的测试人员的工作环境是怎样的?

将先进的经验或思想固化到过程中，通过过程改进和能力提高来改进软件质量。

214.【简答题】【测试前景】请问你怎么看待软件测试的潜力和挑战

软件测试是正在快速发展，充满挑战的领域。尽管现在许多自动化测试软件的出现使得传统手工测试的方式被代替，但自动化测试工具的开发、安全测试、测试建模、精准测试、性能测试、可靠性测试等专项测试中仍然需要大量具有专业技能与专业素养的测试人员，并且随着云计算、物联网、大数据的发展，传统的测试技术可能不再适用，测试人员也因此面临着挑战，需要深入了解新场景并针对不同场景尝试新的测试方法，同时敏捷测试、Devops 的出现也显示了软件测试的潜力。

215.【简答题】【生命周期】您是否了解以往所工作的企业的软件开发过程?如果了解，请试述一个完整的开发过程需要完成哪些工作?分别由哪些不同的角色来完成这些工作?您在以往的测试工作中都曾经具体从事过哪些工作?其中最擅长哪部分工作?

开发过程---需求调研(需求人员)、需求分析(需求人员)、概要设计(设计人员)、详细设计(设计人员)、编码(开发人员)

测试过程---需求评审、系统测试设计、概要设计评审、集成测试设计、详细设计评审、单元测试设计、测试执行

测试工作的整个过程都做过，擅长做测试设计

过程决定质量，软件的过程改进正是为了提高软件的质量，将过往的种种经验教训积累起来。

216.【简答题】【单元测试】你觉得单元测试可行吗

可行，单元测试可以有效地测试某个程序模块的行为，是未来重构代码的信心保证。事前可以保证质量，事后可以快速复现问题，并在修改代码后做回归自测。可行性考虑的是要用一些可行的方法做到关键的代码可测试，如通过边界条件、等价类划分、错误、因果，设计测试用例要覆盖常用的输入组合、边界条件和异常。

217.【简答题】【测试脚本】请问你有没有写过测试脚本，怎么写的？

然后，撰写测试桩与驱动，白盒测试保证代码逻辑中循环和分支都能够走到，黑盒测试保证函数和首先，代码走查结合动态单步跟踪以及观察日志与文件输出，网络、CPU 状态。

功能脚本接口正确，输入输出符合设计预期。

对于异常处理，特别是变量的检查需要特别关注，变量在使用前都需要进行检查，是否为空？或者为 0？对于文件名和路径必须检查，确认文件是否存在，路径是否可达之后再进行后续操作。

另外，需要考虑所依赖的其他功能脚本以及二进制工具，这些功能性单元应该如何使用，调用后的返回会有哪些情况，对于正常和异常结果，脚本是否能够捕捉到并且作出正确的判断。

218.【简答题】【测试工具】请问你有用过什么测试工具吗，用过哪些？

自动化测试工具用过 selenium 和 appium

性能测试工具有用过 Jmeter

219.【简答题】【web 测试】请问你有没有写过 web 测试，怎么写的？

Web 测试主要从下面几个大方向考虑

功能测试，主要做链接测试，表单测试，cookies 测试，设计语言测试等

性能测试，考虑连接速度测试，以及负载测试，例如：Web 应用系统能允许多少个用户同时在线？如果超过了这个数量，会出现什么现象？Web 应用系统能否处理大量用户对同一个页面的请求？还有压力测试

可用性测试，比如导航测试，图形测试，内容测试，整体界面测试等

兼容性测试，市场上有很多不同的操作系统类型，最常见的有 Windows、Unix、Macintosh、Linux 等。Web 应用系统的最终用户究竟使用哪一种操作系统，取决于用户系统的配置。这样，就可能会发生兼容性问题，同一个应用可能在某些操作系统下能正常运行，但在另外的操作系统下可能会运行失败。因此，在 Web 系统发布之前，需要在各种操作系统下对 Web 系统进行兼容性测试。

安全性测试，

（1）现在的 Web 应用系统基本采用先注册，后登陆的方式。因此，必须测试有效和无效的用户名和密码，要注意到是否大小写敏感，可以试多少次的限制，是否可以不登陆而直接浏览某个页面等。

（2）Web 应用系统是否有超时的限制，也就是说，用户登陆后在一定时间内（例如 15 分钟）没有点击任何页面，是否需要重新登陆才能正常使用。

（3）为了保证 Web 应用系统的安全性，日志文件是至关重要的。需要测试相关信息是否写进了日志文件、是否可追踪。

（4）当使用了安全套接字时，还要测试加密是否正确，检查信息的完整性。

（5）服务器端的脚本常常构成安全漏洞，这些漏洞又常常被黑客利用。所以，还要测试没有经过授权，就不能在服务器端放置和编辑脚本的问题。

220.【多选题】【CSS】小李测试一款新开发的手机 APP 应用界面，那么，属于界面元素测试内容的是：（ ）？

A、 文字测试
B、 菜单测试
C、 窗口测试
D、 功能点测试

答案：A，B、C。

解析：界面元素测试包括：窗口测试、菜单测试、图标测试、文字测试、鼠标测试

221.【简答题】【头脑风暴】5 只猫 五分钟捉 5 只老鼠 请问 100 分钟捉 100 只老鼠需要多少只猫？

5*X*5M=5Y

K*X*100M=100Y

K=5

需要 5 只猫

222.【简答题】【头脑风暴】圆桌，两个人，轮流放硬币，不能重叠，半径为 1，某一方不能放下去，则为输。问先手赢 后手赢。

先手赢。圆桌对称，先手放一个，后手都能找到对称的位置放，但是除了圆心。

223.【简答题】【头脑风暴】逻辑题：3 升的杯子一个，5 升的杯子一个，杯子不规则形状 问怎么得到 4 升的水 水无限多

1、将 3 升的装满倒入 5 升的；

2、再一次将 3 升的转满，倒入 5 升的，把 5 升装满；

3、3 升杯里剩下的就是 1 升水；

4、倒掉 5 升的，把 1 升水倒入 5 升杯；

5、第三次加满 3 升杯，倒入 5 升杯，得到 4 升水。

224.【简答题】【头脑风暴】晚上有四个人过桥，一次只能过两个人，但是只有一只手电筒，四个人过桥时间分别是 1，2，5，8，求最短过桥时间

假设这四人依次是甲乙丙丁：首先甲和乙过桥,甲带手电筒回来；然后丙和丁过桥,由乙带手电筒回来；最后甲再和乙一起过桥.所以最少用时间是 2+1+8+2+2=15（分钟）

225.【简答题】【头脑风暴】两个容积分别为 5 升和 6 升的桶，最后如何只装 3 升？

第一步：先取来 6 升水，倒进 5 升桶的水桶里，即得到 6 升桶里余下的 1 升水；

第二步：把 5L 桶清掉，把取到的 1 升水放进 5 升的水桶里保留不动，然后再取 6 升水，倒进 5 升的水桶里，6 升的桶得到的是 2 升水，把 5L 桶清掉，存放这 2 升水；

第三步：5 升水桶有 2 升水．再取 6 升水，倒进 5 升水桶里，原有 2L 升+3 升=5 升，这时 6 升-3 升=3 升，6 升里余下的就是 3 升水了。

226.【简答题】【头脑风暴】有十张扑克牌，每次可以只出一张，也可以只出两张，要出完有多少种出法

还有一张牌就出完 10 张，可能的情况有两种，从 9 到 10 和从 8 到 10，已知了从 0 到 9 的出法有 N 种，如果再知道从 0 到 8 的出法有 P 种，那么从 0 到 10 级的出法就是 N+P，那么可得出：

F(9)=N；

F(8)=P；

F(10)=N+P；

F(10)=F(9)+F(8);

又有：

F(1)=1;

F(2)=2;

最后推出：F(10)=89

227.【简答题】【头脑风暴】井盖为什么是圆的

井道大都是圆形的，所以井盖就做成圆形的。

很多井道都是圆形的，所以井盖自然也就是圆形的了。那为什么井大都是圆形的呢？因为建筑学和土木工程学

中，圆形通道最有利于保持土壤的压力。

圆的受力更均匀不容易碎裂和塌陷

圆形井盖受力后，会向四周扩散压力，由于扩散均匀不容易碎裂和塌陷。

矩形的井盖由于受力不均匀，导致碎裂的几率远大于圆形。所以通过耐用性方面考虑还是圆形井盖合适。

圆形井盖从任何方向都不会掉落井下，也方便操作

矩形对角线的长度都大于矩形的长和宽。所以在对角线方向把井盖竖起来就容易掉落井下。

相对节省生成材料成本

相对于矩形或者正方形，矩形内切圆形的面积最小，生成用的材料也更少。

228.【简答题】【头脑风暴】用 5L 和 6L 的桶，没有刻度，怎么量出 3L 的水

1.先将 6L 杯装满水放,然后将 6L 杯里的水倒满 5L 杯,此时 5L 杯水满为 5L 水,6L 杯剩 1L 水；

2.清空 5L 杯,将 6L 杯的 1L 水放到 5L 杯,此时 5L 杯有 1L 水,6L 杯是空的；

3.将 6L 杯装满水,然后往 5L 杯倒,刚好倒满 5L 杯（原有 1L+6L 杯倒过来的 4L=5L）的时候,6L 杯里还有 2L 水。

4.清空 5L 杯里的水,将 6L 杯的 2L 水倒到 5L 杯,此时 5L 杯里有 2L 水,6L 杯为空；

5.将 6L 杯装满水,然后往 5L 杯里倒,刚好倒满 5L 杯（5L 杯原来有 2L+6L 杯倒过来的 3L=5L 刚好满）时,6L 杯里剩下的水就是 3L.

229.【简答题】【网络基础】tcp 如何实现可靠传输

TCP 传输控制协议是主机对主机层的传输控制协议，提供可靠的连接服务，采用三次握手、四次挥手建立一个传输。

230.【简答题】【网络基础】请你说一下 HTTP 的报文段是什么样的

1、请求方法

GET：请求获取 Request——URL 所标识的资源

POST：在 Request——URL 所标识的资源后附加资源

HEAD：请求获取由 Request——URL 所标识的资源的响应消息报头

PUT：请求服务器存储一个资源，由 Request——URL 作为其标识

DELETE：请求服务器删除由 Request——URL 所标识的资源

TRACE：请求服务器回送收到的请求信息（用于测试和诊断）

CONNECT：保留

OPTIONS：请求查询服务器性能

2、URL

URI 全名为 Uniform Resource Indentifier（统一资源标识），用来唯一的标识一个资源，是一个通用的概念，URI

由两个主要的子集 URL 和 URN 组成。URL 全名为 Uniform Resource Locator（统一资源定位），通过描述资源的位置来标识资源。URN 全名为 Uniform Resource Name（统一资源命名），通过资源的名字来标识资源，与其所处的位置无关，这样即使资源的位置发生变动，其 URN 也不会变化。

3、协议版本

格式为 HTTP/主版本号.次版本号，常用为：HTTP/1.1 HTTP/1.0

4、请求头部

Host：接受请求的服务器地址，可以是 IP 或者是域名

User-Agent：发送请求的应用名称

Connection：指定与连接相关的属性，例如（Keep_Alive，长连接）

Accept-Charset：通知服务器端可以发送的编码格式

Accept-Encoding：通知服务器端可以发送的数据压缩格式

Accept-Language：通知服务器端可以发送的语言

1、协议版本，同请求报文

2、状态码，100~199 表示请求已收到继续处理，200~299 表示成功，300~399 表示资源重定向，400~499 表示客户端请求出错，500~599 表示服务器端出错

200：响应成功

302：跳转，重定向

400：客户端有语法错误

403：服务器拒绝提供服务

404：请求资源不存在

500：服务器内部错误

3、响应头部

Server：服务器应用软件的名称和版本

Content-Type：响应正文的类型

Content-Length：响应正文的长度

Content-Charset：响应正文所使用的编码

Content-Encoding：响应正文使用的数据压缩格式

Content-Language：响应正文使用的语言

231.【简答题】【网络基础】请你回答一下 HTTP 用的什么连接

在 HTTP/1.0 中，默认使用的是短连接。也就是说，浏览器和服务器每进行一次 HTTP 操作，就建立一次连接，但任务结束就中断连接。如果客户端浏览器访问的某个 HTML 或其他类型的 Web 页中包含有其他的 Web 资源，如 JavaScript 文件、图像文件、CSS 文件等；当浏览器每遇到这样一个 Web 资源，就会建立一个 HTTP 会话。

但从 HTTP/1.1 起，默认使用长连接，用以保持连接特性。使用长连接的 HTTP 协议，会在响应头有加入这行

代码：Connection:keep-alive

在使用长连接的情况下，当一个网页打开完成后，客户端和服务器之间用于传输 HTTP 数据的 TCP 连接不会关闭，如果客户端再次访问这个服务器上的网页，会继续使用这一条已经建立的连接。Keep-Alive 不会永久保持连接，它有一个保持时间，可以在不同的服务器软件（如 Apache）中设定这个时间。实现长连接要客户端和服务端都支持长连接。

232.【简答题】【网络基础】请你说一说 TCP 的三次握手

第一次握手：建立连接时，客户端发送 syn 包（syn=j）到服务器，并进入 SYN_SENT 状态，等待服务器确认；SYN：同步序列编号（Synchronize Sequence Numbers）。

第二次握手：服务器收到 syn 包，必须确认客户的 SYN（ack=j+1），同时自己也发送一个 SYN 包（syn=k），即 SYN+ACK 包，此时服务器进入 SYN_RECV 状态；

第三次握手：客户端收到服务器的 SYN+ACK 包，向服务器发送确认包 ACK(ack=k+1)，此包发送完毕，客户端和服务器进入 ESTABLISHED（TCP 连接成功）状态，完成三次握手。

233.【简答题】【网络基础】请你说一下在浏览器中输入一个网址它的运行过程是怎样的

1、查询 DNS，获取域名对应的 IP。

（1）检查浏览器缓存、检查本地 hosts 文件是否有这个网址的映射，如果有，就调用这个 IP 地址映射，解析完成。

（2）如果没有，则查找本地 DNS 解析器缓存是否有这个网址的映射，如果有，返回映射，解析完成。

（3）如果没有，则查找填写或分配的首选 DNS 服务器，称为本地 DNS 服务器。服务器接收到查询时：

如果要查询的域名包含在本地配置区域资源中，返回解析结果，查询结束，此解析具有权威性。

如果要查询的域名不由本地 DNS 服务器区域解析，但服务器缓存了此网址的映射关系，返回解析结果，查询结束，此解析不具有权威性。

（4）如果本地 DNS 服务器也失效：

如果未采用转发模式（迭代），本地 DNS 就把请求发至 13 台根 DNS，根 DNS 服务器收到请求后，会判断这个域名（如.com）是谁来授权管理，并返回一个负责该顶级域名服务器的 IP，本地 DNS 服务器收到顶级域名服务器 IP 信息后，继续向该顶级域名服务器 IP 发送请求，该服务器如果无法解析，则会找到负责这个域名的下一级 DNS 服务器（如 http://baidu.com）的 IP 给本地 DNS 服务器，循环往复直至查询到映射，将解析结果返回本地 DNS 服务器，再由本地 DNS 服务器返回解析结果，查询完成。

如果采用转发模式（递归），则此 DNS 服务器就会把请求转发至上一级 DNS 服务器，如果上一级 DNS 服务器不能解析，则继续向上请求。最终将解析结果依次返回本地 DNS 服务器，本地 DNS 服务器再返回给客户机，查询完成。

2、得到目标服务器的 IP 地址及端口号（http 80 端口，https 443 端口），会调用系统库函数 socket，请求一个 TCP 流套接字。客户端向服务器发送 HTTP 请求报文：

（1）应用层：客户端发送 HTTP 请求报文。

（2）传输层：（加入源端口、目的端口）建立连接。实际发送数据之前，三次握手客户端和服务器建立起一个 TCP 连接。

（3）网络层：（加入 IP 头）路由寻址。

（4）数据链路层：（加入 frame 头）传输数据。

（5）物理层：物理传输 bit。

3、服务器端经过物理层→数据链路层→网络层→传输层→应用层，解析请求报文，发送 HTTP 响应报文。

4、关闭连接，TCP 四次挥手。

5、客户端解析 HTTP 响应报文，浏览器开始显示 HTML

234.【简答题】【网络基础】请你说一说 http rest

REST(Representational State Transfer)一种轻量级的 Web Service 架构。可以完全通过 HTTP 协议实现。其实现和操作比 SOAP 和 XML-RPC 更为简洁，还可以利用缓存 Cache 来提高响应速度，性能、效率和易用性上都优于 SOAP 协议。REST 架构对资源的操作包括获取、创建、修改和删除资源的操作对应 HTTP 协议提供的 GET、POST、PUT 和 DELETE 方法。REST 提供了一组架构约束，当作为一个整体来应用时，强调组件交互的可伸缩性、接口的通用性、组件的独立部署、以及用来减少交互延迟、增强安全性、封装遗留系统的中间组件。

REST 架构约束：

客户-服务器（Client-Server），提供服务的服务器和使用服务的客户需要被隔离对待，客户和服务器之间通过一个统一的接口来互相通讯。

无状态（Stateless），服务端并不会保存有关客户的任何状态，客户端自身负责用户状态的维持，并在每次发送请求时都需要提供足够的信息。

可缓存（Cachable），REST 系统需要能够恰当地缓存请求，以尽量减少服务端和客户端之间的信息传输，以提高性能。

分层系统（Layered System），服务器和客户之间的通信必须被这样标准化：允许服务器和客户之间的中间层（Ross：代理，网关等）可以代替服务器对客户的请求进行回应，而且这些对客户来说不需要特别支持。

统一接口（Uniform Interface），客户和服务器之间通信的方法必须是统一化的。

235.【简答题】【网络基础】请你说一说 http 请求报文

http 请求报文：

1、请求方法

GET：请求获取 Request——URL 所标识的资源

POST：在 Request——URL 所标识的资源后附加资源

HEAD：请求获取由 Request——URL 所标识的资源的响应消息报头

PUT：请求服务器存储一个资源，由 Request——URL 作为其标识

DELETE：请求服务器删除由 Request——URL 所标识的资源

TRACE：请求服务器回送收到的请求信息（用于测试和诊断）

CONNECT：保留

OPTIONS：请求查询服务器性能

2、URL

URI 全名为 Uniform Resource Indentifier（统一资源标识），用来唯一的标识一个资源，是一个通用的概念，URI 由两个主要的子集 URL 和 URN 组成。URL 全名为 Uniform Resource Locator（统一资源定位），通过描述资源的位置来标识资源。URN 全名为 Uniform Resource Name（统一资源命名），通过资源的名字来标识资源，与其所处的位置无关，这样即使资源的位置发生变动，其 URN 也不会变化。

3、协议版本

格式为 HTTP/主版本号.次版本号，常用为：HTTP/1.1 HTTP/1.0

4、请求头部

Host：接受请求的服务器地址，可以是 IP 或者是域名

User-Agent：发送请求的应用名称

Connection：指定与连接相关的属性，例如（Keep_Alive，长连接）

Accept-Charset：通知服务器端可以发送的编码格式

Accept-Encoding：通知服务器端可以发送的数据压缩格式

Accept-Language：通知服务器端可以发送的语言

236.【简答题】【网络基础】请你说一说 get 和 post 区别

GET：从指定的资源请求数据。

POST：向指定的资源提交要被处理的数据。

237.【简答题】【网络基础】请你说一下 tcp 和 udp 的区别

1、TCP 面向连接（如打电话要先拨号建立连接）;UDP 是无连接的，即发送数据之前不需要建立连接

2、TCP 提供可靠的服务。也就是说，通过 TCP 连接传送的数据，无差错，不丢失，不重复，且按序到达;UDP 尽最大努力交付，即不保 证可靠交付

3、TCP 面向字节流，实际上是 TCP 把数据看成一连串无结构的字节流;UDP 是面向报文的，应用层交给 UDP 多长的报文，UDP 就照样发送，即一次发送一个报文。UDP 没有拥塞控制，因此网络出现拥塞不会使源主机的发送速率降低（对实时应用很有用，如 IP 电话，实时视频会议等）

4、每一条 TCP 连接只能是点到点的;UDP 支持一对一，一对多，多对一和多对多的交互通信

5、TCP 首部开销 20 字节;UDP 的首部开销小，只有 8 个字节

6、TCP 的逻辑通信信道是全双工的可靠信道，UDP 则是不可靠信道

238.【简答题】【网络基础】请你说一下为什么 tcp 可靠，哪些方法保证可靠

[1] 确认和重传机制

建立连接时三次握手同步双方的“序列号 + 确认号 + 窗口大小信息”，是确认重传、流控的基础

传输过程中，如果 Checksum 校验失败、丢包或延时，发送端重传。

[2] 数据排序

TCP 有专门的序列号 SN 字段，可提供数据 re-order

[3] 流量控制

滑动窗口和计时器的使用。TCP 窗口中会指明双方能够发送接收的最大数据量，发送方通过维持一个发送滑动窗口来确保不会发生由于发送方报文发送太快接收方无法及时处理的问题。

