

Work Simulation:

1. 改bug，PM说不test，问你该怎么做，几个选项选efficiency。然后你的同事也要改bug，他想直接push，问你该怎么做
2. 你的代码收到意见说结构不好，要重新写，问你该怎么做
3. 网页有bug，让你找bug，问你需要知道哪些数据来判断bug，然后给log和代码找bug：array 长度不同
4. 规划路径送货，两个算法问你怎么选。然后跟你说可以提供些数据，问你哪两个数据是最需要的用来判断算法，最后选一个算法
5. 要改两个feature：1.添加更多注册渠道 2.修复取消订单容易crash的bug，问你哪个更重要。我选了2（不是说顾客至上么呜呜）之后有个女人说最好先提升feature1，然后问你这时候你该怎么办。最后还是决定提升feature1了。。

最后一道从Log找BUG的题，不是德国地址的题，答案是

 以下内容需要积分高于 150 您已经可以浏览

用户REVIEW的数目和LIST里的数目不相符合

看完小土刀做的题，但是Reasoning 24道题还是没有答完，最后一道没时间看了

小土刀链接：<https://wdx tub.com/interview/14520850399861.html>

第一部分的题都是字母的各种变化，一定要准备一张字母数字的变换表，这样方便很多。
应用题

 以下内容需要积分高于 150 您已经可以浏览

有环保公司， 录用问题, 还有一道是圆桌问题

每个问题要看清楚邮件的描述，当前任务的要求是什么，比如说有一个里面说只有三天的时间就没有注意到。。。之后就时刻把时间记在脑子里。

感觉并不是根据不同的选项会有不同的后续，而是不管选什么都会出现正确选项的后续。所以楼主选完某选项以后出现一个其他选项的场景时，就知道自己选错了。。。。

和人/多人沟通是不错的选项。比如说schedule一个involve所有人的meeting，或者上报给senior engineer, manager, VP等等，不过工作中其他人真的这么有时间吗？

逻辑题有一个是石油公司选人，family income要求小于某值，楼主想当然认为是大于某值，选错了。。。

work simulation要重视，还是很有区分度的。

指导原则，之前有人总结的，我把第二、三条改了一下：

- 1) deadline 是最重要的（说三遍），有用户需求的时候requirement更重要。
- 2) 多跟manager/VP/senior/other team/colleague交流，不能自己暗搓搓做。
- 3) 自己要主动多帮忙。

补充内容 (2019-10-19 10:58):

今天收到email说过了，据我观察通知都是一波一波发的，每月两次这样。oa2 copy random node忘了检查random pointer不为空，但是tests都pass了。oa3选两个算法最后选了要再研究因为没看到deadline requirement。

补充内容 (2019-10-19 11:01):

oa3 code review第一个选和那个人私下沟通，一起开会放到第三。选feature别人不同意怎么办把report 大头(VP /director)放到了第三尽管下一题就是大佬开始说啥啥啥。逻辑题至少错了一道。这些是oa3拿不准的。

一、Work Simulation

[四个小时，5个Module，每个Module平均2至3至4题]

答题形式：

1.（区别效能）对于某个问题，某个情况，你有五种不同的态度。（最有效，很有效，正常有效，一般有效，没什么效果）。将5-7个句子分别放入你认为对应的不同等级的框框里。（大部分题的一个框可以放多个，开头介绍会说哪种框可以放多个，那种只能放一个）

2.（选择最佳）从多个选择中，选择你认为最好的。多选一

3.（睡一觉起来忘记了还有哪种）反正就是多选几。

具体如下：

两个Delivery Algorithm 选一个，根据提供的100天数据统计表。（区别效能：其中我把多要点数据放在了首位；选择最佳）

两个feature选一（1.选feature1的4个优点 2.选feature2的2个优点 3.选优先哪个feature。按照地里面经我选了1，然后一个女的说看起来你像是个新手，我们应该选2问你怎么办（区别效能，如何沟通）4.最终大boss还是选feature1）所以其实第三问你选哪个后面紧接着都是说选另一个我猜。但是最终的最终还是说选1

队友有个代码没有test想直接publish。我们作为责任心极强的准亚麻人。那必须不能让他得逞呀！好说歹说他还是怏怏咪上了。结果就bug了。咨询mannager, senior, 组织开会。全给他整上。最后一题我选了设计一个自动化程序阻止没有test就上传代码的行为！

三个同事（ABC）AB说good job。C说 你用错design pattern类似的。怎么沟通。大致几题（区别效能）我选的是【开会—开会—交流】

一些顾客无法查看网站上某些商品。（1.问你觉得什么原因可能造成（区别效能）2.看俩log分析最终原因是啥（选择最佳）就是代码不work的地方是json最后俩object的内容长度不一致。就是用户review和打分数不匹配）其他面经也都说了。

二、Logic

[35分钟，24道题]