[4] 拥塞控制

TCP 的拥塞控制由 4 个核心算法组成：

“慢启动”（Slow Start）

“拥塞避免”（Congestion avoidance）

“快速重传 ”（Fast Retransmit）

“快速恢复”（Fast Recovery）

239.【简答题】【网络基础】请你说一说 TCP 的流量控制

滑动窗口机制：

滑动窗口协议的基本原理就是在任意时刻，发送方都维持了一个连续的允许发送的帧的序号，称为发送窗口；同时，接收方也维持了一个连续的允许接收的帧的序号，称为接收窗口。发送窗口和接收窗口的序号的上下界不一定要一样，甚至大小也可以不同。不同的滑动窗口协议窗口大小一般不同。发送方窗口内的序列号代表了那些已经被发送，但是还没有被确认的帧，或者是那些可以被发送的帧。

举例：

发送和接受方都会维护一个数据帧的序列，这个序列被称作窗口。发送方的窗口大小由接受方确定，目的在于控制发送速度，以免接受方的缓存不够大，而导致溢出，同时控制流量也可以避免网络拥塞。图中的 4,5,6 号数据帧已经被发送出去，但是未收到关联的 ACK，7,8,9 帧则是等待发送。可以看出发送端的窗口大小为 6，这是由接受端告知的（事实上必须考虑拥塞窗口 cwnd，这里暂且考虑 cwnd>rwnd）。此时如果发送端收到 4 号 ACK，则窗口的左边缘向右收缩，窗口的右边缘则向右扩展，此时窗口就向前“滑动了”，即数据帧 10 也可以被发送。

240.【简答题】【网络基础】请你回答一下 TCP 三次握手，以及为什么不是两次

第一次握手：建立连接时，客户端发送 syn 包（syn=j）到服务器，并进入 SYN_SENT 状态，等待服务器确认；SYN：同步序列编号（Synchronize Sequence Numbers）。

第二次握手：服务器收到 syn 包，必须确认客户的 SYN（ack=j+1），同时自己也发送一个 SYN 包（syn=k），即

SYN+ACK 包，此时服务器进入 SYN_RECV 状态；

第三次握手：客户端收到服务器的 SYN+ACK 包，向服务器发送确认包 ACK(ack=k+1)，此包发送完毕，客户端和服务器进入 ESTABLISHED（TCP 连接成功）状态，完成三次握手。

为什么不是两次：

在服务端对客户端的请求进行回应(第二次握手)后，就会理所当然的认为连接已建立，而如果客户端并没有收到服务端的回应呢？此时，客户端仍认为连接未建立，服务端会对已建立的连接保存必要的资源，如果大量的这种情况，服务端会崩溃。

241.【简答题】【网络基础】请你回答一下 ipv6 的位数

IPv6 的 128 位地址通常写成 8 组，每组由四个十六进制数组成。

242.【简答题】【网络基础】请你说一说 osi 七层模型

物理层

在 OSI 参考模型中，物理层（Physical Layer）是参考模型的最低层，也是 OSI 模型的第一层。

物理层的主要功能是：利用传输介质为数据链路层提供物理连接，实现比特流的透明传输。

物理层的作用是实现相邻计算机节点之间比特流的透明传送，尽可能屏蔽掉具体传输介质和物理设备的差异。使其上面的数据链路层不必考虑网络的具体传输介质是什么。“透明传送比特流”表示经实际电路传送后的比特流没有发生变化，对传送的比特流来说，这个电路好像是看不见的。

数据链路层

数据链路层（Data Link Layer）是 OSI 模型的第二层，负责建立和管理节点间的链路。该层的主要功能是：通过各种控制协议，将有差错的物理信道变为无差错的、能可靠传输数据帧的数据链路。

在计算机网络中由于各种干扰的存在，物理链路是不可靠的。因此，这一层的主要功能是在物理层提供的比特流的基础上，通过差错控制、流量控制方法，使有差错的物理线路变为无差错的数据链路，即提供可靠的通过物理介质传输数据的方法。

该层通常又被分为介质访问控制（MAC）和逻辑链路控制（LLC）两个子层。

MAC 子层的主要任务是解决共享型网络中多用户对信道竞争的问题，完成网络介质的访问控制；

LLC 子层的主要任务是建立和维护网络连接，执行差错校验、流量控制和链路控制。

数据链路层的具体工作是接收来自物理层的位流形式的数据，并封装成帧，传送到上一层；同样，也将来自上层的数据帧，拆装为位流形式的数据转发到物理层；并且，还负责处理接收端发回的确认帧的信息，以便提供可靠的数据传输。

网络层

网络层（Network Layer）是 OSI 模型的第三层，它是 OSI 参考模型中最复杂的一层，也是通信子网的最高一层。它在下两层的基础上向资源子网提供服务。其主要任务是：通过路由选择算法，为报文或分组通过通信子网选择最适当的路径。该层控制数据链路层与传输层之间的信息转发，建立、维持和终止网络的连接。具体地说，数据链路层的数据在这一层被转换为数据包，然后通过路径选择、分段组合、顺序、进/出路由等控制，将信息从一个网

络设备传送到另一个网络设备。

一般地，数据链路层是解决同一网络内节点之间的通信，而网络层主要解决不同子网间的通信。例如在广域网之间通信时，必然会遇到路由（即两节点间可能有多条路径）选择问题。

在实现网络层功能时，需要解决的主要问题如下：

寻址：数据链路层中使用的物理地址（如 MAC 地址）仅解决网络内部的寻址问题。在不同子网之间通信时，为了识别和找到网络中的设备，每一子网中的设备都会被分配一个唯一的地址。由于各子网使用的物理技术可能不同，因此这个地址应当是逻辑地址（如 IP 地址）。

交换：规定不同的信息交换方式。常见的交换技术有：线路交换技术和存储转发技术，后者又包括报文交换技术和分组交换技术。

路由算法：当源节点和目的节点之间存在多条路径时，本层可以根据路由算法，通过网络为数据分组选择最佳路径，并将信息从最合适的路径由发送端传送到接收端。

连接服务：与数据链路层流量控制不同的是，前者控制的是网络相邻节点间的流量，后者控制的是从源节点到目的节点间的流量。其目的在于防止阻塞，并进行差错检测。

传输层

OSI 下 3 层的主要任务是数据通信，上 3 层的任务是数据处理。而传输层（Transport Layer）是 OSI 模型的第 4 层。因此该层是通信子网和资源子网的接口和桥梁，起到承上启下的作用。

该层的主要任务是：向用户提供可靠的端到端的差错和流量控制，保证报文的正确传输。传输层的作用是向高层屏蔽下层数据通信的细节，即向用户透明地传送报文。该层常见的协议：TCP/IP 中的 TCP 协议、Novell 网络中的 SPX 协议和微软的 NetBIOS/NetBEUI 协议。

传输层提供会话层和网络层之间的传输服务，这种服务从会话层获得数据，并在必要时，对数据进行分割。然后，传输层将数据传递到网络层，并确保数据能正确无误地传送到网络层。因此，传输层负责提供两节点之间数据的可靠传送，当两节点的联系确定之后，传输层则负责监督工作。综上，传输层的主要功能如下：

传输连接管理：提供建立、维护和拆除传输连接的功能。传输层在网络层的基础上为高层提供“面向连接”和“面向无接连”的两种服务。

处理传输差错：提供可靠的“面向连接”和不太可靠的“面向无连接”的数据传输服务、差错控制和流量控制。在提供“面向连接”服务时，通过这一层传输的数据将由目标设备确认，如果在指定的时间内未收到确认信息，数据将被重发。

监控服务质量。

会话层

会话层（Session Layer）是 OSI 模型的第 5 层，是用户应用程序和网络之间的接口，主要任务是：向两个实体的表示层提供建立和使用连接的方法。将不同实体之间的表示层的连接称为会话。因此会话层的任务就是组织和协调两个会话进程之间的通信，并对数据交换进行管理。

用户可以按照半双工、单工和全双工的方式建立会话。当建立会话时，用户必须提供他们想要连接的远程地址。而这些地址与 MAC（介质访问控制子层）地址或网络层的逻辑地址不同，它们是为用户专门设计的，更便于用户记忆。域名（DN）就是一种网络上使用的远程地址例如：www.3721.com 就是一个域名。会话层的具体功能如下：

会话管理：允许用户在两个实体设备之间建立、维持和终止会话，并支持它们之间的数据交换。例如提供单方向会话或双向同时会话，并管理会话中的发送顺序，以及会话所占用时间的长短。

会话流量控制：提供会话流量控制和交叉会话功能。

寻址：使用远程地址建立会话连接。l

出错控制：从逻辑上讲会话层主要负责数据交换的建立、保持和终止，但实际的工作却是接收来自传输层的数据，并负责纠正错误。会话控制和远程过程调用均属于这一层的功能。但应注意，此层检查的错误不是通信介质的错误，而是磁盘空间、打印机缺纸等类型的高级错误。

表示层

表示层（Presentation Layer）是 OSI 模型的第六层，它对来自应用层的命令和数据进行解释，对各种语法赋予相应的含义，并按照一定的格式传送给会话层。其主要功能是“处理用户信息的表示问题，如编码、数据格式转换和加密解密”等。表示层的具体功能如下：

数据格式处理：协商和建立数据交换的格式，解决各应用程序之间在数据格式表示上的差异。

数据的编码：处理字符集和数字的转换。例如由于用户程序中的数据类型（整型或实型、有符号或无符号等）、用户标识等都可以有不同的表示方式，因此，在设备之间需要具有在不同字符集或格式之间转换的功能。

压缩和解压缩：为了减少数据的传输量，这一层还负责数据的压缩与恢复。

数据的加密和解密：可以提高网络的安全性。

应用层

应用层（Application Layer）是 OSI 参考模型的最高层，它是计算机用户，以及各种应用程序和网络之间的接口，其功能是直接向用户提供服务，完成用户希望在网络上完成的各种工作。它在其他 6 层工作的基础上，负责完成网络中应用程序与网络操作系统之间的联系，建立与结束使用者之间的联系，并完成网络用户提出的各种网络服务及应用所需的监督、管理和服务等各种协议。此外，该层还负责协调各个应用程序间的工作。

应用层为用户提供的服务和协议有：文件服务、目录服务、文件传输服务（FTP）、远程登录服务（Telnet）、电子邮件服务（E-mail）、打印服务、安全服务、网络管理服务、数据库服务等。上述的各种网络服务由该层的不同应用协议和程序完成，不同的网络操作系统之间在功能、界面、实现技术、对硬件的支持、安全可靠性以及具有的各种应用程序接口等各个方面的差异是很大的。应用层的主要功能如下：

用户接口：应用层是用户与网络，以及应用程序与网络间的直接接口，使得用户能够与网络进行交互式联系。

实现各种服务：该层具有的各种应用程序可以完成和实现用户请求的各种服务。

243.【简答题】【测试基础】请你说一说 DNS 解析过程

1、浏览器先检查自身缓存中有没有被解析过的这个域名对应的 ip 地址，如果有，解析结束。同时域名被缓存的时间也可通过 TTL 属性来设置。

2、如果浏览器缓存中没有（专业点叫还没命中），浏览器会检查操作系统缓存中有没有对应的已解析过的结果。而操作系统也有一个域名解析的过程。在 windows 中可通过 c 盘里一个叫 hosts 的文件来设置，如果你在这里指定了一个域名对应的 ip 地址，那浏览器会首先使用这个 ip 地址。

但是这种操作系统级别的域名解析规程也被很多黑客利用，通过修改你的 hosts 文件里的内容把特定的域名解

析到他指定的 ip 地址上，造成所谓的域名劫持。所以在 windows7 中将 hosts 文件设置成了 readonly，防止被恶意篡改。

3.如果至此还没有命中域名，才会真正的请求本地域名服务器（LDNS）来解析这个域名，这台服务器一般在你的城市的某个角落，距离你不会很远，并且这台服务器的性能都很好，一般都会缓存域名解析结果，大约 80%的域名解析到这里就完成了。

4. 如果 LDNS 仍然没有命中，就直接跳到 Root Server 域名服务器请求解析

5. 根域名服务器返回给 LDNS 一个所查询域的主域名服务器（gTLD Server，国际顶尖域名服务器，如.com .cn .org 等）地址

6. 此时 LDNS 再发送请求给上一步返回的 gTLD

7. 接受请求的 gTLD 查找并返回这个域名对应的 Name Server 的地址，这个 Name Server 就是网站注册的域名服务器

8. Name Server 根据映射关系表找到目标 ip，返回给 LDNS

9. LDNS 缓存这个域名和对应的 ip

10. LDNS 把解析的结果返回给用户，用户根据 TTL 值缓存到本地系统缓存中，域名解析过程至此结束

244.【简答题】【网络基础】请你说一说 http 和 https 区别

HTTP 协议传输的数据都是未加密的，也就是明文的，因此使用 HTTP 协议传输隐私信息非常不安全，为了保证这些隐私数据能加密传输，于是网景公司设计了 SSL（Secure Sockets Layer）协议用于对 HTTP 协议传输的数据进行加密，从而就诞生了 HTTPS。简单来说，HTTPS 协议是由 SSL+HTTP 协议构建的可进行加密传输、身份认证的网络协议，要比 http 协议安全。

HTTPS 和 HTTP 的区别主要如下：

1、https 协议需要到 ca 申请证书，一般免费证书较少，因而需要一定费用。

2、http 是超文本传输协议，信息是明文传输，https 则是具有安全性的 ssl 加密传输协议。

3、http 和 https 使用的是完全不同的连接方式，用的端口也不一样，前者是 80，后者是 443。

4、http 的连接很简单，是无状态的；HTTPS 协议是由 SSL+HTTP 协议构建的可进行加密传输、身份认证的网络协议，比 http 协议安全。

245.【简答题】【网络基础】请你说一下 https 中 SSL 层原理

SSL 利用数据加密、身份验证和消息完整性验证机制，为网络上数据的传输提供安全性保证。SSL 支持各种应用层协议。由于 SSL 位于应用层和传输层之间，所以可以为任何基于 TCP 等可靠连接的应用层协议提供安全性保证。

1.身份验证机制

SSL 利用数字签名来验证通信对端的身份。非对称密钥算法可以用来实现数字签名。由于通过私钥加密后的数据只能利用对应的公钥进行解密，因此根据解密是否成功，就可以判断发送者的身份，如同发送者对数据进行了“签名”。例如，Alice 使用自己的私钥对一段固定的信息加密后发给 Bob，Bob 利用 Alice 的公钥解密，如果解密结

果与固定信息相同，那么就能够确认信息的发送者为 Alice，这个过程就称为数字签名。使用数字签名验证身份时，需要确保被验证者的公钥是真实的，否则，非法用户可能会冒充被验证者与验证者通信。如下图所示，Cindy 冒充 Bob，将自己的公钥发给 Alice，并利用自己的私钥计算出签名发送给 Alice，Alice 利用“Bob”的公钥（实际上为 Cindy 的公钥）成功验证该签名，则 Alice 认为 Bob 的身份验证成功，而实际上与 Alice 通信的是冒充 Bob 的 Cindy。SSL 利用 PKI 提供的机制保证公钥的真实性。

2.数据传输的机密性

SSL 加密通道上的数据加解密使用对称密钥算法，目前主要支持的算法有 DES、3DES、AES 等，这些算法都可以有效地防止交互数据被破解。对称密钥算法要求解密密钥和加密密钥完全一致。因此，利用对称密钥算法加密传输数据之前，需要在通信两端部署相同的密钥。

3. 消息完整性验证

为了避免网络中传输的数据被非法篡改，SSL 利用基于 MD5 或 SHA 的 MAC 算法来保证消息的完整性。MAC 算法是在密钥参与下的数据摘要算法，能将密钥和任意长度的数据转换为固定长度的数据。利用 MAC 算法验证消息完整性的过程如下图所示。发送者在密钥的参与下，利用 MAC 算法计算出消息的 MAC 值，并将其加在消息之后发送给接收者。接收者利用同样的密钥和 MAC 算法计算出消息的 MAC 值，并与接收到的 MAC 值比较。如果二者相同，则报文没有改变；否则，报文在传输过程中被修改，接收者将丢弃该报文。

MAC 算法要求通信双方具有相同的密钥，否则 MAC 值验证将会失败。因此，利用 MAC 算法验证消息完整性之前，需要在通信两端部署相同的密钥。

4.利用非对称密钥算法保证密钥本身的安全

对称密钥算法和 MAC 算法要求通信双方具有相同的密钥，否则解密或 MAC 值验证将失败。因此，要建立加密通道或验证消息完整性，必须先在通信双方部署一致的密钥。SSL 利用非对称密钥算法加密密钥的方法实现密钥交换，保证第三方无法获取该密钥。如下图所示，SSL 客户端（如 Web 浏览器）利用 SSL 服务器（如 Web 服务器）的公钥加密密钥，将加密后的密钥发送给 SSL 服务器，只有拥有对应私钥的 SSL 服务器才能从密文中获取原始的密钥。SSL 通常采用 RSA 算法加密传输密钥。（Server 端公钥加密密钥，私钥解密密钥）

实际上，SSL 客户端发送给 SSL 服务器的密钥不能直接用来加密数据或计算 MAC 值，该密钥是用来计算对称密钥和 MAC 密钥的信息，称为 premaster secret。SSL 客户端和 SSL 服务器利用 premaster secret 计算出相同的主密钥（master secret），再利用 master secret 生成用于对称密钥算法、MAC 算法等的密钥。premaster secret 是计算对称密钥、MAC 算法密钥的关键。

5.利用 PKI 保证公钥的真实性

PKI 通过数字证书来发布用户的公钥，并提供了验证公钥真实性的机制。数字证书（简称证书）是一个包含用户的公钥及其身份信息的文件，证明了用户与公钥的关联。数字证书由权威机构——CA 签发，并由 CA 保证数字证书的真实性。

SSL 客户端把密钥加密传递给 SSL 服务器之前，SSL 服务器需要将从 CA 获取的证书发送给 SSL 客户端，SSL 客户端通过 PKI 判断该证书的真实性。如果该证书确实属于 SSL 服务器，则利用该证书中的公钥加密密钥，发送给 SSL 服务器。

验证 SSL 服务器/SSL 客户端的身份之前，SSL 服务器/SSL 客户端需要将从 CA 获取的证书发送给对端，对端

通过 PKI 判断该证书的真实性。如果该证书确实属于 SSL 服务器/SSL 客户端，则对端利用该证书中的公钥验证 SSL 服务器/SSL 客户端的身份。

246.【简答题】【网络基础】请你说一说 TCP 断连过程，以及单向连接关闭后还能否通信

由于 TCP 连接是全双工的，因此每个方向都必须单独进行关闭。这个原则是当一方完成它的数据发送任务后就能发送一个 FIN 来终止这个方向的连接。收到一个 FIN 只意味着这一方向上没有数据流动，一个 TCP 连接在收到一个 FIN 后仍能发送数据。首先进行关闭的一方将执行主动关闭，而另一方执行被动关闭。四次挥手过程：

（1）客户端 A 发送一个 FIN，用来关闭客户 A 到服务器 B 的数据传送。

（2）服务器 B 收到这个 FIN，它发回一个 ACK，确认序号为收到的序号加 1。和 SYN 一样，一个 FIN 将占用一个序号。

（3）服务器 B 关闭与客户端 A 的连接，发送一个 FIN 给客户端 A。

（4）客户端 A 发回 ACK 报文确认，并将确认序号设置为收到序号加 1。

四次挥手原因：这是因为服务端的 LISTEN 状态下的 SOCKET 当收到 SYN 报文的建连请求后，它可以把 ACK 和 SYN（ACK 起应答作用，而 SYN 起同步作用）放在一个报文里来发送。但关闭连接时，当收到对方的 FIN 报文通知时，它仅仅表示对方没有数据发送给你了；但未必你所有的数据都全部发送给对方了，所以你可以未必会马上会关闭 SOCKET,也即你可能还需要发送一些数据给对方之后，再发送 FIN 报文给对方来表示你同意现在可以关闭连接了，所以它这里的 ACK 报文和 FIN 报文多数情况下都是分开发送的。

247.【简答题】【网络基础】请你说说 TCP 和 UDP 用一个端口发送信息是否冲突

不冲突，TCP、UDP 可以绑定同一端口来进行通信，许多协议已经这样做了，例如 DNS 适用于 udp / 53 和 tcp / 53。因为数据接收时时根据五元组{传输协议，源 IP，目的 IP，源端口，目的端口}判断接受者的。

248.【简答题】【网络基础】请你说说 HTTP 常见头

1、Accept：text/html, application/xhtml+xml, application/xml;q=0.9, image/webp, image/apng, */*; q=0.8

作用：向服务器申明客户端（浏览器）可以接受的媒体类型（MIME）的资源

解释：浏览器可以接受 text/html、application/xhtml+xml、application/xml 类型，通配符*/* 表示任意类型的数据。并且浏览器按照该顺序进行接收。(text/html —> application/xhtml+xml —> application/xml)

2、Accept-encoding: gzip, deflate, br

作用：向服务器申明客户端（浏览器）接收的编码方法，通常为压缩方法

解释：浏览器支持采用经过 gzip，deflate 或 br 压缩过的资源

3、Accept-Language: en-US,en;q=0.9,zh-CN;q=0.8,zh;q=0.7

作用：向服务器申明客户端（浏览器）接收的语言

解释：浏览器能够接受 en-US, en 和 zh-CN 三种语言，其中 en-US 的权重最高（q 最高为 1，最低为 0），服务器优先返回 en-US 语言

延伸：语言与字符集的区别：zh-CN 为汉语，汉语中有许多的编码：gbk2312 等

4、Cache-control: max-age=0

作用：控制浏览器的缓存，常见值为 private、no-cache、max-age、alidate，默认为 private，根据浏览器查看页面不同的方式来进行区别

解释：浏览器在访问了该页面后，不再会访问服务器

5、Cookie:

作用：告诉服务器关于 Session 的信息，存储让服务器辨识用户身份的信息。

6、Refer：https://www.baidu.com/xxxxxxxxxx

作用：告诉服务器该页面从哪个页面链接的

解释：该页面从 https://www.baidu.com 中的搜索结果中点击过来的

7、Upgrade-insecure-requests：1

作用：申明浏览器支持从 http 请求自动升级为 https 请求，并且在以后发送请求的时候都使用 https

解释：当页面中包含大量的 http 资源的时候（图片、iframe），如果服务器发现一旦存在上述的响应头的时候，会在加载 http 资源的时候自动替换为 https 请求

8、User-agent：Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3239.132 Safari/537.36

作用：向服务器发送浏览器的版本、系统、应用程序的信息。

解释：Chrome 浏览器的版本信息为 63.0.3239.132，并将自己伪装成 Safari，使用的是 WebKit 引擎，WebKit 伪装成 KHTML，KHTML 伪装成 Gecko（伪装是为了接收那些为 Mozilla、safari、gecko 编写的界面）

延伸：可以随便填（但不应该随便填）不过一般用于统计。

9、X-Chrome-UMA-Enabled、X-Client-Data ：与 Chrome 浏览器相关的数据

Response Headers

249.【简答题】【网络基础】请你说说 HTTP 状态码

状态码，100~199 表示请求已收到继续处理，200~299 表示成功，300~399 表示资源重定向，400~499 表示客户端请求出错，500~599 表示服务器端出错

200：响应成功

302：跳转，重定向

400：客户端有语法错误

403：服务器拒绝提供服务

404：请求资源不存在

500：服务器内部错误

250.【简答题】【网络基础】请你说说 soket 编程和 http 协议

由于通常情况下 Socket 连接就是 TCP 连接，因此 Socket 连接一旦建立，通信双方即可开始相互发送数据内容，直到双方连接断开。但在实际网络应用中，客户端到服务器之间的通信往往需要穿越多个中间节点，例如路由器、网关、防火墙等，大部分防火墙默认会关闭长时间处于非活跃状态的连接而导致 Socket 连接断连，因此需要通过轮询告诉网络，该连接处于活跃状态。

而 HTTP 连接使用的是“请求—响应”的方式，不仅在请求时需要先建立连接，而且需要客户端向服务器发出请求后，服务器端才能回复数据。

很多情况下，需要服务器端主动向客户端推送数据，保持客户端与服务器数据的实时与同步。此时若双方建立的是 Socket 连接，服务器就可以直接将数据传送给客户端；若双方建立的是 HTTP 连接，则服务器需要等到客户端发送一次请求后才能将数据传回给客户端，因此，客户端定时向服务器端发送连接请求，不仅可以保持在线，同时也是在“询问”服务器是否有新的数据，如果有就将数据传给客户端。

251.【简答题】【网络基础】请你说说 tcp 三次握手四次挥手

第一次握手：建立连接时，客户端发送 syn 包（syn=j）到服务器，并进入 SYN_SENT 状态，等待服务器确认；SYN：同步序列编号（Synchronize Sequence Numbers）。

第二次握手：服务器收到 syn 包，必须确认客户的 SYN（ack=j+1），同时自己也发送一个 SYN 包（syn=k），即 SYN+ACK 包，此时服务器进入 SYN_RECV 状态；

第三次握手：客户端收到服务器的 SYN+ACK 包，向服务器发送确认包 ACK(ack=k+1)，此包发送完毕，客户端和服务器进入 ESTABLISHED（TCP 连接成功）状态，完成三次握手。

由于 TCP 连接是全双工的，因此每个方向都必须单独进行关闭。这个原则是当一方完成它的数据发送任务后就能发送一个 FIN 来终止这个方向的连接。收到一个 FIN 只意味着这一方向上没有数据流动，一个 TCP 连接在收到一个 FIN 后仍能发送数据。首先进行关闭的一方将执行主动关闭，而另一方执行被动关闭。四次挥手过程：

（1）客户端 A 发送一个 FIN，用来关闭客户 A 到服务器 B 的数据传送。

（2）服务器 B 收到这个 FIN，它发回一个 ACK，确认序号为收到的序号加 1。和 SYN 一样，一个 FIN 将占用一个序号。

（3）服务器 B 关闭与客户端 A 的连接，发送一个 FIN 给客户端 A。

（4）客户端 A 发回 ACK 报文确认，并将确认序号设置为收到序号加 1。

四次挥手原因：这是因为服务端的 LISTEN 状态下的 SOCKET 当收到 SYN 报文的建连请求后，它可以把 ACK 和 SYN（ACK 起应答作用，而 SYN 起同步作用）放在一个报文里来发送。但关闭连接时，当收到对方的 FIN 报文通知时，它仅仅表示对方没有数据发送给你了；但未必你所有的数据都全部发送给对方了，所以你可以未必会马上会关闭 SOCKET,也即你可能还需要发送一些数据给对方之后，再发送 FIN 报文给对方来表示你同意现在可以关闭连接了，所以它这里的 ACK 报文和 FIN 报文多数情况下都是分开发送的。

252.【简答题】【网络基础】请你说一说 http 缓存问题，缓存寿命，以及怎么判断文件在服务器是否更改的

1 缓存的类型:

缓存是一种保存资源副本并在下次请求中直接使用该副本的技术，缓存能够节约网络资源，提升页面响应速度。常见的缓存类型分为共享缓存和私有缓存

1.1 私有缓存

私有缓存只能用于单独用户，常见的浏览器缓存便是私有缓存。私有缓存能够存储用户通过 http 下载过的文档，从而在用户再次访问时直接提供给用户，而不用向服务器发送请求。

1.2 共享缓存

共享缓存能够被多个用户使用，常用的 web 代理中便使用的共享缓存

缓存寿命

缓存寿命的计算的依据依次是：

请求头中的 Cache-Control: max-age=N。相应的缓存寿命即为 N，从设置开始，N 秒之后过期。

Expires 属性，Expires 属性的值为过期的时间点，在这个时间点后，该缓存被认为过期

Last-Modified 信息。缓存的寿命为头里面 Date 表示的事件点减去 Last-Modified 的时间点的结果乘以 10%

判断文件是否更改可以看文件时间戳

253.【简答题】【网络基础】请你回答一下搜索敏感词汇时，页面被重置的原理

根据 TCP 协议的规定，用户和服务器建立连接需要三次握手：第一次握手用户向服务器发送 SYN 数据包发出请求（SYN, x:0），第二次握手服务器向用户发送 SYN/ACK 数据包发出回应（SYN/ACK, y:x+1），第三次握手用户向服务器发送 ACK 数据包发出确认（ACK, x+1:y+1），至此一个 TCP 连接建立成功。其中 x 为用户向服务器发送的序列号，y 为服务器向用户发送的序列号。

关键字检测，针对明文或者 base64 等弱加密通讯内容，与准备好的敏感词库进行匹配，当发现敏感词时，将服务器发回的 SYN/ACK 包改成 SYN/ACK, Y:0，这代表 TCP 连接被重置，用户便主动放弃了连接，提示连接失败。让用户误认为服务器拒绝连接，而主动放弃继续与服务器连接，自动阻断记录含有敏感词的网页

254.【简答题】【网络基础】请你说一说 http 缓存问题，缓存寿命，怎么判断文件在服务器是否更改的

1 缓存的类型:

缓存是一种保存资源副本并在下次请求中直接使用该副本的技术，缓存能够节约网络资源，提升页面响应速度。常见的缓存类型分为共享缓存和私有缓存

1.1 私有缓存

私有缓存只能用于单独用户，常见的浏览器缓存便是私有缓存。私有缓存能够存储用户通过 http 下载过的文档，从而在用户再次访问时直接提供给用户，而不用向服务器发送请求。

1.2 共享缓存

共享缓存能够被多个用户使用，常用的 web 代理中便使用的共享缓存

缓存寿命

缓存寿命的计算的依据依次是：

请求头中的 Cache-Control: max-age=N。相应的缓存寿命即为 N，从设置开始，N 秒之后过期。

Expires 属性，Expires 属性的值为过期的时间点，在这个时间点后，该缓存被认为过期

Last-Modified 信息。缓存的寿命为头里面 Date 表示的事件点减去 Last-Modified 的时间点的结果乘以 10%

判断文件是否更改可以看文件时间戳

255.【简答题】【网络基础】请你说一说什么是 http 协议，http 的数据段包括什么？http 为什么是无状态的，http 和 https 的区别?ip 地址的 abcd 类是怎样分的，ABCD 分层协议为什么如此分层，什么是长连接和短链接

什么是 http 协议？

http（hyperText transport Protocol）是超文本传输协议的缩写，它用于传送 www 方式的数据，关于 http 协议采用了请求/响应模型，客户端向服务器发送了一个请求，服务器以一个状态行作为响应

http 的数据段包括什么？

通常 http 消息包括客户机向服务器请求消息和服务器向客户机的响应消息，这两种类型的消息由一个起始行，一个或多个头域，一个指示头域结束的空行和可选的消息体组成，http 的头域包括通用头，请求头，响应头，和实体头四个部分，每个头域由一个域名，冒号，和域值三部分组成，域名是大小写无关的，域值前可以添加任何数量的空格符，头域可以被扩张成多行，在每行开始处，使用至少一个空格或制表符。

http 为什么是无状态的？

无状态是指协议对于事务处理没有记忆能力，因为 http 协议目的在于支持超文本的传输，更加广义一点就是支持资源的传输，那么在客户端浏览器向服务器发送请求，继而服务器将相应的资源发回客户这样一个过程中，无论对于客户端还是服务器，都没有必要记录这个过程，因为每一次请求和响应都是相对独立的，一般而言，一个 url 对应唯一的超文本，正因为这样 d 唯一性，所以 http 协议被设计为无状态的链接协议符合他本身的需求。

http 和 https 的区别?

http 和 https 的区别主要如下：

1、https 需要到 ca 申请证书，因而需要一定费用

2、http 是超文本传输协议，信息是明文传输，https 则是具有安全性的 ssl 加密传输协议

3、http 的连接很简单，是无状态的，https 协议是由 ssl+http 协议构建的可进行加密串苏，身份验证的网络协议

4、http 用的端口是 80，https 用的端口是 443

ip 地址的 abcd 类是怎样分的

A 类地址的表示范围是：0.0.0.0-126.255.255.255，默认网络掩码为：255.0.0.0，A 类地址分配给规模特别大的网络使用，

B 类地址表示范围是：128.0.0.0-191.255.255.255，默认网络掩码为欸：255.255.0.0，B 类地址分配给一般的中型网络

C 类地址的表示范围是 192.0.0.0-223.255.255.255，默认网络掩码是：255.255.255.0，C 类地址分配给小型网络，如局域网

D 类地址称为广播地址，共特殊协议向选定的节点发送信息使用。

这样便于寻址和层次化的构造网络。

什么是长连接和短连接?

http1.0 中默认使用短连接，服务器和客户端没进行一次 http 操作，就建立一次连接，任务结束就终端连接，http1.1 起。默认使用长连接，用以保持连接特性，当一个网页打开完成后，服务器和客户端之间用于传输 http 数据的 tcp 连接不会关闭，客户端再次访问这个服务器时，会继续使用这一条已经建立好的连接。

256.【简答题】【网络基础】请你说一下 tcp 与 udp 的区别，以及 tcp 为什么可靠，tcp 滑动窗口，同传，拆包组装包是如何实现的

1、TCP 提供面向对象的连接，通信前要建立三次握手机制的连接，UDP 提供无连接的传输，传输前不用建立连接

2、TCP 提供可靠的，有序的，不丢失的传输，UDP 提供不可靠的传输

3、TCP 提供面向字节流的传输，它能将信息分割成组，并在接收端将其充足，UDP 提供面向数据报的传输，没有分组开销

4、TCP 提供拥塞控制，流量控制机制，UDP 没有

TCP 为什么可靠

1、确认和重传机制

建立连接时三次握手连接机制是确认重传流量控制的基础，传输过程中如校验失败，丢包或延时，发送端重传

2、数据排序

TCP 有专门的序列号字段，可提供数据 reorder

3、流量控制

TCP 窗口会指明双方能够发送接收的最大数据量

4、拥塞控制

TCP 滑动窗口

TCP 建立连接时，各端分配一个缓冲区用来存储接受的数据，并将缓冲区的尺寸发送给另一端，接收方发送的确认消息中包含了自己剩余的缓冲区尺寸，剩余缓冲区空间的数量叫做窗口，所谓滑动窗口，就是接收端可以根据自己的状况通告窗口大小，从而控制发送端的接收，进行流量控制.

Tcp 如何进行拆包、组装包？

拆包：

对于拆包目前常用的是以下两种方式：

1、动态缓冲区暂存方式。之所以说缓冲区是动态的是因为当需要缓冲的数据长度超出缓冲区的长度时会增大缓冲区长度。

大概过程描述如下：

A 为每一个连接动态分配一个缓冲区，同时把此缓冲区和 SOCKET 关联，常用的是通过结构体关联。

B 当接收到数据时首先把此段数据存放在缓冲区中。

C 判断缓存区中的数据长度是否够一个包头的长度，如不够，则不进行拆包操作。

D 根据包头数据解析出里面代表包体长度的变量。

E 判断缓存区中除包头外的数据长度是否够一个包体的长度，如不够，则不进行拆包操作。

F 取出整个数据包，这里的"取"的意思是不光从缓冲区中拷贝出数据包，而且要把此数据包从缓存区中删除掉。删除的办法就是把此包后面的数据移动到缓冲区的起始地址。

这种方法有两个缺点：

1）为每个连接动态分配一个缓冲区增大了内存的使用；

2）有三个地方需要拷贝数据，一个地方是把数据存放在缓冲区，一个地方是把完整的数据包从缓冲区取出来，一个地方是把数据包从缓冲区中删除。这种拆包的改进方法会解决和完善部分缺点。

2、利用底层的缓冲区来进行拆包

由于 TCP 也维护了一个缓冲区，所以我们完全可以利用 TCP 的缓冲区来缓存我们的数据，这样一来就不需要为每一个连接分配一个缓冲区了。另一方面我们知道 recv 或者 wsarecv 都有一个参数，用来表示我们要接收多长长度的数据。利用这两个条件我们就可以对第一种方法进行优化了。

对于阻塞 SOCKET 来说，我们可以利用一个循环来接收包头长度的数据，然后解析出代表包体长度的那个变量，再用一个循环来接收包体长度的数据。

257.【简答题】【网络基础】请你介绍下 session

Session：在 web 开发中，服务器可以为每个用户创建一个会话对象(session 对象)，默认情况下一个浏览器独占一个 session 对象，因此在需要保存用户数据时，服务器程序可以把用户数据写到用户浏览器独占的 session 中，当用户使用浏览器访问其他程序时，其他程序可以从用户的 session 中取出该用户的数据，为用户服务，其实现原理是服务器创建 session 出来后，会把 session 的 id 号，以 cookie 的形式回写给客户机，这样只要客户机的浏览器不关，再去访问服务器时，都会带着 session 的 id 号去，服务器发现客户机浏览器带 session id 过来了，就会使用内存中与之对应的 session 服务。

Session 和 cookie 的区别：

1、cookie 是把用户的数据写给用户浏览器

2、session 是把用户的数据写到用户独占的 session 中

3、session 对象由服务器创建，开发人员可以调用 request 对象的 getsession 方法得到 session 对象

258.【简答题】【网络基础】TCP/IP 五层协议

应用层、传输层、网络层、数据链路层、硬件层

259.【简答题】【网络基础】Internet 采用哪种网络协议?该协议的主要层次结构?Internet 物理地址和 IP 地址转换采用什么协议?

TCP/IP 协议主要层次结构为： 应用层/传输层/网络层/数链路层。

ARP (Address Resolution Protocol)(地据址解析协议)

260.【简答题】【java】请你说一下多态

什么是多态？

多态就是指程序中定义的引用变量所指向的具体类型和通过该引用变量发出的方法调用在编程时并不确定，而是在程序运行期间才确定，即一个引用变量倒底会指向哪个类的实例对象，该引用变量发出的方法调用到底是哪个类中实现的方法，必须在由程序运行期间才能决定。因为在程序运行时才确定具体的类，这样，不用修改源程序代码，就可以让引用变量绑定到各种不同的类实现上，从而导致该引用调用的具体方法随之改变，即不修改程序代码就可以改变程序运行时所绑定的具体代码，让程序可以选择多个运行状态，这就是多态性。

Java 实现多态有三个必要条件：继承、重写、向上转型。

继承：在多态中必须存在有继承关系的子类和父类。

重写：子类对父类中某些方法进行重新定义，在调用这些方法时就会调用子类的方法。

向上转型：在多态中需要将子类的引用赋给父类对象，只有这样该引用才能够具备技能调用父类的方法和子类的方法。

Java 中有两种形式可以实现多态，继承和接口：

基于继承的实现机制主要表现在父类和继承该父类的一个或多个子类对某些方法的重写，多个子类对同一方法的重写可以表现出不同的行为。

基于接口的多态中，指向接口的引用必须是指定这实现了该接口的一个类的实例程序，在运行时，根据对象引用的实际类型来执行对应的方法。

261.【简答题】【Java】请问 Java 中接口与抽象类是否相同？

不同

抽象类是用来捕捉子类的通用特性的。它不能被实例化，只能被用作子类的超类。抽象类是被用来创建继承层级里子类的模板。

接口是抽象方法的集合。如果一个类实现了某个接口，那么它就继承了这个接口的抽象方法。

抽象类和接口的对比：

参数

抽象类

接口

默认的方法实现

它可以有默认的方法实现

接口完全是抽象的，不存在方法的实现

实现

子类使用 extends 关键字来继承抽象类。如果子类不是抽象类的话，它需要提供抽象类中所有声明的方法的实现。

子类使用关键字 implements 来实现接口。它需要提供接口中所有声明的方法的实现

构造器

抽象类可以有构造器

接口不能有构造器

与正常 Java 类的区别

除了你不能实例化抽象类之外，它和普通 Java 类没有任何区别

接口是完全不同的类型

访问修饰符

抽象方法可以有 public、protected 和 default 这些修饰符

接口方法默认修饰符是 public，不可以使用其它修饰符。

main 方法

抽象方法可以有 main 方法并且可以运行

接口没有 main 方法。

多继承

抽象方法可以继承一个类和实现多个接口

接口只可以继承一个或多个其它接口

速度

它比接口速度要快

接口是稍微有点慢的，因为它需要时间去寻找在类中实现的方法。

添加新方法

如果你往抽象类中添加新的方法，你可以给它提供默认的实现。因此你不需要改变你现在的代码。

如果你往接口中添加方法，那么你必须改变实现该接口的类。

什么时候使用抽象类和接口：

如果拥有一些方法并且想让它们中的一些有默认实现，使用抽象类。

如果想实现多重继承，必须使用接口。由于 Java 不支持多继承，子类不能够继承多个类，但可以实现多个接口。

如果基本功能在不断改变，那么就需要使用抽象类。如果不断改变基本功能并且使用接口，那么就需要改变所有实现了该接口的类。

262.【简答题】【Java】请你说一下垃圾回收机制

垃圾回收(Garbage Collection)是 Java 虚拟机(JVM)垃圾回收器提供的一种用于在空闲时间不定时回收无任何对象引用的对象占据的内存空间的一种机制。

引用：如果 Reference 类型的数据中存储的数值代表的是另外一块内存的起始地址，就称这块内存代表着一个引用。

（1）强引用（Strong Reference）：如“Object obj = new Object（）”，这类引用是 Java 程序中最普遍的。只要强引用还存在，垃圾收集器就永远不会回收掉被引用的对象。

（2）软引用（Soft Reference）：它用来描述一些可能还有用，但并非必须的对象。在系统内存不够用时，这类引用关联的对象将被垃圾收集器回收。JDK1.2 之后提供了 SoftReference 类来实现软引用。

（3）弱引用（Weak Reference）：它也是用来描述非须对象的，但它的强度比软引用更弱些，被弱引用关联的对象只能生存到下一次垃圾收集发生之前。当垃圾收集器工作时，无论当前内存是否足够，都会回收掉只被弱引用关联的对象。在 JDK1.2 之后，提供了 WeakReference 类来实现弱引用。