喜欢中文看小土刀

喜欢英文看：这个 <https://aonocode.com/Amazon-online-assessment>

1.前十道左右为字母找规律。我遇到的基本都是+1 or -1 or 间隔相同 or 倒着看。非常简单。

2.中间几道，有关方向（1. Lily家在哪，注意from是相对于家还是Lily，和小土刀刚好相反。2.一个人骑自行车向南一段距离，左转一段距离，左转一段距离，右转一段距离，问距离起始点位置。）也非常简单。

3.接着几个阅读理解（GRE）类似的（1.AB俩网站都可以买东西，A可以买卖二手。B只能买商品可能（不限于二手）。A还可以拍卖，拍卖设置一个底价，价高者获得，但出价最高者不买要受到法律制裁。5选1）这类题慢慢读一遍题再对一下答案就可以。也或者先看一个答案，再读一段原文就能找到。原则就是！！！！不要慌啊，慢慢读。别看到字多就慌了。其实就像是问你((((0+(1+1)+0))))是多少！

4.最后应用题我是【八产品】（四道）follow up和小土刀一样。然后一道小土刀没有的；

【选择最佳的人选】。选的好像是学校运动员（candidate：1.年龄在16-18，有学校认证 2.参加过10场以上校级比赛 3.赢了至少5场校级比赛 4有学校教练推荐信 5.赢过至少一场国际比赛

exception：没满足3报告director；没满足1报告chair）我的答案：第一个学生报告director因为他其他都满足就是只赢了4场校级好像是。第二个学生好像是不符合要求。因为没有推荐信因为他犯规了还是啥的。

Work simulation(原则有先后顺序)

目前两大做题中最重要的原则:

1.requirement排在第一, deadline第二。

2.有manager出现的选项无脑选manager, manager就是一个组的地头蛇。

Amazon9条主要原则

原则1: 客户是上帝, requirement优先, 任何影响上帝的事情都不能干,

如某个requirement影响了上帝的体验,

你就是死键盘上也不能砍了, 宁愿miss deadline

原则2: 为长远考虑, 即客户几年之后可能会出现的需求也要考虑到,

不会为了交付短期的deadline,

而牺牲长期的价值。(比如 global api 和 local api)

原则3: 最高标准, “最高”对应上面的“长远”。

原则4: 一般情况, 能请示manager就请示manager, manager一般不会出错

原则5: 速度很重要, 决策和行动都可以改变, 因此不需要进行过于广泛的推敲

, 但提倡在深思熟虑下进行冒险。

原则6: 不需要一定要坚持“非我发明”, 需求帮助也是可以的, 四处寻找创意

, 并且接受长期被误导的可能

原则7: 敢于承担责任, 任劳任怨, 比如领导说谁会java, 你会你就跳出来说我

原则8: 对问题刨根问底, 探究细节

原则9: 服从大局 (团队比个人重要)

打分不是关键, 排序才是关键。

大部分情况下其实并没有deadline 和 requirement谁更好, 更多还是

这个组合中你对ddl 和 requirement整体的权衡。

每个选项可以评1~5分, most effective 是5, 然后1是least effective

刚开始让你看一些介绍amazon工作环境的视频

1.上来给一段video, 场景是项目的晨会, 就是把team正在推进的项目描述一下,

期间会有多个项目和你有关系, 后面会遇到

2.进入工作界面, 可以看到接受到邮件, 接收到的instant message

3.进入工作状态。会有同事给你发邮件, 发信息。需要你对他们提出的问题做一些判断, 也就是给解决问题的选项评分

4.一个21题, 有log分析bug, 有给报告出问题结论, 有判断项目 走向的

情境1: 给图书馆写图书推荐系统, 关于book api

两个人, 在表达不同的观点

选择: tell me more

一开始其实每个人都在强调自己是对的, 即使有一个人更对一些,

也应该选tell me more (原则8), 选了之后会得到更多信息

情境2: 选图书馆的服务器有没有开放关于实体书的api
两个小哥讨论图书推荐的api应该是自己做还是用现成的。
自己做api覆盖面广, 但是due赶不上, 别人做的能赶上due。
requirement优先 (原则2), tell me more层层递进

情境3: 经理说咱们最近服务器老挂, 什么情况?
先选看internal bug的记录
选 I think service 3 is the problem,
but I would like to see another report to confirm
烙印, 义正言辞说自己做了20年服务器, 不可能有错误,
刚刚调试过服务器, 不可能是内部错误。
选自己去查, 问题的关键在于不要麻烦别人
增加开发过程中测试的时间/测试覆盖更多case, 放5
写Manuel test, 放3
还有个是unit test, 也放3
增加QA的人手, 放1
让客户来当小白鼠发现问题, 放1

情境4: Amazon recommendation system item,
给你推荐一些你感兴趣的item, 第一个issue总是失败,
第二个issue总是显示germany
第一个问题是因为username 太长所以一直报错。
第二个问题是因为他用proxy的name来决定是不是语言了。