（4）虚引用（Phantom Reference）：最弱的一种引用关系，完全不会对其生存时间构成影响，也无法通过虚引用来取得一个对象实例。为一个对象设置虚引用关联的唯一目的是希望能在这个对象被收集器回收时收到一个系统通知。JDK1.2 之后提供了 PhantomReference 类来实现虚引用。

垃圾：无任何对象引用的对象。

判断对象是否是垃圾的算法：

引用计数算法（Reference Counting Collector）、根搜索算法（Tracing Collector）：

回收：清理“垃圾”占用的内存空间而非对象本身。

Tracing 算法（Tracing Collector） 标记—清除算法：分为“标记”和“清除”两个阶段：首先标记出所需回收的对象，在标记完成后统一回收掉所有被标记的对象，它的标记过程其实就是前面的根搜索算法中判定垃圾对象的标记过程。

Compacting 算法（Compacting Collector）标记—整理算法：标记的过程与标记—清除算法中的标记过程一样，但对标记后出的垃圾对象的处理情况有所不同，它不是直接对可回收对象进行清理，而是让所有的对象都向一端移动，然后直接清理掉端边界以外的内存。在基于 Compacting 算法的收集器的实现中，一般增加句柄和句柄表。

Copying 算法（Copying Collector）：将内存按容量分为大小相等的两块，每次只使用其中的一块（对象面），当这一块的内存用完了，就将还存活着的对象复制到另外一块内存上面（空闲面），然后再把已使用过的内存空间一次清理掉。

Adaptive 算法（Adaptive Collector）：监控当前堆的使用情况，并将选择适当算法的垃圾收集器。

发生地点：一般发生在堆内存中，因为大部分的对象都储存在堆内存中。

（堆内存为了配合垃圾回收有什么不同区域划分，各区域有什么不同？）

Java 的堆内存基于 Generation 算法（Generational Collector）划分为新生代、年老代和持久代。新生代又被进一步划分为 Eden 和 Survivor 区，最后 Survivor 由 FromSpace（Survivor0）和 ToSpace（Survivor1）组成。所有通过 new 创建的对象的内存都在堆中分配，其大小可以通过-Xmx 和-Xms 来控制。分代收集基于这样一个事实：不同的对象的生命周期是不一样的。因此，可以将不同生命周期的对象分代，不同的代采取不同的回收算法进行垃圾回收

（GC），以便提高回收效率。

按执行机制划分 Java 有四种类型的垃圾回收器：

（1）串行垃圾回收器（Serial Garbage Collector）

（2）并行垃圾回收器（Parallel Garbage Collector）

（3）并发标记扫描垃圾回收器（CMS Garbage Collector）

（4）G1 垃圾回收器（G1 Garbage Collector）

发生时间：程序空闲时间不定时回收。

263.【简答题】【Java】请你说一下 Java 中的异常处理机制

在 Java 应用程序中，异常处理机制为：抛出异常，捕捉异常。

抛出异常：当一个方法出现错误引发异常时，方法创建异常对象并交付运行时系统，异常对象中包含了异常类型和异常出现时的程序状态等异常信息。运行时系统负责寻找处置异常的代码并执行。

捕获异常：在方法抛出异常之后，运行时系统将转为寻找合适的异常处理器（exception handler）。潜在的异常处理器是异常发生时依次存留在调用栈中的方法的集合。当异常处理器所能处理的异常类型与方法抛出的异常类型相符时，即为合适 的异常处理器。运行时系统从发生异常的方法开始，依次回查调用栈中的方法，直至找到含有合适异常处理器的方法并执行。当运行时系统遍历调用栈而未找到合适 的异常处理器，则运行时系统终止。同时，意味着 Java 程序的终止。

对于运行时异常、错误或可查异常，Java 技术所要求的异常处理方式有所不同：

由于运行时异常的不可查性，为了更合理、更容易地实现应用程序，Java 规定，运行时异常将由 Java 运行时系统自动抛出，允许应用程序忽略运行时异常。

对于方法运行中可能出现的 Error，当运行方法不欲捕捉时，Java 允许该方法不做任何抛出声明。因为，大多数 Error 异常属于永远不能被允许发生的状况，也属于合理的应用程序不该捕捉的异常。

对于所有的可查异常，Java 规定：一个方法必须捕捉，或者声明抛出方法之外。也就是说，当一个方法选择不捕捉可查异常时，它必须声明将抛出异常。

通常使用关键字 try、catch、finally 来捕获异常：

try 块：用于捕获异常。其后可接零个或多个 catch 块，如果没有 catch 块，则必须跟一个 finally 块。

catch 块：用于处理 try 捕获到的异常。

finally 块：无论是否捕获或处理异常，finally 块里的语句都会被执行。当在 try 块或 catch 块中遇到 return 语句时，finally 语句块将在方法返回之前被执行。在以下 4 种特殊情况下，finally 块不会被执行：

1）在 finally 语句块中发生了异常。

2）在前面的代码中用了 System.exit()退出程序。

3）程序所在的线程死亡。

4）关闭 CPU。

try、catch、finally 语句块的执行顺序:

1)当 try 没有捕获到异常时：try 语句块中的语句逐一被执行，程序将跳过 catch 语句块，执行 finally 语句块和其后的语句；

2)当 try 捕获到异常，catch 语句块里没有处理此异常的情况：当 try 语句块里的某条语句出现异常时，而没有处理此异常的 catch 语句块时，此异常将会抛给 JVM 处理，finally 语句块里的语句还是会被执行，但 finally 语句块后的语句不会被执行；

3)当 try 捕获到异常，catch 语句块里有处理此异常的情况：在 try 语句块中是按照顺序来执行的，当执行到某一条语句出现异常时，程序将跳到 catch 语句块，并与 catch 语句块逐一匹配，找到与之对应的处理程序，其他的 catch 语句块将不会被执行，而 try 语句块中，出现异常之后的语句也不会被执行，catch 语句块执行完后，执行 finally 语句块里的语句，最后执行 finally 语句块后的语句。

264.【简答题】【Java】请问多线程是什么？

最开始，线程只是用于分配单个处理器的处理时间的一种工具。但假如操作系统本身支持多个处理器，那么每个线程都可分配给一个不同的处理器，真正进入“并行运算”状态。从程序设计语言的角度看，多线程操作最有价值的特性之一就是程序员不必关心到底使用了多少个处理器。程序在逻辑意义上被分割为数个线程;假如机器本身安装了多个处理器，那么程序会运行得更快，毋需作出任何特殊的调校。根据前面的论述，大家可能感觉线程处理非常简单。但必须注意一个问题：共享资源!如果有多个线程同时运行，而且它们试图访问相同的资源，就会遇到一个问题。举个例子来说，两个线程不能将信息同时发送给一台打印机。为解决这个问题，对那些可共享的资源来说(比如打印机)，它们在使用期间必须进入锁定状态。所以一个线程可将资源锁定，在完成了它的任务后，再解开(释放)这个锁，使其他线程可以接着使用同样的资源。

多线程是为了同步完成多项任务，不是为了提高运行效率，而是为了提高资源使用效率来提高系统的效率。线程是在同一时间需要完成多项任务的时候实现的。

一个采用了多线程技术的应用程序可以更好地利用系统资源。其主要优势在于充分利用了 CPU 的空闲时间片，可以用尽可能少的时间来对用户的要求做出响应，使得进程的整体运行效率得到较大提高，同时增强了应用程序的灵活性。更为重要的是，由于同一进程的所有线程是共享同一内存，所以不需要特殊的数据传送机制，不需要建立共享存储区或共享文件，从而使得不同任务之间的协调操作与运行、数据的交互、资源的分配等问题更加易于解决。

265.【简答题】【Java】请你来聊一聊集合类和内存

一、集合类。

Java 中的集合包含多种数据结构，如链表、队列、哈希表等。从类的继承结构来说，可以分为两大类，一类是继承自 Collection 接口，这类集合包含 List、Set 和 Queue 等集合类。另一类是继承自 Map 接口，这主要包含了哈希表相关的集合类。

1、List、Set 和 Queue 类的继承结构图：绿色的虚线代表实现，绿色实线代表接口之间的继承，蓝色实线代表类之间的继承。

Collection 接口除了实现映射的集合类之外的所有集合类定义了一些方法

List 集合类型：描述了一种按位置存储数据的对象，是有序的。用的比较多 List 包括 ArrayList 和 LinkedList，这两者的区别：ArrayList 的底层的通过数组实现，所以其随机访问的速度比较快，但是对于需要频繁的增删的情况，效率就比较低了。而对于 LinkedList，底层通过链表来实现，所以增删操作比较容易完成，但是对于随机访问的效率比较低。

Queue：一般可以直接使用 LinkedList 完成，LinkedList 继承自 Deque，所以 LinkedList 具有双端队列的功能。PriorityQueue 是为每个元素提供一个优先级，优先级高的元素会优先出队列。

Set：Set 与 List 的主要区别是 Set 是不允许元素是重复的，而 List 则可以允许元素是重复的。HashSet 和 LinkedHashSet 的区别在于后者可以保证元素插入集合的元素顺序与输出顺序保持一致。而 TresSet 的区别在于其排序是按照 Comparator 来进行排序的，默认情况下按照字符的自然顺序进行升序排列。

Iterable：Collection 类继承自 Iterable，该接口的作用是提供元素遍历的功能，也就是说所有的集合类（除 Map 相关的类）都提供元素遍历的功能。Iterable 里面包含了 Iterator 的迭代器。

2、Map 类型的集合：最大的优点在于其查找效率比较高，理想情况下可以实现 O(1)的时间复杂度。Map 中最常用的是 HashMap，LinkedHashMap 与 HashMap 的区别在于前者能够保证插入集合的元素顺序与输出顺序一致。这两者与 TreeMap 的区别在于 TreeMap 是根据键值进行排序的，其底层的实现也有本质的区别，HashMap 底层是一个哈希表，而 TreeMap 的底层数据结构是一棵树。

二、Java 内存区域划分

1.程序计数器：

可以看做是当前线程所执行的字节码的行号指示器。在 JVM 的概念模型里，字节码解释器工作时就是通过改变这个计数器的值来选取下一条需要执行的字节码指令。

每条线程都有一个独立的程序计数器，所以程序计数器是线程私有的内存区域。

如果线程执行的是一个 Java 方法，计数器记录的是正在执行的虚拟机字节码指令的地址；如果线程执行的是一个 Native 方法，计数器的值为空。

Java 虚拟机规范中唯一一个没有规定任何 OutOfMemoryError 情况的区域。

2.Java 虚拟机栈：

描述 Java 方法执行的内存模型，每个方法执行的同时会创建一个栈帧，栈帧用于存储局部变量表、操作数栈、动态链接、方法出口等信息。每个方法从调用直至执行完成的过程，就对应着一个栈帧在虚拟机栈中入栈到出栈的过程。

Java 虚拟机栈是线程私有的，它的生命周期与线程相同。

局部变量表存放了编译时期可知的各种基本数据类型和对象引用。局部变量表所需的内存空间在编译时期完成分配，当进入一个方法时，这个方法需要在栈帧中分配多大的局部变量空间是完全确定的，在方法运行期间不会改变局部变量表的大小。

Java 虚拟机规范对这个区域规定了两种异常情况：

如果线程请求的栈深度大于虚拟机所允许的深度，将抛出 StackOverflowError 异常；

如果虚拟机栈可以动态扩展，如果扩展时无法申请到足够的内存，就会抛出 OutOfMemoryError 异常；

3.本地方法栈：

本地方法栈与虚拟机栈的区别：虚拟机栈为虚拟机执行 Java 方法服务（也就是字节码），而本地方法栈为虚拟

机使用到的 Native 方法服务。

Java 虚拟机规范对这个区域规定了两种异常情况：StackOverflowError 和 OutOfMemoryError 异常。

4.Java 堆：

Java 堆是被所有的线程共享的一块内存区域，在虚拟机启动时创建。Java 堆的唯一目的就是存放对象实例，几乎所有的对象实例都在这里分配内存。

Java 堆是垃圾回收器管理的主要区域，从内存回收的角度看，由于现在收集器基本都采用分代收集算法，所以 Java 堆可以细分为：新生代、老生代；从内存分配的角度看，线程共享的 Java 堆可能划分出多个线程私有的分配缓冲区（TLAB）。

Java 堆可以处于物理上不连续的内存空间中，只要逻辑上是连续的即可。

Java 虚拟机规范规定，如果在堆上没有内存完成实例分配，并且堆上也无法再扩展时，将会抛出 OutOfMemoryError 异常。

Java 堆内存的 OOM 异常：

内存泄露：指程序中一些对象不会被 GC 所回收，它始终占用内存，即被分配的对象引用链可达但已无用。

内存溢出：程序运行过程中无法申请到足够的内存而导致的一种错误。内存溢出通常发生于 OLD 段或 Perm 段垃圾回收后，仍然无内存空间容纳新的 Java 对象的情况。

5.方法区：

被所有的线程共享的一块内存区域。它用于存储已被虚拟机加载的类信息、常量、静态变量、即时编译器编译后的代码等数据。

不需要连续的内存和可以选择固定大小或者可扩展之外，还可以选择不实现垃圾回收。

Java 虚拟机规范规定，当方法区无法满足内存分配的需求时，将抛出 OutOfMemoryError 异常。

266.【简答题】【Java】请你说一下 java jvm 的内存机制

Java 内存区域划分

1.程序计数器：

可以看做是当前线程所执行的字节码的行号指示器。在 JVM 的概念模型里，字节码解释器工作时就是通过改变这个计数器的值来选取下一条需要执行的字节码指令。

每条线程都有一个独立的程序计数器，所以程序计数器是线程私有的内存区域。

如果线程执行的是一个 Java 方法，计数器记录的是正在执行的虚拟机字节码指令的地址；如果线程执行的是一个 Native 方法，计数器的值为空。

Java 虚拟机规范中唯一一个没有规定任何 OutOfMemoryError 情况的区域。

2.Java 虚拟机栈：

描述 Java 方法执行的内存模型，每个方法执行的同时会创建一个栈帧，栈帧用于存储局部变量表、操作数栈、动态链接、方法出口等信息。每个方法从调用直至执行完成的过程，就对应着一个栈帧在虚拟机栈中入栈到出栈的过程。

Java 虚拟机栈是线程私有的，它的生命周期与线程相同。

局部变量表存放了编译时期可知的各种基本数据类型和对象引用。局部变量表所需的内存空间在编译时期完成分配，当进入一个方法时，这个方法需要在栈帧中分配多大的局部变量空间是完全确定的，在方法运行期间不会改变局部变量表的大小。

Java 虚拟机规范对这个区域规定了两种异常情况：

如果线程请求的栈深度大于虚拟机所允许的深度，将抛出 StackOverflowError 异常；

如果虚拟机栈可以动态扩展，如果扩展时无法申请到足够的内存，就会抛出 OutOfMemoryError 异常；

3.本地方法栈：

本地方法栈与虚拟机栈的区别：虚拟机栈为虚拟机执行 Java 方法服务（也就是字节码），而本地方法栈为虚拟机使用到的 Native 方法服务。

Java 虚拟机规范对这个区域规定了两种异常情况：StackOverflowError 和 OutOfMemoryError 异常。

4.Java 堆：

Java 堆是被所有的线程共享的一块内存区域，在虚拟机启动时创建。Java 堆的唯一目的就是存放对象实例，几乎所有的对象实例都在这里分配内存。

Java 堆是垃圾回收器管理的主要区域，从内存回收的角度看，由于现在收集器基本都采用分代收集算法，所以 Java 堆可以细分为：新生代、老生代；从内存分配的角度看，线程共享的 Java 堆可能划分出多个线程私有的分配缓冲区（TLAB）。

Java 堆可以处于物理上不连续的内存空间中，只要逻辑上是连续的即可。

Java 虚拟机规范规定，如果在堆上没有内存完成实例分配，并且堆上也无法再扩展时，将会抛出 OutOfMemoryError 异常。

Java 堆内存的 OOM 异常：

内存泄露：指程序中一些对象不会被 GC 所回收，它始终占用内存，即被分配的对象引用链可达但已无用。

内存溢出：程序运行过程中无法申请到足够的内存而导致的一种错误。内存溢出通常发生于 OLD 段或 Perm 段垃圾回收后，仍然无内存空间容纳新的 Java 对象的情况。

5.方法区：

被所有的线程共享的一块内存区域。它用于存储已被虚拟机加载的类信息、常量、静态变量、即时编译器编译后的代码等数据。

不需要连续的内存和可以选择固定大小或者可扩展之外，还可以选择不实现垃圾回收。

Java 虚拟机规范规定，当方法区无法满足内存分配的需求时，将抛出 OutOfMemoryError 异常。

267.【简答题】【Java】请你说一说有哪几种垃圾回收算法

Tracing 算法（Tracing Collector） 标记—清除算法：分为“标记”和“清除”两个阶段：首先标记出所需回收的对象，在标记完成后统一回收掉所有被标记的对象，它的标记过程其实就是前面的根搜索算法中判定垃圾对象的标记过程。

Compacting 算法（Compacting Collector）标记—整理算法：标记的过程与标记—清除算法中的标记过程一样，但对标记后出的垃圾对象的处理情况有所不同，它不是直接对可回收对象进行清理，而是让所有的对象都向一端移

动，然后直接清理掉端边界以外的内存。在基于 Compacting 算法的收集器的实现中，一般增加句柄和句柄表。

Copying 算法（Copying Collector）：将内存按容量分为大小相等的两块，每次只使用其中的一块（对象面），当这一块的内存用完了，就将还存活着的对象复制到另外一块内存上面（空闲面），然后再把已使用过的内存空间一次清理掉。

Adaptive 算法（Adaptive Collector）：监控当前堆的使用情况，并将选择适当算法的垃圾收集器。

268.【简答题】【Java】请你说一说垃圾收集机制

垃圾回收(Garbage Collection)是 Java 虚拟机(JVM)垃圾回收器提供的一种用于在空闲时间不定时回收无任何对象引用的对象占据的内存空间的一种机制。

引用：如果 Reference 类型的数据中存储的数值代表的是另外一块内存的起始地址，就称这块内存代表着一个引用。

（1）强引用（Strong Reference）：如“Object obj = new Object（）”，这类引用是 Java 程序中最普遍的。只要强引用还存在，垃圾收集器就永远不会回收掉被引用的对象。

（2）软引用（Soft Reference）：它用来描述一些可能还有用，但并非必须的对象。在系统内存不够用时，这类引用关联的对象将被垃圾收集器回收。JDK1.2 之后提供了 SoftReference 类来实现软引用。

（3）弱引用（Weak Reference）：它也是用来描述非须对象的，但它的强度比软引用更弱些，被弱引用关联的对象只能生存到下一次垃圾收集发生之前。当垃圾收集器工作时，无论当前内存是否足够，都会回收掉只被弱引用关联的对象。在 JDK1.2 之后，提供了 WeakReference 类来实现弱引用。

（4）虚引用（Phantom Reference）：最弱的一种引用关系，完全不会对其生存时间构成影响，也无法通过虚引用来取得一个对象实例。为一个对象设置虚引用关联的唯一目的是希望能在这个对象被收集器回收时收到一个系统通知。JDK1.2 之后提供了 PhantomReference 类来实现虚引用。

垃圾：无任何对象引用的对象。

判断对象是否是垃圾的算法：

引用计数算法（Reference Counting Collector）、根搜索算法（Tracing Collector）：

回收：清理“垃圾”占用的内存空间而非对象本身。

Tracing 算法（Tracing Collector） 标记—清除算法：分为“标记”和“清除”两个阶段：首先标记出所需回收的对象，在标记完成后统一回收掉所有被标记的对象，它的标记过程其实就是前面的根搜索算法中判定垃圾对象的标记过程。

Compacting 算法（Compacting Collector）标记—整理算法：标记的过程与标记—清除算法中的标记过程一样，但对标记后出的垃圾对象的处理情况有所不同，它不是直接对可回收对象进行清理，而是让所有的对象都向一端移动，然后直接清理掉端边界以外的内存。在基于 Compacting 算法的收集器的实现中，一般增加句柄和句柄表。

Copying 算法（Copying Collector）：将内存按容量分为大小相等的两块，每次只使用其中的一块（对象面），当这一块的内存用完了，就将还存活着的对象复制到另外一块内存上面（空闲面），然后再把已使用过的内存空间一次清理掉。

Adaptive 算法（Adaptive Collector）：监控当前堆的使用情况，并将选择适当算法的垃圾收集器。

发生地点：一般发生在堆内存中，因为大部分的对象都储存在堆内存中。

（堆内存为了配合垃圾回收有什么不同区域划分，各区域有什么不同？）

Java 的堆内存基于 Generation 算法（Generational Collector）划分为新生代、年老代和持久代。新生代又被进一步划分为 Eden 和 Survivor 区，最后 Survivor 由 FromSpace（Survivor0）和 ToSpace（Survivor1）组成。所有通过 new 创建的对象的内存都在堆中分配，其大小可以通过-Xmx 和-Xms 来控制。分代收集基于这样一个事实：不同的对象的生命周期是不一样的。因此，可以将不同生命周期的对象分代，不同的代采取不同的回收算法进行垃圾回收（GC），以便提高回收效率。

按执行机制划分 Java 有四种类型的垃圾回收器：

（1）串行垃圾回收器（Serial Garbage Collector）

（2）并行垃圾回收器（Parallel Garbage Collector）

（3）并发标记扫描垃圾回收器（CMS Garbage Collector）

（4）G1 垃圾回收器（G1 Garbage Collector）

发生时间：程序空闲时间不定时回收。

269.【简答题】【Java】请你回答一下 GC Root 可以是哪些

1 、 虚拟机栈（栈帧中的本地变量表）中引用的对象。

2、 本地方法栈中 JNI（即一般说的 native 方法）引用的对象。

3、 方法区中的静态变量和常量引用的对象。

270.【简答题】【Java】请你说一下 OOM 可能发生在哪，怎么查看，怎么调优

除了程序计数器不会抛出 OOM 外，其他各个内存区域都可能会抛出 OOM。

最常见的 OOM 情况有以下三种：

- java.lang.OutOfMemoryError: Java heap space ------>java 堆内存溢出，此种情况最常见，一般由于内存泄露或者堆的大小设置不当引起。对于内存泄露，需要通过内存监控软件查找程序中的泄露代码，而堆大小可以通过虚拟机参数-Xms,-Xmx 等修改。
- java.lang.OutOfMemoryError: PermGen space ------>java 永久代溢出，即方法区溢出了，一般出现于大量 Class 或者 jsp 页面，或者采用 cglib 等反射机制的情况，因为上述情况会产生大量的 Class 信息存储于方法区。此种情况可以通过更改方法区的大小来解决，使用类似-XX:PermSize=64m -XX:MaxPermSize=256m 的形式修改。另外，过多的常量尤其是字符串也会导致方法区溢出。
- java.lang.StackOverflowError ------> 不会抛 OOM error，但也是比较常见的 Java 内存溢出。JAVA 虚拟机栈溢出，一般是由于程序中存在死循环或者深度递归调用造成的，栈大小设置太小也会出现此种溢出。可以通过虚拟机参数-Xss 来设置栈的大小。

OOM 分析--heapdump

要 dump 堆的内存镜像，可以采用如下两种方式：

- 设置 JVM 参数-XX:+HeapDumpOnOutOfMemoryError，设定当发生 OOM 时自动 dump 出堆信息。不过该

方法需要 JDK5 以上版本。

- 使用 JDK 自带的 jmap 命令。"jmap -dump:format=b,file=heap.bin <pid>"　其中 pid 可以通过 jps 获取。

dump 堆内存信息后，需要对 dump 出的文件进行分析，从而找到 OOM 的原因。常用的工具有：

- mat: eclipse memory analyzer, 基于 eclipse RCP 的内存分析工具。
- jhat：JDK 自带的 java heap analyze tool，可以将堆中的对象以 html 的形式显示出来，包括对象的数量，大小等等，并支持对象查询语言 OQL，分析相关的应用后，可以通过 http://localhost:7000 来访问分析结果。不推荐使用，因为在实际的排查过程中，一般是先在生产环境 dump 出文件来，然后拉到自己的开发机器上分析，所以，不如采用高级的分析工具比如前面的 mat 来的高效。

271.【简答题】【Java】请你说一下类加载

1、什么是类的加载

类的加载指的是将类的.class 文件中的二进制数据读入到内存中，将其放在运行时数据区的方法区内，然后在堆区创建一个 java.lang.Class 对象，用来封装类在方法区内的数据结构。类的加载的最终产品是位于堆区中的 Class 对象，Class 对象封装了类在方法区内的数据结构，并且向 Java 程序员提供了访问方法区内的数据结构的接口。类加载器并不需要等到某个类被“首次主动使用”时再加载它，JVM 规范允许类加载器在预料某个类将要被使用时就预先加载它，如果在预先加载的过程中遇到了.class 文件缺失或存在错误，类加载器必须在程序首次主动使用该类时才报告错误（LinkageError 错误）如果这个类一直没有被程序主动使用，那么类加载器就不会报告错误。

加载.class 文件的方式

– 从本地系统中直接加载

– 通过网络下载.class 文件

– 从 zip，jar 等归档文件中加载.class 文件

– 从专有数据库中提取.class 文件

– 将 Java 源文件动态编译为.class 文件

2、类的生命周期

类加载的过程包括了加载、验证、准备、解析、初始化五个阶段。在这五个阶段中，加载、验证、准备和初始化这四个阶段发生的顺序是确定的，而解析阶段则不一定，它在某些情况下可以在初始化阶段之后开始，这是为了支持 Java 语言的运行时绑定（也成为动态绑定或晚期绑定）。另外注意这里的几个阶段是按顺序开始，而不是按顺序进行或完成，因为这些阶段通常都是互相交叉地混合进行的，通常在一个阶段执行的过程中调用或激活另一个阶段。

A.加载：查找并加载类的二进制数据

加载时类加载过程的第一个阶段，在加载阶段，虚拟机需要完成以下三件事情：

1、通过一个类的全限定名来获取其定义的二进制字节流。

2、将这个字节流所代表的静态存储结构转化为方法区的运行时数据结构。

3、在 Java 堆中生成一个代表这个类的 java.lang.Class 对象，作为对方法区中这些数据的访问入口。

相对于类加载的其他阶段而言，加载阶段（准确地说，是加载阶段获取类的二进制字节流的动作）是可控性最

强的阶段，因为开发人员既可以使用系统提供的类加载器来完成加载，也可以自定义自己的类加载器来完成加载。加载阶段完成后，虚拟机外部的二进制字节流就按照虚拟机所需的格式存储在方法区之中，而且在 Java 堆中也创建一个 java.lang.Class 类的对象，这样便可以通过该对象访问方法区中的这些数据。

B.连接

– 验证：确保被加载的类的正确性

验证是连接阶段的第一步，这一阶段的目的是为了确保 Class 文件的字节流中包含的信息符合当前虚拟机的要求，并且不会危害虚拟机自身的安全。验证阶段大致会完成 4 个阶段的检验动作：

- 文件格式验证：验证字节流是否符合 Class 文件格式的规范；例如：是否以 0xCAFEBABE 开头、主次版本号是否在当前虚拟机的处理范围之内、常量池中的常量是否有不被支持的类型。
- 元数据验证：对字节码描述的信息进行语义分析（注意：对比 javac 编译阶段的语义分析），以保证其描述的信息符合 Java 语言规范的要求；例如：这个类是否有父类，除了 java.lang.Object 之外。
- 字节码验证：通过数据流和控制流分析，确定程序语义是合法的、符合逻辑的。
- 符号引用验证：确保解析动作能正确执行。

验证阶段是非常重要的，但不是必须的，它对程序运行期没有影响，如果所引用的类经过反复验证，那么可以考虑采用-Xverifynone 参数来关闭大部分的类验证措施，以缩短虚拟机类加载的时间。

– 准备：为类的静态变量分配内存，并将其初始化为默认值

准备阶段是正式为类变量分配内存并设置类变量初始值的阶段，这些内存都将在方法区中分配。

– 解析：把类中的符号引用转换为直接引用

解析阶段是虚拟机将常量池内的符号引用替换为直接引用的过程，解析动作主要针对类或接口、字段、类方法、接口方法、方法类型、方法句柄和调用点限定符 7 类符号引用进行。符号引用就是一组符号来描述目标，可以是任何字面量。直接引用就是直接指向目标的指针、相对偏移量或一个间接定位到目标的句柄。

C.初始化

初始化，为类的静态变量赋予正确的初始值，JVM 负责对类进行初始化，主要对类变量进行初始化。在 Java 中对类变量进行初始值设定有两种方式：

①声明类变量是指定初始值

②使用静态代码块为类变量指定初始值

JVM 初始化步骤

1、假如这个类还没有被加载和连接，则程序先加载并连接该类

2、假如该类的直接父类还没有被初始化，则先初始化其直接父类

3、假如类中有初始化语句，则系统依次执行这些初始化语句

类初始化时机：只有当对类的主动使用的时候才会导致类的初始化，类的主动使用包括以下六种：

– 创建类的实例，也就是 new 的方式

– 访问某个类或接口的静态变量，或者对该静态变量赋值

– 调用类的静态方法

– 反射（如 Class.forName(“com.shengsiyuan.Test”)）

– 初始化某个类的子类，则其父类也会被初始化

– Java 虚拟机启动时被标明为启动类的类（Java Test），直接使用 java.exe 命令来运行某个主类

D.结束生命周期

在如下几种情况下，Java 虚拟机将结束生命周期

– 执行了 System.exit()方法

– 程序正常执行结束

– 程序在执行过程中遇到了异常或错误而异常终止

– 由于操作系统出现错误而导致 Java 虚拟机进程终止

3、类的加载

类加载有三种方式：

- 命令行启动应用时候由 JVM 初始化加载
- 通过 Class.forName()方法动态加载
- 通过 ClassLoader.loadClass()方法动态加载

4、双亲委派模型

双亲委派模型的工作流程是：如果一个类加载器收到了类加载的请求，它首先不会自己去尝试加载这个类，而是把请求委托给父加载器去完成，依次向上，因此，所有的类加载请求最终都应该被传递到顶层的启动类加载器中，只有当父加载器在它的搜索范围中没有找到所需的类时，即无法完成该加载，子加载器才会尝试自己去加载该类。

双亲委派机制:

- 当 AppClassLoader 加载一个 class 时，它首先不会自己去尝试加载这个类，而是把类加载请求委派给父类加载器 ExtClassLoader 去完成。
- 当 ExtClassLoader 加载一个 class 时，它首先也不会自己去尝试加载这个类，而是把类加载请求委派给 BootStrapClassLoader 去完成。
- 如果 BootStrapClassLoader 加载失败（例如在$JAVA_HOME/jre/lib 里未查找到该 class），会使用 ExtClassLoader 来尝试加载；
- 若 ExtClassLoader 也加载失败，则会使用 AppClassLoader 来加载，如果 AppClassLoader 也加载失败，则会报出异常 ClassNotFoundException。

272.【简答题】【Java】请你说一下 AQS

Java 并发包（JUC）中提供了很多并发工具，这其中，很多我们耳熟能详的并发工具，譬如 ReentrangLock、Semaphore，它们的实现都用到了一个共同的基类--AbstractQueuedSynchronizer,简称 AQS。AQS 是一个用来构建锁和同步器的框架，使用 AQS 能简单且高效地构造出应用广泛的大量的同步器，比如我们提到的 ReentrantLock，Semaphore，其他的诸如 ReentrantReadWriteLock，SynchronousQueue，FutureTask 等等皆是基于 AQS 的。

AQS 使用一个 int 成员变量来表示同步状态，通过内置的 FIFO 队列来完成获取资源线程的排队工作。状态信息通过 procted 类型的 getState，setState，compareAndSetState 进行操作

AQS 支持两种同步方式：1.独占式 2.共享式。这样方便使用者实现不同类型的同步组件，独占式如 ReentrantLock，共享式如 Semaphore，CountDownLatch，组合式的如 ReentrantReadWriteLock。总之，AQS 为使用提供了底层支撑，如何组装实现，使用者可以自由发挥。同步器的设计是基于模板方法模式的，一般的使用方式是这样：

1.使用者继承 AbstractQueuedSynchronizer 并重写指定的方法。（这些重写方法很简单，无非是对于共享资源 state 的获取和释放）

2.将 AQS 组合在自定义同步组件的实现中，并调用其模板方法，而这些模板方法会调用使用者重写的方法。

273.【简答题】【Java】请你说一下 volatile

一旦一个共享变量（类的成员变量、类的静态成员变量）被 volatile 修饰之后，那么就具备了两层语义：1）保证了不同线程对这个变量进行操作时的可见性，即一个线程修改了某个变量的值，这新值对其他线程来说是立即可见的。2）禁止进行指令重排序。

《深入理解 Java 虚拟机》中对 volatile 的描述：“观察加入 volatile 关键字和没有加入 volatile 关键字时所生成的汇编代码发现，加入 volatile 关键字时，会多出一个 lock 前缀指令”

lock 前缀指令实际上相当于一个内存屏障（也成内存栅栏），内存屏障会提供 3 个功能：

1）它确保指令重排序时不会把其后面的指令排到内存屏障之前的位置，也不会把前面的指令排到内存屏障的后面；即在执行到内存屏障这句指令时，在它前面的操作已经全部完成；

2）它会强制将对缓存的修改操作立即写入主存；

3）如果是写操作，它会导致其他 CPU 中对应的缓存行无效。

因为 volatile 关键字无法保证操作的原子性。通常来说，使用 volatile 必须具备以下 2 个条件：

1）对变量的写操作不依赖于当前值

2）该变量没有包含在具有其他变量的不变式中

274.【简答题】【Java】请你说一下死锁的原因，以及如何打破，如何查看死锁进程状态

1、死锁是指在一组进程中的各个进程均占有不会释放的资源，但因互相申请被其他进程所站用不会释放的资源而处于的一种永久等待状态。死锁的四个必要条件：

- 互斥条件(Mutual exclusion)：资源不能被共享，只能由一个进程使用。
- 请求与保持条件(Hold and wait)：已经得到资源的进程可以再次申请新的资源。
- 非剥夺条件(No pre-emption)：已经分配的资源不能从相应的进程中被强制地剥夺。
- 循环等待条件(Circular wait)：系统中若干进程组成环路，该环路中每个进程都在等待相邻进程正占用的资源。

java 中产生死锁可能性的最根本原因是：1）是多个线程涉及到多个锁，这些锁存在着交叉，所以可能会导致了一个锁依赖的闭环；2）默认的锁申请操作是阻塞的。

如，线程在获得一个锁 L1 的情况下再去申请另外一个锁 L2，也就是锁 L1 想要包含了锁 L2，在获得了锁 L1，并且没有释放锁 L1 的情况下，又去申请获得锁 L2，这个是产生死锁的最根本原因。

2、避免死锁：

- 方案一：破坏死锁的循环等待条件。
- 方法二：破坏死锁的请求与保持条件，使用 lock 的特性，为获取锁操作设置超时时间。这样不会死锁（至少不会无尽的死锁）
- 方法三：设置一个条件遍历与一个锁关联。该方法只用一把锁，没有 chopstick 类，将竞争从对筷子的争夺转换成了对状态的判断。仅当左右邻座都没有进餐时才可以进餐。提升了并发度。

3、linux 中查看死锁进程状态

使用 pstack 和 gdb 工具对死锁程序进行分析

pstack 进程号 查看各个线程的堆栈信息

当进程吊死的时候，多次使用，死锁的线程将一直处于等锁的状态，确定某些线程一直没有变化，一直处于等锁的状态。那么这些线程很可能是死锁了。如果怀疑哪些线程发生死锁了，可以采用 gdb 进一步 attach 线程并进行分析。

执行命令 gdb attach 进程号，进入 gdb 调试终端

运行：(gdb) info thread

275.【简答题】【Java】请你说一下内存泄漏

当一个对象已经不需要再使用本该被回收时，另外一个正在使用的对象持有它的引用从而导致它不能被回收，这导致本该被回收的对象不能被回收而停留在堆内存中，这就产生了内存泄漏。内存泄漏是造成应用程序 OOM 的主要原因之一。我们知道 Android 系统为每个应用程序分配的内存是有限的，而当一个应用中产生的内存泄漏比较多时，这就难免会导致应用所需要的内存超过系统分配的内存限额，这就造成了内存溢出从而导致应用 Crash。

常见的内存泄漏：

1、单例造成的内存泄漏

由于单例的静态特性使得其生命周期和应用的生命周期一样长，如果一个对象已经不再需要使用了，而单例对象还持有该对象的引用，就会使得该对象不能被正常回收，从而导致了内存泄漏。

2、非静态内部类创建静态实例造成的内存泄漏

非静态内部类默认会持有外部类的引用，而该非静态内部类又创建了一个静态的实例，该实例的生命周期和应用的一样长，这就导致了该静态实例一直会持有该 Activity 的引用，从而导致 Activity 的内存资源不能被正常回收。

3、Handler 造成的内存泄漏

4、线程造成的内存泄漏

如果任务在 Activity 销毁之前还未完成，那么将导致 Activity 的内存资源无法被回收，从而造成内存泄漏。

5、资源未关闭造成的内存泄漏

对于使用了 BraodcastReceiver，ContentObserver，File，Cursor，Stream，Bitmap 等资源，应该在 Activity 销毁时及时关闭或者注销，否则这些资源将不会被回收，从而造成内存泄漏。

6、使用 ListView 时造成的内存泄漏

7、集合容器中的内存泄露

8、WebView 造成的泄露

避免内存泄漏：

1、在涉及使用 Context 时，对于生命周期比 Activity 长的对象应该使用 Application 的 Context。

2、对于需要在静态内部类中使用非静态外部成员变量（如：Context、View ），可以在静态内部类中使用弱引用来引用外部类的变量来避免内存泄漏。

3、对于不再需要使用的对象，显示的将其赋值为 null，比如使用完 Bitmap 后先调用 recycle()，再赋为 null。

4、保持对对象生命周期的敏感，特别注意单例、静态对象、全局性集合等的生命周期。

5、对于生命周期比 Activity 长的内部类对象，并且内部类中使用了外部类的成员变量，可以这样做避免内存泄漏：

1）将内部类改为静态内部类

2）静态内部类中使用弱引用来引用外部类的成员变量

276.【简答题】【Java】请你说一说 class 和 interface 的区别

1、接口类似于类，但接口的成员都没有执行方式，它只是方法、属性、事件和索引的组合而已，并且也只能包含这四种成员；类除了这四种成员之外还可以有别的成员(如字段)。

2、不能实例化一个接口，接口只包括成员的签名；而类可以实例化(abstract 类除外)。

3、接口没有构造函数，类有构造函数。

4、接口不能进行运算符的重载，类可以进行运算符重载。

5、接口的成员没有任何修饰符，其成员总是公共的，而类的成员则可以有修饰符(如：虚拟或者静态)。

6、派生于接口的类必须实现接口中所有成员的执行方式，而从类派生则不然。

277.【简答题】【Java】请你说一下内存泄漏的原因

当一个对象已经不需要再使用本该被回收时，另外一个正在使用的对象持有它的引用从而导致它不能被回收，这导致本该被回收的对象不能被回收而停留在堆内存中，这就产生了内存泄漏。内存泄漏是造成应用程序 OOM 的主要原因之一。我们知道 Android 系统为每个应用程序分配的内存是有限的，而当一个应用中产生的内存泄漏比较多时，这就难免会导致应用所需要的内存超过系统分配的内存限额，这就造成了内存溢出从而导致应用 Crash。

常见的内存泄漏：

1、单例造成的内存泄漏

由于单例的静态特性使得其生命周期和应用的生命周期一样长，如果一个对象已经不再需要使用了，而单例对象还持有该对象的引用，就会使得该对象不能被正常回收，从而导致了内存泄漏。

2、非静态内部类创建静态实例造成的内存泄漏

非静态内部类默认会持有外部类的引用，而该非静态内部类又创建了一个静态的实例，该实例的生命周期和应用的一样长，这就导致了该静态实例一直会持有该 Activity 的引用，从而导致 Activity 的内存资源不能被正常回

收。

3、Handler 造成的内存泄漏

4、线程造成的内存泄漏

如果任务在 Activity 销毁之前还未完成，那么将导致 Activity 的内存资源无法被回收，从而造成内存泄漏。

5、资源未关闭造成的内存泄漏

对于使用了 BraodcastReceiver，ContentObserver，File，Cursor，Stream，Bitmap 等资源，应该在 Activity 销毁时及时关闭或者注销，否则这些资源将不会被回收，从而造成内存泄漏。

6、使用 ListView 时造成的内存泄漏

7、集合容器中的内存泄露

8、WebView 造成的泄露

避免内存泄漏：

1、在涉及使用 Context 时，对于生命周期比 Activity 长的对象应该使用 Application 的 Context。

2、对于需要在静态内部类中使用非静态外部成员变量（如：Context、View ），可以在静态内部类中使用弱引用来引用外部类的变量来避免内存泄漏。

3、对于不再需要使用的对象，显示的将其赋值为 null，比如使用完 Bitmap 后先调用 recycle()，再赋为 null。

4、保持对对象生命周期的敏感，特别注意单例、静态对象、全局性集合等的生命周期。

5、对于生命周期比 Activity 长的内部类对象，并且内部类中使用了外部类的成员变量，可以这样做避免内存泄漏：

1）将内部类改为静态内部类

2）静态内部类中使用弱引用来引用外部类的成员变量

278.【简答题】【Java】请你说一说强引用和弱引用

强引用：

强引用是使用最普遍的引用。如果一个对象具有强引用，那垃圾回收器绝不会回收它。如下：

Object o=new Object(); // 强引用

当内存空间不足，Java 虚拟机宁愿抛出 OutOfMemoryError 错误，使程序异常终止，也不会靠随意回收具有强引用的对象来解决内存不足的问题。如果不使用时，要通过如下方式来弱化引用，如下：

o=null; // 帮助垃圾收集器回收此对象

显式地设置 o 为 null，或超出对象的生命周期范围，则 gc 认为该对象不存在引用，这时就可以回收这个对象。具体什么时候收集这要取决于 gc 的算法。

举例：

```
public void test(){
Object o=new Object();


}
```

在一个方法的内部有一个强引用，这个引用保存在栈中，而真正的引用内容（Object）保存在堆中。当这个方法运行完成后就会退出方法栈，则引用内容的引用不存在，这个 Object 会被回收。但是如果这个 o 是全局的变量时，就需要在不用这个对象时赋值为 null，因为强引用不会被垃圾回收。

弱引用：

弱引用也是用来描述非必需对象的，只具有弱引用的对象拥有更短暂的生命周期。在垃圾回收器线程扫描它所管辖的内存区域的过程中，一旦发现了只具有弱引用的对象，不管当前内存空间足够与否，都会回收它的内存。不过，由于垃圾回收器是一个优先级很低的线程，因此不一定会很快发现那些只具有弱引用的对象。提供 WeakReference 类实现弱引用。

例子：

String str=new String("abc");

WeakReference<String> abcWeakRef = new WeakReference<String>(str);

str=null;

当垃圾回收器进行扫描回收时等价于：

str = null;

System.gc();

如果这个对象是偶尔的使用，并且希望在使用时随时就能获取到，但又不想影响此对象的垃圾收集，那么你应该用 Weak Reference 来记住此对象。

String abc = abcWeakRef.get(); //该代码会让 str 再次变为一个强引用：

弱引用可以和一个引用队列（ReferenceQueue）联合使用，如果弱引用所引用的对象被垃圾回收，Java 虚拟机就会把这个弱引用加入到与之关联的引用队列中。当你想引用一个对象，但是这个对象有自己的生命周期，你不想介入这个对象的生命周期，这时候你就是用弱引用。这个引用不会在对象的垃圾回收判断中产生任何附加的影响。

279.【简答题】【Java】请说一下你对多态的理解

什么是多态？

多态就是指程序中定义的引用变量所指向的具体类型和通过该引用变量发出的方法调用在编程时并不确定，而是在程序运行期间才确定，即一个引用变量倒底会指向哪个类的实例对象，该引用变量发出的方法调用到底是哪个类中实现的方法，必须在由程序运行期间才能决定。因为在程序运行时才确定具体的类，这样，不用修改源程序代码，就可以让引用变量绑定到各种不同的类实现上，从而导致该引用调用的具体方法随之改变，即不修改程序代码就可以改变程序运行时所绑定的具体代码，让程序可以选择多个运行状态，这就是多态性。

Java 实现多态有三个必要条件：继承、重写、向上转型。

1. 继承：在多态中必须存在有继承关系的子类和父类。
2. 重写：子类对父类中某些方法进行重新定义，在调用这些方法时就会调用子类的方法。
3. 向上转型：在多态中需要将子类的引用赋给父类对象，只有这样该引用才能够具备技能调用父类的方法和子类的方法。

Java 中有两种形式可以实现多态，继承和接口：

1. 基于继承的实现机制主要表现在父类和继承该父类的一个或多个子类对某些方法的重写，多个子类对同一方法的重写可以表现出不同的行为。

2. 基于接口的多态中，指向接口的引用必须是指定这实现了该接口的一个类的实例程序，在运行时，根据对象引用的实际类型来执行对应的方法。

280.【简答题】【Java】请你说一下 java 里内存泄漏和溢出的区别

1、内存泄漏 memory leak :是指程序在申请内存后，无法释放已申请的内存空间，一次内存泄漏似乎不会有大的影响，但内存泄漏堆积后的后果就是内存溢出。

2、内存溢出 out of memory :指程序申请内存时，没有足够的内存供申请者使用，或者说，给了你一块存储 int 类型数据的存储空间，但是你却存储 long 类型的数据，那么结果就是内存不够用，此时就会报错 OOM,即所谓的内存溢出。

281.【简答题】【Java】请问你用过什么语言，用这些语言写过什么程序

java 虚拟机在执行 Java 程序的过程中会把它所管理的内存划分为若干个不同的数据区域。这些区域存储不同类型的数据，这些区域的内存分配和销毁的时间也不同，有的区域随着虚拟机进程的启动而存在，有些区域则是依赖用户线程的启动和结束而建立和销毁。根据《Java 虚拟机规范（第 2 版）》的规定，Java 虚拟机管理的内存包括五个运行时数据区域，

1、方法区

方法区（Method Area）是各个线程共享的内存区域，它用于存储已被虚拟机加载的类信息（包括类的名称、方法信息、成员变量信息）、常量、静态变量、以及编译器编译后的代码等数据

2、虚拟机栈和本地方法栈

虚拟机栈是线程私有的内存空间，每个线程都有一个线程栈，本地方法栈也是线程私有的内存空间，本地方法栈与 Java 栈所发挥的作用是非常相似的，它们之间的区别不过是 Java 栈执行 Java 方法，本地方法栈执行的是本地方法，有的虚拟机直接把本地方法栈和虚拟机栈合二为一。

3、堆

Java 堆是 Java 虚拟机所管理的内存中最大的一块，在虚拟机启动时创建，此内存区域的目的就是存放对象实例，几乎所有的对象实例都在这里分配内存。从内存分配的角度来看，线程共享的 Java 堆中可能划分出多个线程私有的分配缓冲区(TLAB)。Java 堆可以处于物理上不连续的内存空间，只要逻辑上连续即可，在实现上，既可以实现固定大小的，也可以是扩展的。如果堆中没有足够的内存分配给实例，并且堆也无法再拓展时，将会抛出 OutOfMemeryError 异常。

Java 内存回收：

对于栈空间，当方法调用结束后，基本类型变量，引用类型变量，形参占据的空寂会被自动释放，但引用类型指向的对象在堆中，堆中的无用内存有垃圾回收线程回收，GC 线程优先级最低，只有当没有工作线程存在时 GC 线程才会执行，或者堆空间不足时会自动出发 GC 线程工作，

282.【简答题】【Java】请你说一下 Java 里 integer 和 int 的区别，以及如何比较相等

Integer 和 int 的区别：

1、integer 是 int 的包装类，int 是 Java 的一种基本数据结构

2、integer 变量必须实例化后才能使用，int 变量不需要

3、integer 实际是对象的引用，int 是直接存储数据值

4、integer 的默认值是 null，int 的默认值是 0

如何比较相等，首先要明白 equals 和==的区别

Equals 通常用来比较两个对象的内容是否相等，==用来比较两个对象的地址是否相等，Object 类中的 equals 方法定义为判断两个对象的地址是否相等（可以理解成是否是同一个对象），地址相等则认为是对象相等。这也就意味着，我们新建的所有类如果没有复写 equals 方法，那么判断两个对象是否相等时就等同于“==”，也就是两个对象的地址是否相等。但在我们的实际开发中，通常会认为两个对象的内容相等时，则两个对象相等，equals 返回 true。对象内容不同，则返回 false。

所以可以总结为两种情况

1、类未复写 equals 方法，则使用 equals 方法比较两个对象时，相当于==比较，即两个对象的地址是否相等。地址相等，返回 true，地址不相等，返回 false。

2、类复写 equals 方法，比较两个对象时，则走复写之后的判断方式。通常，我们会将 equals 复写成：当两个对象内容相同时，则 equals 返回 true，内容不同时，返回 false。

283.【简答题】【Java】请你介绍一下 gc，另外如果 Java 里写一个方法，这个方法里只有一条语句，即 new 一个对象，请问方法结束以后这个对象怎么回收的？

GC（garbage collection）垃圾收集，是指 JVM 用于释放那些不再使用的对象所占用的内存，常用的 JVM 都有 GC，而且大多数 gc 都使用类似的算法管理内存和执行手机操作，GC 可以通过确定对象是否被活动对象引用来确定是否收集该对象，常用的方法有引用计数和对象引用遍历

284.【简答题】【Java】请你回答一下 protected，public，private 的区别

作用域	当前类	同一 package	子孙类	其它 package
public	√	√	√	√
protected	√	√	√	×
friendly	√	√	×	×
private	√	×	×	×

285.【简答题】【Java】请你说一下抽象类和接口的区别

从语法层面来说，

1、抽象类可以提供成员方法的实现细节，而接口中只能存在抽象方法

2、抽象类中成员变量可以是多种类型，接口中成员变量必须用 public，static，final 修饰

3、一个类只能继承一个抽象类，但可以实现多个接口

4、抽象类中允许含有静态代码块和静态方法，接口不能

从设计层面而言

1.抽象类是对整一个类的属性，行为等方面进行抽象，而接口则是对行为抽象。就好比飞机和鸟，抽象类抽象出的是飞行物类。而接口则是抽闲出飞行方法。

2.抽象类是一个模板式的设计，当在开发过程中出现需求更改的情况，只需要更改抽象类而不需要更改它的子类。接口是一种辐射性设计，当接口的内容发生改变时，需要同时对实现它的子类进行相应的修改。

3.抽象类可以类比为模板，而接口可以类比为协议

286.【简答题】【Java】请你说一下 List 和 ArrayList 的区别，以及 arrayList 和 HashSet 区别

List：是一个有序的集合，可以包含重复的元素。提供了按索引访问的方式。它继承 Collection。

List 有两个重要的实现类：ArrayList 和 LinkedList

ArrayList：我们可以将其看作是能够自动增长容量的数组。

利用 ArrayList 的 toArray()返回一个数组。

Arrays.asList()返回一个列表。

1.ArrayList 底层采用数组实现，当使用不带参数的构造方法生成 ArrayList 对象时，实际上会在底层生成一个长度为 10 的 Object 类型数组

2.如果增加的元素个数超过了 10 个，那么 ArrayList 底层会新生成一个数组，长度为原数组的 1.5 倍+1，然后将原数组的内容复制到新数组当中，并且后续增加的内容都会放到新数组当中。当新数组无法容纳增加的元素时，重复该过程。

3.对于 ArrayList 元素的删除操作，需要将被删除元素的后续元素向前移动，代价比较高。

4.集合当中只能放置对象的引用，无法放置原生数据类型，我们需要使用原生数据类型的包装类才能加入到集合当中。

5.集合当中放置的都是 Object 类型，因此取出来的也是 Object 类型，那么必须要使用强制类型转换将其转换为真正的类型（放置进去的类型）

287.【简答题】【Java】请你回答一下 Java 的内存结构是什么，全局变量，临时变量，静态变量分别存在哪里，堆分为哪几块，比如说新生代老生代，那么新生代又分为什么

代码区：是编译器生成的一个 exe 区段，存放函数体的二进制代码

栈区：存放函数的参数，局部变量的值等，其操作方式类似于数据结构中的栈，const 局部变量也是放在栈里

堆区：就是 malloc 和 new 之类的内存所在区段，一般由程序员分配释放，分配方式类似于链表

静态数据区：是编译器生成的一个 exe 区段，初始和未初始化的全局变量和局部变量都放在这里，

常量区：是编译器生成的一个 exe 区段，const 全局变量也放在常量区。

全局变量，临时变量，静态变量分别存在哪里

局部变量保存在栈中，全局变量和静态变量存储在静态数据区。

堆分为哪几块，比如说新生代老生代，那么新生代又分为什么？

java 垃圾收集管理器的主要区域，因此很多时候被称为“GC 堆”。

分为新生代和老年代；

新生代分为：Eden 和 Survivor。

288.【简答题】【Java】手写代码：给出一个 int 类型 123，写一个函数，返回反转的值 321

```
void my_itoa(int x,char *s,char radix)
{
 char zm[37]="0123456789abcdefghijklmnopqrstuvwxyz";
 int i=0;
 int sum=x;
 while(sum>0)
 {
 s[i++]=zm[sum%radix];
 sum/=radix;
 }
 }
 int atoi_my( char *str)
 {
 int s=0;
 while(*str>='0'&&*str<='9')
 {
 s=s*10+*str-'0';
 str++;
 if(s<0)
 {
 s=2147483647;
 break;
 }
 }
```

```
return s; //
}
```

289.【简答题】【Java】请你回答一下怎么判断哪些对象是可以删除的，可达是什么意思

判断方法有两种，一种是引用计数器，给对象添加一个引用计数器，每当有一个地方引用他时，计数器就加一，引用失败计数器减一，计数器为零的对象就不可能再被使用，

第二种方法是可达性分析算法。这个算法的基本思路就是通过一系列的称为"GC Roots"的对象作为起始点，从这些节点开始向下搜索，搜索走过的路径称为引用链。当一个对象到 GC Roots 没有任何引用链相连的时候，则证明此对象是不可用的。否则即是可达的。在 java 语言中，可作为 GC Roots 的对象包括下面几种：虚拟机栈中引用的对象、方法区中类静态属性引用的对象、方法区中常量引用的对象和本地方法栈中 JNI（一般说的 Native 方法）引用的对象。

290.【简答题】【Java】请你说一说接口有什么限制

1、变量会被隐式地指定为 public static final 变量，并且只能是 public static final 变量，用 private 修饰会报编译错误

2、方法会被隐式地指定为 public abstract 方法且只能是 public abstract 方法（用其他关键字，比如 private、protected、static、 final 等修饰会报编译错误）

3、接口中所有的方法不能有具体的实现，也就是说，接口中的方法必须都是抽象方法

291.【简答题】【Java】请问 Java 中线程如何实现，如何实现多线程，线程安全在 Java 中是如何实现的，线程的工作区是哪里

Java 中实现线程有三种方式：

1、1.继承 Thread 类

public class Thread extends Object implements Runnable

定义 Thread 类的子类，并重写 Thread 类的 run()方法，创建子类对象（即线程对象），调用线程对象的 start()方法来启动该线程。

2.实现 Runnable 接口

public interface Runnable

定义 Runnable 接口的实现类，并重写该接口的 run()方法，该 run()方法同样是该线程的执行体。创建该 Runnable 实现类的实例，并将此实例作为 Thread 的 target（即构造函数中的参数）来创建 Thread 对象（该 Thread 对象才是真正的线程对象，只是该 Threa

3.使用 Callable 和 Future

创建 Callable 接口的实现类，并实现 call()方法，该方法有返回值；创建 Callable 实现类的实例，使用

FutureTask 来包装 Callable 对象，并且也封装了 call()方法的返回值；使用 FutureTask 作为 Thread 类的 target 创建并启动线程；调用 FutureTask 对象的 get()方法返回子线程执行结束后的返回值。

如何实现多线程：

首先是继承 Thread 类并重写 run（）方法

```
package com.csu.multiThread;
public class MultiThreadExtendsThread extends Thread{
String name;
public MultiThreadExtendsThread(String name) {
this.name = name;
}
public void run() {
for(int i=0;i<5;i++) {
System.out.println(name+"运行： "+i);
}
}
public static void main(String[] args) {
MultiThreadExtendsThread thread1 = new MultiThreadExtendsThread("A");
MultiThreadExtendsThread thread2 = new MultiThreadExtendsThread("B");
thread1.start();
thread2.start();

}
```

二是实现 Runnable 接口，然后重写 run()方法，

```
package com.csu.multiThread;



public class MultiThreadImplRunnable implements Runnable{
String name;
public MultiThreadImplRunnable(String name) {
this.name = name;
}

@Override
public void run() {
for(int i=0;i<5;i++) {
System.out.println(name+"运行： "+i);
```

```
}
}
public static void main(String[] args) {
MultiThreadExtendsThread thread1 = new MultiThreadExtendsThread("A");
MultiThreadExtendsThread thread2 = new MultiThreadExtendsThread("B");
new Thread(thread1).start();
new Thread(thread2).start();
}
}
```

线程安全的实现：

最基本的：synchronized 关键字。这个方法是最常用的，它通过互斥的方式保证同步。我们知道 java 中有几个操作是可以保证原子性的，其中 lock/unlock 就是一对。虽然 java 没有提供这两个字节码的接口，但是我们可以通过 monitorenter/monitorexit，而 synchronized 会在块的前后调用两个字节码指令。同时 synchronize 对于同一条线程来说是可重入的；其次它也是阻塞的。我们知道 java 线程是映射到操作系统上的，而且是混用的内核态线程和用户态线程（N:M），而将线程从阻塞/唤醒，需要将线程从用户态转换到内核态，这样会消耗太亮的资源，所以 synchronize 是一个重量级锁

另外一种和 synchronize 类似的方法：ReentrantLock。它们两个的区别：（1）synchronize 是隐式的，只要块内的代码执行完，就会释放当前的锁；而后者需要显式的调用 unlock()方法手动释放，所以经常搭配 try/finally 方法（忘记在 finally 中 unlock 是非常危险的） （2）后者可以选择等待中断——即在当前持有锁线程长期不释放锁的情况下，正在等待的线程可以选择放弃等待选择处理其他的事情。 （3） 后者可以选择公平锁（虽然默认是非公平的，因为公平锁的吞吐量很受影响）即先来后到，按申请的顺序获得锁。 （4）可以绑定多个条件

前面提到的两种方式都是通过互斥来达到同步的目的，这其实是悲观锁的一种。下面介绍的是乐观锁，基于冲突检测的并发策略，不需要将线程挂起，因此又被成为非阻塞同步。

典型：CAS（Compare And Swap），通过 Unsafe 类提供。有三个操作数，内存位置、旧的预期值、和新的值；当且仅当内存地址 V 符合预期值 A 时，执行将值更新为新的预期值 B。 存在的问题：“ABA”情况，即原值为 A，但在检测之前发生了改变，变成了 B，同时也在检测时变回了 A；即不能保证这个值没有被其他线程更改过。

接下来是无同步方案：

可重入代码（纯代码）：是一种无同步方案，在执行的任何时候去中断，转而执行其他的代码；在重新返回代码后，不会出现任何的错误。可重入性->线程安全，充分不必要条件。即可重入性的代码都是线程安全的，但反之不一定。简单判断原则：一个方法的返回结果是可预测的，只要输入了相同的数据，就都能返回相同的结果。

线程本地存储：即利用 ThreadLocal 类；每个 Thread 类中都有一个变量 ThreadLocalMap，默认是为 null 的。它将为每一个线程创立一个该变量的副本。这样线程之间就不存在数据征用的问题了。适用情况：（1）数据库的 Connection 连接 （2）WEB 中的“一个请求对应一个服务器线程”，在知乎上看到一个回答，解释的蛮清晰的。（3）Spring 中创建的默认模式是 Singleton 单例 （4）“生产者-消费者问题”

ThreadLocal 就是变量在不同线程上的副本，不同线程不共享，所以对变量改动时就不需要考虑线程间同步的问

题了

ThreadLocal 在 web 应用开发中是一种很常见的技巧，当 web 端采用无状态写法时（比如 stateless session bean 和 spring 默认的 singleton），就可以考虑把一些变量放在 ThreadLocal 中

举个简单例子，以理解意思为主：你有两个方法 A 和 B 都要用到变量 userId，又不想传来传去，一个很自然的想法就是把 userId 设为成员变量，但是在无状态时，这样做就很可能有问题，因为多个 request 在同时使用同一个 instance，userId 在不同 request 下值是不一样的，就会出现逻辑错误

但由于同一个 request 下一般都是处于同一个线程，如果放在 ThreadLocal 的话，这个变量就被各个方法共享了，而又不影响其他 request，这种情况下，你可以简单把它理解为是一种没有副作用的成员变量(作者：卡斯帕尔)

线程栈线程的每个方法被执行的时候，都会同时创建一个帧（Frame）用于存储本地变量表、操作栈、动态链接、方法出入口等信息。每一个方法的调用至完成，就意味着一个帧在 VM 栈中的入栈至出栈的过程。如果线程请求的栈深度大于虚拟机所允许的深度，将抛出 StackOverflowError 异常；如果 VM 栈可以动态扩展（VM Spec 中允许固定长度的 VM 栈），当扩展时无法申请到足够内存则抛出 OutOfMemoryError 异常。

292.【简答题】【Java】请你说一说内存溢出和内存泄漏是怎么回事

内存溢出 out of memory，是指程序在申请内存时，没有足够的内存空间供其使用，出现 out of memory；比如申请了一个 integer,但给它存了 long 才能存下的数，那就是内存溢出。

内存泄露 memory leak，是指程序在申请内存后，无法释放已申请的内存空间，一次内存泄露危害可以忽略，但内存泄露堆积后果很严重，无论多少内存,迟早会被占光。

内存泄漏可以分为四类：

1、常发性内存泄漏，发生内存泄漏的代码会被多次执行到，每次执行都会导致内存泄漏

2、偶发性内存泄漏：发生内存泄漏的代码只有在某些特定环境或操作过程下才会发生，

3、一次性内存泄漏，发生内存泄漏的代码只会被执行一次，或者由于算法上的缺陷，导致总会有一块仅且一块内存发生泄漏。

4、隐式内存泄漏。程序在运行过程中不停的分配内存，但是直到结束的时候才释放内存。

从用户使用程序的角度来看，内存泄漏本身不会产生什么危害，真正有危害的是内存泄漏的堆积，这会最终消耗尽系统所有的内存。

内存溢出常见原因：

1.内存中加载的数据量过于庞大，如一次从数据库取出过多数据；

2.集合类中有对对象的引用，使用完后未清空，使得 JVM 不能回收；

3.代码中存在死循环或循环产生过多重复的对象实体；

4.使用的第三方软件中的 BUG；

5.启动参数内存值设定的过小

解决方案：

1、修改 JVM 参数，直接增加内存

2、检查错误日志，查看内存溢出错误前是否有其他异常错误

3、对代码进行走查分析，找出可能发生内存溢出的位置

293.【简答题】【Java】请你介绍一下 HashMap，HashTable，ConcurrentHashMap

1、HashTable 与 HashMap

(1)HashTable 和 HashMap 都实现了 Map 接口，但是 HashTable 的实现是基于 Dictionary 抽象类。

(2)在 HashMap 中，null 可以作为键，这样的键只有一个；可以有一个或多个键所对应的值为 null。当 get()方法返回 null 值时，既可以表示 HashMap 中没有该键，也可以表示该键所对应的值为 null。因此，在 HashMap 中不能由 get()方法来判断 HashMap 中是否存在某个键，而应该用 containsKey()方法来判断。而在 HashTable 中，无论是 key 还是 value 都不能为 null。

(3)HashTable 是线程安全的，它的方法是同步了的，可以直接用在多线程环境中。HashMap 则不是线程安全的。在多线程环境中，需要手动实现同步机制。

2、更好的选择：ConcurrentHashMap Java 5 中新增了 ConcurrentMap 接口和它的实现类 ConcurrentHashMap。ConcurrentHashMap 提供了和 HashTable 以及 SynchronizedMap 中所不同的锁机制。HashTable 中采用的锁机制是一次锁住整个 hash 表，从而同一时刻只能有一个线程对其进行操作；而 ConcurrentHashMap 中则是一次锁住一个桶。ConcurrentHashMap 默认将 hash 表分为 16 个桶，诸如 get，put，remove 等常用操作只锁住当前需要用到的桶。这样，原来只能一个线程进入，现在却能同时有 16 个写线程执行，并发性能的提升是显而易见的。 上面说到的 16 个线程指的是写线程，而读操作大部分时候都不需要用到锁。只有在 size 等操作时才需要锁住整个 hash 表。 在迭代方面，ConcurrentHashMap 使用了一种不同的迭代方式。在这种迭代方式中，当 iterator 被创建后集合再发生改变就不再是抛出 ConcurrentModificationException，取而代之的是，在改变时 new 新的数据从而不影响原有的数据，iterator 完成后再将头指针替换为新的数据，这样 iterator 可以使用原来老的数据，而写线程也可以并发的完成改变。

294.【简答题】【Java】请你说一下 Hashset 有什么特性，以及 hashset 判断存入的对象是否重复是如何比较的

HashSet 是 Set 接口的实现类，因此，HashSet 中的元素也是不能重复的。HashCode 判断元素重复的标准时，首先计算新添加元素的 hashCode 值，当不重复是，则直接加入到该集合中，若发生重复，也称发生了碰撞，则进一步调用 equals 判断元素是否在逻辑上相同。

295.【简答题】【Java】请你说一下 Java 的反射，你目前主要用他做什么，以及 Java 的泛型，他的主要作用是什么

JAVA 反射机制是在运行状态中，对于任意一个类，都能够知道这个类的所有属性和方法；对于任意一个对象，都能够调用它的任意方法和属性；这种动态获取信息以及动态调用对象方法的功能称为 java 语言的反射机制。

Java 反射可以用来获取一个 class 对象或实例化一个 class 表示的类的对象，还可以获取构造方法，成员变量，成员方法。

java 中泛型的引入主要是为了解决两个方面的问题：1.集合类型元素在运行期出现类型装换异常，增加编译时类型的检查，2. 解决的时重复代码的编写，能够复用算法。下面通过例子来说明编译器的类型检查。

296.【简答题】【Java】请问类加载器你了解吗

类加载器就是把类加载阶段中的”通过一个类的全限定名来获取描述此类的二进制字节流”这个动作放到 java 虚拟机外部来实现的代码模块。

类加载器的分类：

启动类加载器、扩展类加载器、应用类加载器（系统类加载器）、用户自定义类加载器。

启动类加载器：这个类负责将存放在 JAVA_HOME/lib 目录或者被-Xbootclasspath 参数所指定的路径中的并且是虚拟机内存中。

扩展类加载器：负责加载 JAVA_HOME/lib/ext 目录中或者被 java.ext.dirs 系统变量指定路径中的所有类库，开发者可以直接使用扩展类加载器。

应用程序类加载器：负责加载用户类路径上指定的类加载器，一般情况下就是程序中默认的类加载器。

297.【简答题】【Java】ReentranceLock 和 synchronized 有什么区别

1.可重入性

字面的意思就是可以再次进入的锁，synchronized 其实也是可重锁，同一线程每进入一次，锁的计数器都会加一，在释放锁是计数器都会减一，只有当计数器为 0 时才能释放锁

2.锁的实现

ReentrantLock 是 JDK 实现的 Synchronized 是 JVM 实现

前者可以直接看到源码，后者实现难以看到

3.性能的区别

在 Synchronized 优化以前，synchronized 的性能是比 ReenTrantLock 差很多的，但是自从 Synchronized 引入了偏向锁，轻量级锁（自旋锁）后，两者的性能就差不多了，在两种方法都可用的情况下，官方甚至建议使用 synchronized，其实 synchronized 的优化我感觉就借鉴了 ReenTrantLock 中的 CAS 技术。都是试图在用户态就把加锁问题解决，避免进入内核态的线程阻塞。

4.功能的区别

便利性：很明显 Synchronized 的使用比较方便简洁，并且由编译器去保证锁的加锁和释放，而 ReenTrantLock 需要手工声明来加锁和释放锁，为了避免忘记手工释放锁造成死锁，所以最好在 finally 中声明释放锁。

锁的细粒度和灵活度：很明显 ReenTrantLock 优于 Synchronized

当你需要时候一下三种功能是需要使用 ReentrantLock

ReentranLock 可以指定公平锁还是非公平锁

（公共锁就是先等待的线程先获得锁）

实现自旋锁，通过循环调用 CAS 操作来实现加锁，性能比较好，避免进入内核态的线程阻塞。

提供了 Condition 类，可以分组唤醒需要唤醒的线程

提供能够中断等待锁的线程的机制，lock.lockInterruptibly()

具体使用场景要根据实际的业务进行分析

使用 Synchronized 时不需要释放锁，jvm 会帮助我们做释放锁的操作

298.【简答题】【C 语言】请你说一下数组和指针的区别

数组：数组是用于储存多个相同类型数据的集合。

指针：指针相当于一个变量，但是它和不同变量不一样，它存放的是其它变量在内存中的地址。

区别：

- 赋值：同类型指针变量可以相互赋值，数组不行，只能一个一个元素的赋值或拷贝
- 存储方式：数组：数组在内存中是连续存放的，开辟一块连续的内存空间。数组是根据数组的下进行访问的，多维数组在内存中是按照一维数组存储的，只是在逻辑上是多维的。指针：指针很灵活，它可以指向任意类型的数据。指针的类型说明了它所指向地址空间的内存。
- 求 sizeof：数组所占存储空间的内存：sizeof（数组名），数组的大小：sizeof（数组名）/sizeof（数据类型）。在 32 位平台下，无论指针的类型是什么，sizeof（指针名）都是 4，在 64 位平台下，无论指针的类型是什么，sizeof（指针名）都是 8。
- 初始化方式不同。
- 传参方式：数组传参时，会退化为指针，C 语言将数组的传参进行了退化。将整个数组拷贝一份传入函数时，将数组名看做常量指针，传数组首元素的地址。一级指针传参可以接受的参数类型：（1）可以是一个整形指针（2）可以是整型变量地址 （3）可以是一维整型数组数组名；当函数参数部分是二级指针时，可以接受的参数类型：（1）二级指针变量（2）一级指针变量地址（3）一维指针数组的数组名

299.【简答题】【C 语言】请你说一说 STL 常用的容器

（1）vector

vector 是一种动态数组，在内存中具有连续的存储空间，支持快速随机访问。由于具有连续的存储空间，所以在插入和删除操作方面，效率比较慢。vector 有多个构造函数，默认的构造函数是构造一个初始长度为 0 的内存空间，且分配的内存空间是以 2 的倍数动态增长的，即内存空间增长是按照 20,21,22,23.....增长的，在 push_back 的过程中，若发现分配的内存空间不足，则重新分配一段连续的内存空间，其大小是现在连续空间的 2 倍，再将原先空间中的元素复制到新的空间中，性能消耗比较大，尤其是当元素是非内部数据时(非内部数据往往构造及拷贝构造函数相当复杂)。

API：

vector <T> v;//采用模板实现类实现，默认构造函数

assign(begin(),end());//将【begin(),end()】区间中的元素拷贝给本身

size();//返回元素容器中元素个数

at(int idx);//返回索引 idx 所指的数据，如果 idx 越界，抛出 out_of_range 异常

（2）deque

deque 和 vector 类似，支持快速随机访问。二者最大的区别在于，vector 只能在末端插入数据，而 deque 支持双端插入数据。deque 的内存空间分布是小片的连续，小片间用链表相连，实际上内部有一个 map 的指针。deque 空间的重新分配要比 vector 快，重新分配空间后，原有的元素是不需要拷贝的。

API：

deque<T>deqT;//默认构造形式

assign(begin,end);//将【begin，end】区间的数据拷贝赋值给自身

deque.size();//返回容器中元素的个数

push_back(elem);//在容器尾部添加一个数据

（3）list

list 是一个双向链表，因此它的内存空间是可以不连续的，通过指针来进行数据的访问，这使 list 的随机存储变得非常低效，因此 list 没有提供[]操作符的重载。但 list 可以很好地支持任意地方的插入和删除，只需移动相应的指针即可。

API：

list<T>lstT;//list 采用模板类实现对象的默认构造函数

push_back(elem);//在容器尾部加入一个元素

size();//返回元素容器中元素个数

empty();//判断容器是否为空

（4）map

map 是一种关联容器，该容器用唯一的关键字来映射相应的值，即具有 key-value 功能。map 内部自建一棵红黑树（一种自平衡二叉树），这棵树具有数据自动排序的功能，所以在 map 内部所有的数据都是有序的，以二叉树的形式进行组织。

API：

map<t1,T2>mapT;//map 默认构造函数

size();//返回元素中元素的数目

empty();//判断容器是否为空

clear();//删除所有元素

（5）set

set 也是一种关联性容器，它同 map 一样，底层使用红黑树实现，插入删除操作时仅仅移动指针即可，不涉及内存的移动和拷贝，所以效率比较高。set 中的元素都是唯一的，而且默认情况下会对元素进行升序排列。所以在 set 中，不能直接改变元素值，因为那样会打乱原本正确的顺序，要改变元素值必须先删除旧元素，再插入新元素。不提供直接存取元素的任何操作函数，只能通过迭代器进行间接存取。

API：

set<T>sT;//set 默认构造函数

size();//返回容器中元素的数目

empty();//判断容器是否为空

insert(elem);//在容器中插入元素

clear();//清除所有元素

（6）queue

queue 是一个队列，实现先进先出功能，queue 不是标准的 STL 容器，却以标准的 STL 容器为基础。queue 是在 deque 的基础上封装的。之所以选择 deque 而不选择 vector 是因为 deque 在删除元素的时候释放空间，同时在重新申请空间的时候无需拷贝所有元素。

API：

queue<T>queT;//queue 采用模板类实现，queue 对象的默认构造形式

push(elem);//往队尾添加元素

（7）stack

stack 是实现先进后出的功能，和 queue 一样，也是内部封装了 deque，这也是为啥称为容器适配器的原因吧（纯属猜测）。自己不直接维护被控序列的模板类，而是它存储的容器对象来为它实现所有的功能。stack 的源代码原理和实现方式均跟 queue 相同。

API：

stack<T>stkT;//采用模板类实现，stack 对象的默认构造形式

push(elem);//向栈顶添加元素

300.【简答题】【C 语言】请你说一下虚函数

拥有 Virtual 关键字的函数称之为虚函数，虚函数的作用是实现动态绑定的，也就是说程序在运行的时候动态的的选择合适的成员函数。要成为虚函数必须满足两点，一就是这个函数依赖于对象调用，因为虚函数就是依赖于对象调用，因为虚函数是存在于虚函数表中，有一个虚函数指针指向这个虚表，所以要调用虚函数，必须通过虚函数指针，而虚函数指针是存在于对象中的。二就是这个函数必须可以取地址，因为我们的虚函数表中存放的是虚函数函数入口地址，如果函数不能寻址，就不能成为虚函数。

301.【简答题】【C 语言】请你说一下动态内存分配

在程序执行的过程中动态地分配或者回收存储空间的分配内存的方法。动态内存分配不象数组等静态内存分配方法那样需要预先分配存储空间，而是由系统根据程序的需要即时分配，且分配的大小就是程序要求的大小。

302.【简答题】【C 语言】请你说一下深 copy 浅 copy

浅拷贝是增加了一个指针，指向原来已经存在的内存。而深拷贝是增加了一个指针，并新开辟了一块空间让指

针指向这块新开辟的空间。浅拷贝在多个对象指向一块空间的时候，释放一个空间会导致其他对象所使用的空间也被释放了，再次释放便会出现错误。

303.【简答题】【C 语言】请你说一下 C 中申请和释放内存的方法

C++:

new 运算符申请内存：

将调用相应的 operator new(size_t) 函数动态分配内存，在分配到的动态内存块上 初始化 相应类型的对象（构造函数）并返回其首地址。如果调用构造函数初始化对象时抛出异常，则自动调用 operator delete(void*, void*) 函数释放已经分配到的内存。

delete 运算符释放内存：

调用相应类型的析构函数，处理类内部可能涉及的资源释放，调用相应的 operator delete(void *) 函数。

C：

内存区域可以分为栈，堆，静态存储区和常量存储区。局部变量，函数形参，临时变量都是在栈上获得内存的，它们获取的方式都是由编译器自动执行的。

而 C 标准函数库提供了许多函数来实现对堆上内存管理，其中包括：malloc 函数，free 函数，calloc 函数和 realloc 函数。使用这些函数需要包含头文件 stdlib.h。

（1）malloc 函数

malloc 函数可以从堆上获得指定字节的内存空间，其函数声明如下：

void * malloc(int n);

其中，形参 n 为要求分配的字节数。如果函数执行成功，malloc 返回获得内存空间的首地址；如果函数执行失败，那么返回值为 NULL。由于 malloc 函数值的类型为 void 型指针，因此，可以将其值类型转换后赋给任意类型指针，这样就可以通过操作该类型指针来操作从堆上获得的内存空间。需要注意的是,malloc 函数分配得到的内存空间是未初始化的。因此，一般在使用该内存空间时，要调用另一个函数 memset 来将其初始化为全 0。memset 函数的声明如下：

void * memset (void * p,int c,int n) ;

该函数可以将指定的内存空间按字节单位置为指定的字符 c。其中，p 为要清零的内存空间的首地址，c 为要设定的值，n 为被操作的内存空间的字节长度。

（2）free 函数

从堆上获得的内存空间在程序结束以后，系统不会将其自动释放，需要程序员来自己管理。一个程序结束时，必须保证所有从堆上获得的内存空间已被安全释放，否则，会导致内存泄露。

void free (void * p);

由于形参为 void 指针，free 函数可以接受任意类型的指针实参。

但是，free 函数只是释放指针指向的内容，而该指针仍然指向原来指向的地方，此时，指针为野指针，如果此时操作该指针会导致不可预期的错误。安全做法是：在使用 free 函数释放指针指向的空间之后，将指针的值置为 NULL。

（3）calloc 函数

calloc 函数的功能与 malloc 函数的功能相似，都是从堆分配内存。其函数声明如下：

void *calloc(int n,int size);

函数返回值为 void 型指针。如果执行成功，函数从堆上获得 size X n 的字节空间，并返回该空间的首地址。如果执行失败，函数返回 NULL。该函数与 malloc 函数的一个显著不同时是，**calloc 函数得到的内存空间是经过初始化的，其内容全为 0。**calloc 函数适合为数组申请空间，可以将 size 设置为数组元素的空间长度，将 n 设置为数组的容量。

（4）realloc 函数

realloc 函数的功能比 malloc 函数和 calloc 函数的功能更为丰富，可以实现内存分配和内存释放的功能，其函数声明如下：

void * realloc(void * p,int n);

其中，指针 p 必须为指向堆内存空间的指针，即由 malloc 函数、calloc 函数或 realloc 函数分配空间的指针。realloc 函数将指针 p 指向的内存块的大小改变为 n 字节。如果 n 小于或等于 p 之前指向的空间大小，那么。保持原有状态不变。如果 n 大于原来 p 之前指向的空间大小，那么，系统将重新为 p 从堆上分配一块大小为 n 的内存空间，同时，将原来指向空间的内容依次复制到新的内存空间上，p 之前指向的空间被释放。relloc 函数分配的空间也是未初始化的。

304.【简答题】【C 语言】请你说一说 C++和 C 的区别

- C 是面向过程的语言，而 C++是面向对象的语言
- C 和 C++动态管理内存的方法不一样，C 是使用 malloc/free 函数，而 C++除此之外还有 new/delete 关键字
- C 中的 struct 和 C++的类，C++的类是 C 所没有的，但是 C 中的 struct 是可以在 C++中正常使用的，并且 C++对 struct 进行了进一步的扩展，使 struct 在 C++中可以和 class 一样当做类使用，而唯一和 class 不同的地方在于 struct 的成员默认访问修饰符是 public,而 class 默认的是 private;
- C++支持函数重载，而 C 不支持函数重载，而 C++支持重载的依仗就在于 C++的名字修饰与 C 不同，例如在 C++中函数 int fun(int ,int)经过名字修饰之后变为 _fun_int_int ,而 C 是 _fun，一般是这样的，所以 C++才会支持不同的参数调用不同的函数；
- C++中有引用，而 C 没有；
- C++全部变量的默认链接属性是外链接，而 C 是内连接；
- C 中用 const 修饰的变量不可以用在定义数组时的大小，但是 C++用 const 修饰的变量可以

305.【简答题】【C 语言】请你回答一下 C++中的多态是怎么实现的

C++的多态性是用虚函数和延迟绑定来实现的，在基类的函数前加上 virtual 关键字，在派生类中重写该函数，运行时将会根据对象的实际类型来调用相应的函数。如果对象类型是派生类，就调用派生类的函数；如果对象类型是基类，就调用基类的函数。

1. 用 virtual 关键字申明的函数叫做虚函数，虚函数肯定是类的成员函数。

2. 存在虚函数的类都有一个一维的虚函数表叫做虚表。类的对象有一个指向虚表开始的虚指针。虚表是和类对应的，虚表指针是和对象对应的。

3. 多态性是一个接口多种实现，是面向对象的核心。分为类的多态性和函数的多态性。

4. 多态用虚函数来实现，结合动态绑定。

5. 纯虚函数是虚函数再加上= 0。

6. 抽象类是指包括至少一个纯虚函数的类。

306.【简答题】【C 语言】请你说一下 C 语言的内存分配

在 C 语言中，对象可以使用静态或动态的方式分配内存空间

静态分配：编译器在处理程序源代码时分配，由于是在程序执行之前进行所以效率比较高

动态分配：程序在执行时调用 malloc 库函数申请分配，可以灵活的处理未知数目的对象

静态与动态内存分配的主要区别如下：

静态对象是有名字的变量，可以直接对其进行操作；动态对象是没有名字的变量，需要通过指针间接地对它进行操作。

静态对象的分配与释放由编译器自动处理；动态对象的分配与释放必须由程序员显式地管理，它通过 malloc() 和 free 两个函数（C++中为 new 和 delete 运算符）来完成。

307.【简答题】【C 语言】请你回答一下什么是指针，以及指针和数组的区别，指针和引用的区别

指针就是一个存放地址的变量，当指针指向某个变量，这时这个指针里就存放了那个变量的地址

指针与数组的区别

1、指针和数组的分配，数组是开辟一块连续的内存空间，指针不是

2、空间分配，　　这里又分为两种情况。

第一，如果是全局的和静态的

char *p = “hello”;

这是定义了一个指针，指向 rodata section 里面的“hello”，可以被编译器放到字符串池。在汇编里面的关键字为.ltorg。意思就是在字符串池里的字符串是可以共享的，这也是 编译器优化的一个措施。

char a[] = “hello”;

这是定义了一个数组，分配在可写数据块，不会被放到字符串池。

第二，如果是局部的

char *p = “hello”;

这是定义了一个指针，指向 rodata section 里面的“hello”，可以被编译器放到字符串池。在汇编里面的关键字为.ltorg。意思就是在字符串池里的字符串是可以共享的，这也是 编译器优化的一个措施。另外，在函数中可以返回它的地址，也就是说，指针是局部变量，但是它指向的内容是全局的。

char a[] = “hello”;

这是定义了一个数组，分配在堆栈上，初始化由编译器进行。（短的时候直接用指令填充，长的时候就从全局字符串表拷贝），不会被放到字符串池（同样如前，可 能会从字符串池中拷贝过来）。注意不应该返回它的地址。

3、使用方法

如果是全局指针，用于不需要修改内容，但是可能会修改指针的情况。如果是全局数组，用于不需要修改地址，但是却需要修改内容的情况。如果既需要修改指针，又需要修改内容，那么就定义一个数组，再定义一个指针指向它就可以了。

指针和引用的区别

1、指针是一个变量，只不过这个变量存储的是一个地质，而引用跟原来的变量实质上是一个东西，只不过是原变量的一个别名

2、引用不可以为空，当被创建的时候，必须初始化，而指针可以是空

3、指针可以有多级，但是引用只有一级

4、指针的值在初始化后可以改变，但是引用进行初始化后就不会再改变了

5、sizeof 引用得到的是指向变量的大小，而指针得到的是本身的大小

6、如果返回动态分配的对象或内存，必须使用指针，否则可能引起内存泄漏

308.【简答题】【C 语言】请你说一下 const 和指针的区别，以及运算符优先级

const 和指针的区别

以下三个语法的区别

(1) const int *p1;

(2) int const *p2;

(3) int *const p3;

由于指针*p 的赋值方式有两种

第一种是：p = &a;

第二种是：*p = a;

(1)和(2)的效果是一样的。只能通过第一种方式来修改指针指向的变量

而(3)的方式是只能在一开始的时候指定一个变量，以后不能再指向其他变量。

其实主要是看 const 后面的变量是什么，只有 const 后面的变量无法修改

运算符优先级，简单来说就是！>算术运算符>关系运算符>&&>||>赋值运算符

309.【简答题】【C 语言】手写代码：写一个程序算出 100 以内 7 的倍数

```
void beishu()
{
  int i ;
```

```
i=7;
while(i<=100)
{
if((i%7)==0)
printf("%d",i);
i+=7;
}
}
```

310.【简答题】【C 语言】手写代码：写一个函数，不用加法和乘法，返回他的八倍

```
import org.junit.Test;
public class solution {
 @Test
 public void testFunc(){
 int i = 1;
 int res = beiShu(i);
 System.out.println("res: "+res);
 }
 public int beiShu(int x){
 return x<<3;
 }
  }
```

311.【简答题】【C 语言】请你说一下 new 和 malloc 的区别

1、属性：new 是 C++关键字，需要编译器支持，malloc 是库函数，需要头文件支持

2、参数：使用 new 操作符申请内存分配时无需指定内存块的大小，编译器会根据类型信息自行计算，malloc 则需要显式的指出所需内存的尺寸

3、返回类型:new 返回的是对象类型的指针，malloc 返回 void*，需要通过强制类型转换将 void*指针转换成我们需要的类型

4、分配失败：new 内存分配失败时，会跑出 bac_alloc 异常，malloc 分配内存失败返回 null

5、 自定义类型

new 会先调用 operator new 函数，申请足够的内存（通常底层使用 malloc 实现）。然后调用类型的构造函数，初始化成员变量，最后返回自定义类型指针。delete 先调用析构函数，然后调用 operator delete 函数释放内存（通常底层使用 free 实现）。

malloc/free 是库函数，只能动态的申请和释放内存，无法强制要求其做自定义类型对象构造和析构工作。

6、 重载

C++允许重载 new/delete 操作符，特别的，布局 new 的就不需要为对象分配内存，而是指定了一个地址作为内存起始区域，new 在这段内存上为对象调用构造函数完成初始化工作，并返回此地址。而 malloc 不允许重载。

7、内存区域

new 操作符从自由存储区（free store）上为对象动态分配内存空间，而 malloc 函数从堆上动态分配内存。自由存储区是 C++基于 new 操作符的一个抽象概念，凡是通过 new 操作符进行内存申请，该内存即为自由存储区。而堆是操作系统中的术语，是操作系统所维护的一块特殊内存，用于程序的内存动态分配，C 语言使用 malloc 从堆上分配内存，使用 free 释放已分配的对应内存。自由存储区不等于堆，如上所述，布局 new 就可以不位于堆中。

312.【简答题】【C 语言】请你说一说 C++语言的三大特性

封装，继承，多态

313.【简答题】【C 语言】请你说一说虚函数和纯虚函数区别

虚函数和纯虚函数区别

观点一:类里声明为虚函数的话,这个函数是实现的，哪怕是空实现，它的作用就是为了能让这个函数在它的子类里面可以被重载，这样的话，这样编译器就可以使用后期绑定来达到多态了

纯虚函数只是一个接口，是个函数的声明而已，它要留到子类里去实现。

```
class A{
protected:
void foo();//普通类函数
virtual void foo1();//虚函数
virtual void foo2() = 0;//纯虚函数
}
```

观点二：虚函数在子类里面也可以不重载的；但纯虚必须在子类去实现，这就像 Java 的接口一样。通常我们把很多函数加上 virtual，是一个好的习惯，虽然牺牲了一些性能，但是增加了面向对象的多态性，因为你很难预料到父类里面的这个函数不在子类里面不去修改它的实现

观点三：虚函数的类用于“实作继承”，继承接口的同时也继承了父类的实现。当然我们也可以完成自己的实现。纯虚函数的类用于“介面继承”，主要用于通信协议方面。关注的是接口的统一性，实现由子类完成。一般来说，介面类中只有纯虚函数的。

观点四：带纯虚函数的类叫虚基类，这种基类不能直接生成对象，而只有被继承，并重写其虚函数后，才能使用。这样的类也叫抽象类。

虚函数是为了继承接口和默认行为

纯虚函数只是继承接口，行为必须重新定义

314.【简答题】【C 语言】请你说一下 static 作用

Static 作用：

1、隐藏，当同时编译多个文件时，所有未加 static 的全局变量和函数都具有全局可见性。

2、保持变量内容的持久，存储在静态数据区的变量会在程序放开是运行时就完成初始化，也是唯一一次初始化，共有两种变量存储在静态存储区，全局变量和 static 变量，PS：如果 static 局部变量在函数内定义，他的生存期为整个源程序，但其作用域和自动变量相同，只能在定义该变量的函数内使用，退出该函数后，尽管该变量还存在，但是不能使用。

3、默认初始化为 0，

315.【简答题】【C 语言】请问你怎么理解多态，他有什么好处

所谓多态，就是指程序中定义的引用变量所指向的具体类型和通过该引用变量发出的方法调用在编程时并不确定，而是在程序运行时确定，即一个引用变量到底会指向哪个类的实例对象，调用哪个类的实现方法，由程序运行期间才确定，这样不用修改程序源代码就可以让引用变量绑定到各种不同的类实现上，从而导致该引用调用的具体方法随之改变，即不修改程序代码就可以改变程序运行时所绑定的具体代码，让程序可以选择多个运行状态，这就是多态性。

316.【简答题】【C 语言】手写代码：求两个数的最大公约数

```
int division(int n,int m)
{
if(n<m)
division 考(m,n); //交换 m 与 n

else if(m==0)
return n;
else
{
int temp=n;
n=m;
m=temp%n;
division(n,m); //重复上述过程
}
}
```

317.【简答题】【C 语言】手写代码：将字符串转 int 类型，要求不能用已有的方法

```
public static int stringToInt(String str) {
int result=0;
char[] ch=str.toCharArray();
int len=ch.length;
for(int i=0;i<len;i++) {
result+=(((int)ch[i]-'0')*Math.pow(10, len-1-i));
}
return result;
}
```

318.【简答题】【C 语言】手写代码：求 x 的 n 次方

```
double Pow(double x, int n)
{
double result = 1;
while (n)
{
if (n & 1)              //  等价于  if (n % 2 != 0)


1
result *= x;
n >>= 1;                //右移一位相当于 n/2(类比十进制来理解)
x *= x;
}
return result;
}
```

319.【简答题】【python】手写代码：比较两个 json 数据是否相等

```
for src_list, dst_list in zip(sorted(dict1), sorted(dict2)):
if str(dict1[src_list]) != str(dict2[dst_list]):
print(src_list,dict1[src_list],dst_list,dict2[dst_list])
```

320.【简答题】【python】请问怎么拿到 python 的输入方式？以及 python 怎么打开文件？

python 读取键盘输入通过 input（）内置函数，读文件则通过 open(filename,mode)

Python 打开文件 f = open('your_file.txt','r')

就可以打开一个文件进行操作。第二个参数为对文件的操作方式，'w'是写文件，已存在的同名文件会被清空，不存在则会创建一个；'r'是读取文件，不存在会报错；'a'是在文件尾部添加内容，不存在会创建文件，存在则直接在尾部进行添加；还有'wb'是写二进制文件；'rb'是读取二进制文件，比如图片之类的。

321.【简答题】【Linux】请你说一下 shell 的基本命令，怎么看到行号?怎么查进程的 id？

获取文本对应文本的行号,可以用 grep，也可以用 sed

grep -n "xxx" a.txt | cut -d ":" -f 1

sed -n -e '/xxx/=' a.txt

shell 获取进程 ID 的方法有三种：

1、ps -A |grep "cmdname"| awk '{print $1}'

2、pidof "cmdname"

3、pgrep "cmdname"

322.【简答题】【Linux】请你回答一下常用到的 shell 指令中与网络相关的有哪些，netstat、ping、ifconfig 这三个的区别，分别是什么功能，netstat 里面一般服务器启动后的端口状态是什么

常用的网络相关的命令有以下几个：

1、ifconfig

这个命令用于显示网络接口，子网掩码

2、host 和 nslookup

这两个命令是 DNS 查找工具，当执行 host 时，会列出某个域名的所有 ip，nslookup 是一个类似于 host 的命令，它用于查询 DNS 相关的细节信息，以及名字解析

3、route

显示路由表

4、traceroute

这个命令显示分组途径的所有网关地址

netstat、ping、ifconfig 这三个的区别：

netstat:显示网络状态，利用 netstat 可以让你得知整个 Linux 系统的网络情况，语法为 netstat [-

acCeFghilMnNoprstuvVwx][-A<网络类型>][--ip]

ping:功能是检测主机，因为执行 ping 命令会使用 icmp 传输协议，发出要求回应的信息，若远端主机的网络功能没有问题，就会回应该信息，因而得知该主机运作正常，语法为：ping [-dfnqrRv][-c<完成次数>][-i<间隔秒数>][-I<网络界面>][-l<前置载入>][-p<范本样式>][-s<数据包大小>][-t<存活数值>][主机名称或 IP 地址

ifconnfig：功能是显示或设置网络设备，其语法为：ifconfig [网络设备][down up -allmulti -arp -promisc][add<地址>][del<地址>][<hw<网络设备类型><硬件地址>][io_addr<I/O 地址>][irq<IRQ 地址>][media<网络媒介类型>][mem_start<内存地址>][metric<数目>][mtu<字节>][netmask<子网掩码>][tunnel<地址>][-broadcast<地址>][-pointopoint<地址>][IP 地址]

服务器启动后一般为 listening 状态

323.【简答题】【Linux】请问 linux 两台机器之间传文件，用的什么端口

Linux 主机之间传输文件的几种方法：

1、scp 传输

scp 传输速度较慢，但使用 ssh 通道保证了传输的安全性。

命令：

将本地文件拷贝到远程：

scp 文件名 –用户名@计算机 IP 或者计算机名称:远程路径

从远程将文件拷回本地：

scp –用户名@计算机 IP 或者计算机名称:文件名 本地路径

2、rsync 差异化传输(支持断点续传,数据同步)

rsync 是 Linux 系统下的文件同步和数据传输工具，它采用“rsync”算法，可以将一个客户机和远程文件服务器之间的文件同步，也可以在本地系统中将数据从一个分区备份到另一个分区上。

如果 rsync 在备份过程中出现了数据传输中断，恢复后可以继续传输不一致的部分。rsync 可以执行完整备份或增量备份。

3、管道传输(降低 IO 开销)

gzip -c sda.img | ssh root@192.168.1.110 "gunzip -c - > /image/sda.img"

#对 sda.img 使用 gzip 压缩,-c 参数表示输出到 stdout,即通过管道传送

#gunzip -c - 中的"-"表示接收从管道传进的 sdtin

4、nc 传输(一种网络的数据流重定向)

nc 所做的就是在两台电脑之间建立 tcp 或 udp 链接,并在两个端口之间传输数据流,是一种网络的数据流重定向。

使用 dd 结合 nc 命令网络克隆磁盘分区：

主机:

dd if=/dev/vda | gzip -c | nc -l 50522

待恢复机:

nc 192.168.215.63 50522 | gzip -dc | dd of=/dev/sda

dd 命令克隆/dev/vda 磁盘,并使用 gzip 压缩,把数据流重定向到本机 50522 端口,待恢复机上使用 nc 连接主机 50522 端口,就能接收主机 50522 端口的比特数据流,然后使用 gzip 解压缩,并恢复到/dev/sda 磁盘。

dd 命令读取的是磁盘扇区,所以不论磁盘文件系统,或者分区表,磁盘 MBR 信息,dd 都能够复制,可以使用 bs,count 参数控制要克隆的大小

5、建立文件服务器

通过建立文件服务器,然后通过网络挂载的方式传输,适用于经常性的拷贝。

324.【简答题】【Linux】请你说一说关于 linux 查看进程

ps 命令：

ps 命令查找与进程相关的 PID 号：

ps a 显示现行终端机下的所有程序，包括其他用户的程序。

ps -A 显示所有程序。

ps c 列出程序时，显示每个程序真正的指令名称，而不包含路径，参数或常驻服务的标示。

ps -e 此参数的效果和指定"A"参数相同。

ps e 列出程序时，显示每个程序所使用的环境变量。

ps f 用 ASCII 字符显示树状结构，表达程序间的相互关系。

ps -H 显示树状结构，表示程序间的相互关系。

ps -N 显示所有的程序，除了执行 ps 指令终端机下的程序之外。

ps s 采用程序信号的格式显示程序状况。

ps S 列出程序时，包括已中断的子程序资料。

ps -t<终端机编号> 指定终端机编号，并列出属于该终端机的程序的状况。

ps u 以用户为主的格式来显示程序状况。

ps x 显示所有程序，不以终端机来区分。

最常用的方法是 ps aux

325.【简答题】【Linux】请你说几个基本 Linux 命令

1、file

作用：file 通过探测文件内容判断文件类型，使用权限是所有用户。

格式：file [options] 文件名

options]主要参数

-v：在标准输出后显示版本信息，并且退出。

-z：探测压缩过的文件类型。

-L：允许符合连接。

2、mkdir

作用：mkdir 命令的作用是建立名称为 dirname 的子目录，与 MS DOS 下的 md 命令类似，它的使用权限是所有用户。

格式：mkdir [options] 目录名

[options]主要参数

－m, －－mode=模式：设定权限<模式>;，与 chmod 类似。

－p, －－parents：需要时创建上层目录；如果目录早已存在，则不当作错误。

－v, －－verbose：每次创建新目录都显示信息。

－－version：显示版本信息后离开。

3、grep

作用：grep 命令可以指定文件中搜索特定的内容，并将含有这些内容的行标准输出。grep 全称是 Global Regular Expression Print，表示全局正则表达式版本，它的使用权限是所有用户。

格式：grep [options]

[options]主要参数：

－c：只输出匹配行的计数。

－i：不区分大小写（只适用于单字符）。

－h：查询多文件时不显示文件名。

－l：查询多文件时只输出包含匹配字符的文件名

－n：显示匹配行及行号。

－s：不显示不存在或无匹配文本的错误信息。

－v：显示不包含匹配文本的所有行。

4、find

作用：find 命令的作用是在目录中搜索文件，它的使用权限是所有用户。

格式：find [path][options][expression]path 指定目录路径，系统从这里开始沿着目录树向下查找文件。它是一个路径列表，相互用空格分离，如果不写 path，那么默认为当前目录。

主要参数：

[options]参数：

－depth：使用深度级别的查找过程方式，在某层指定目录中优先查找文件内容。

－maxdepth levels：表示至多查找到开始目录的第 level 层子目录。level 是一个非负数，如果 level 是 0 的话表示仅在当前目录中查找。

－mindepth levels：表示至少查找到开始目录的第 level 层子目录。

－mount：不在其它文件系统（如 Msdos、Vfat 等）的目录和文件中查找。

－version：打印版本。

326.【简答题】【Linux】请你说一下 vector 的特性

vector 特点是： 其容量在需要时可以自动分配，可以在运行时高效地添加元素，本质上是数组形式的存储方

式。即在索引可以在常数时间内完成。缺点是在插入或者删除一项时，需要线性时间。但是在尾部插入或者删除，是常数时间的。

327.【简答题】【Linux】查看端口号、进程的指令是？动态查看日志的指令？怎么判断一个端口存不存在，磁盘满了怎么处理，删除一个目录下的 txt 文件，你还熟悉其他什么 linux 指令？

查看端口号的两种指令:

netstat –tunlp|grep 端口号

lsof -i:端口号

查询进程的指令

ps -ef |grep 进程

ps:将某个进程显示出来

-A 显示所有程序。

-e 此参数的效果和指定"A"参数相同。

-f 显示 UID,PPIP,C 与 STIME 栏位。

动态查看日志

1、先切换到：cd usr/local/tomcat5/logs

2、tail -f catalina.out

3、这样运行时就可以实时查看运行日志了

怎么判断一个端口存不存在：

netstat -anp |grep 端口号，在输出结果中看监控状态为 LISTEN 表示已经被占用，最后一列显示被服务 mysqld 占用，查看具体端口号，只要有如图这一行就表示被占用了。

磁盘满了怎么处理

1. df -h 查看是哪个挂在目录满了，常常是根目录/占满

2. 快速定位一下应用日志大小情况，比如 tomcat 日志，应用系统自己的日志等。

3. 如果能直观地看到日志文件过大，则酌情进行删除。有时候删除日志文件之后再 df -h 查看空间依然被占满，继续排查。

lsof file_name 查看文件占用进程情况，如果删除的日志正在被某个进程占用，则必须重启或者 kill 掉进程。

4. 如果不能直观地排除出是某个日志多大的原因，就需要看一下指定目录下的文件和子目录大小情况，使用 du 命令。

删除一个目录下的 txt 文件

find . -name "*.txt" | xargs rm -rf

我还熟悉文本编辑指令。

328.【简答题】【Linux】请你说一下 vi 里面怎么替换字符串

vi/vim 中可以使用 ：s 命令来替换字符串。该命令有很多种不同细节使用方法，可以实现复杂的功能，记录几种在此，方便以后查询。

：s/vivian/sky/ 替换当前行第一个 vivian 为 sky

：s/vivian/sky/g 替换当前行所有 vivian 为 sky

：n，$s/vivian/sky/ 替换第 n 行开始到最后一行中每一行的第一个 vivian 为 sky

：n，$s/vivian/sky/g 替换第 n 行开始到最后一行中每一行所有 vivian 为 sky

n 为数字，若 n 为 .，表示从当前行开始到最后一行

：%s/vivian/sky/（等同于 ：g/vivian/s//sky/） 替换每一行的第一个 vivian 为 sky

：%s/vivian/sky/g（等同于 ：g/vivian/s//sky/g） 替换每一行中所有 vivian 为 sky

329.【简答题】【Linux】请问 contrab,uptime,du,netstat 这几个指令有什么作用，如何查看磁盘分区状态

Crontab:被用来提交和管理用户的需要周期性执行的任务，当安装完成操作系统后，默认会安装此服务工具，并且会自动启动 crond 进程，crond 进程每分钟会定期检查是否有要执行的任务，如果有要执行的任务，则自动执行该任务。

Uptime：查询服务器已经运行多久

Du:查看文件和目录磁盘使用的空间情况

Netstat: 显示网络状态，利用 netstat 可以让你得知整个 Linux 系统的网络情况

使用 df 命令可以查看磁盘的适用情况以及文件系统被挂载的位置

330.【简答题】【Linux】请问如何将文本中的 T 全部替换成 t,将其中的一行复制新的一行出来

:%s/T/t/g

331.【简答题】【语言区别】请你回答一下 c++和 java 的区别

1、C++创建对象后需要在使用结束后调用 delete 方法将其销毁，Java 有垃圾回收机制，用来监视 new 出来的所有对象，辨别不会再被引用的对象，然后释放内存空间

2、当变量作为类的成员使用时，Java 才确保给定默认值，以确保那些基本类型的成员变量得到初始化，但是 C++没有此功能，

3、c++引入了操作符重载机制，Java 不支持

4、Java 中没有 sizeof()，在 C++中 sizeof()操作符能够告诉我们为数据项分配的字节数，因为 C++中不同的数据

类型在不同的机器上可能有不同的大小，但是在 Java 中所有的数据类型在所有机器中大小都是相同的。

5、Java 的运行速度比 C++慢，因为 Java 是半解释和半编译的。

6、C++中有多继承，Java 中只有单一继承，但是 java 可以通过接口实现多继承

7、在 C++中，数组定义时，已经分配存储空间，并且可以使用，在 Java 中，数组定义时只定义了数组变量，数组是不可以使用的，只有数组 new 之后才会创建数组，并分配存储空间。

8、C++有指针，Java 没有。