情境5: 德国amazon除了什么问题, 让你看log回答问题。问你大概哪里除了问题
亚马逊推荐广告, 给英国人推了德文广告, 给你log文件,
问你可能在哪儿, 找bug in error log

情境6: 员工们讨论case media network服务器最近好多complaints
有德国的, 有invalid recommendation, 有返回404,
找出错原因的相同点
德语因为服务器, 一个因为用户名太长, 一个是有些用户的语言变成德语

情境7: 具体客户ddl 只有两周, 两个方案, 延到四周, 做完整。
另一个说先实现一部分功能做个demo, 再慢慢做。
先做demo放5, 按部就班四周放3, 通知其他组说两走做不完接着做美国放1

情境8: 估计项目开发时间
Manager放5, 找有经验的人请教4, 上网查资料或是先做一段时间再估计都放3,
还有其他裸上的就1。

情境9：一个项目时间表设计

说你是这里最会用什么语言的，比如java

情境10：安排会议

视频会议 5 三个老二开会和老二去找老大开会 3 推迟会议和邮件开会 1

情境11：搞个数据库

两周时间可以搞个数据，**ben可以帮忙，大腿priya可以帮，但是要等一周半
报告manager放5，和**合作等大腿放4，合作/等大腿是3
自己单干，cut feaure都是1

情境12：系统是否升级

做两个feature，一个让100%用户爽，一个让20%用户爽，
但要升级系统，升级系统自己组会爽，但是升级会推迟做的feature，
不升级吧，升级之后还得做一遍
这题的中心是不升级，先做feature，先让用户爽。
先做100的feature再升级，再做20的feature，放5
不升级，因为我们承诺要做feature，放4。
不升级，要搞定feature，可以以后推了其他ddl再升级，放3
不升级，因为对其他组没影响，我们应该focus在request上面，放2
升级，推迟这两个feature的ddl，因为升级造福子孙后代，放2
升级，不然要做两次，放1
这题的关键在于升不升级，要坚定的站在一边

情境13：新产品设计

给8周时间，选择题，让你pick up 一个features的组合要求利益最大化，
每个feature都有相应的价值， $H \gg M \gg L$ 都代表远大于
首先ddl是前提，中位数不能超过8太多，那样的话就算feature再多也没意义，
同价值，按照ddl排序，同ddl按照价值排序。

情境17：代码分析

三段一长选最长

Module1:

制定deliver route plan 需要满足1.enable 80 deliveries per day 2.should not exceed 200 miles

有两个备选方案，第一个每天够了80但是mile数超了好多，第二个mile数没超但送货数量不够

Q1 问你选哪个 五个statement按effective级别排序

Q2 问你如果可以request additional info 哪个更重要 （我选了BC）

A 会影响efficiency的future trends data B mile数和delivery数哪个重要 C 过去100天的data D 城市地图 E 过去3天的performance data

Q3 给了你过去3天和100天的traffic data 问你planA和B哪个好

我真的不知道哪个好啊!!! 最后犹豫半天选了第一个 因为送货数量pattern比较符合traffic，也不知道对不对

-

Module2:

同组的两个人说你的code不错，一个人说不行要重做

Q1问你怎么办

选项大致就是 A 有人approve了所以就按我的来 B 你说不对就按你的来吧 C meeting决定

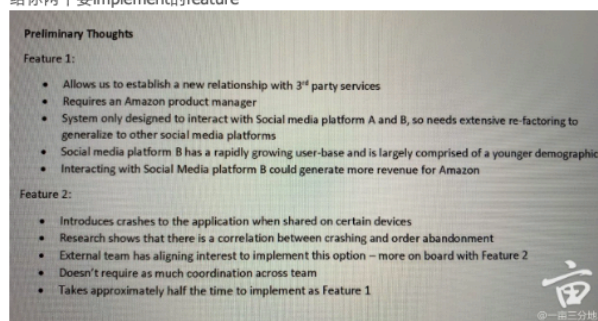
应该是选和这个人单独offline meeting那个选项 因为接下来就和他讨论了

Q2 讨论到了下班，你还是很confused, 但记了好多笔记 问你怎么办 排序

A 明天再full time meeting B 新方法太难了还是用旧的 C 回家过一遍笔记把能写的都写出来 不能写的记下来明天接着meeting D do as much as you can E找那两个说你做的不错的同事帮你 F 和这人再meet三十分钟（这是要把他累死啊）

Module3:

给你两个要implement的feature



Preliminary Thoughts

Feature 1:

- Allows us to establish a new relationship with 3rd party services
- Requires an Amazon product manager
- System only designed to interact with Social media platform A and B, so needs extensive re-factoring to generalize to other social media platforms
- Social media platform B has a rapidly growing user-base and is largely comprised of a younger demographic
- Interacting with Social Media platform B could generate more revenue for Amazon

Feature 2:

- Introduces crashes to the application when shared on certain devices
- Research shows that there is a correlation between crashing and order abandonment
- External team has aligning interest to implement this option – more on board with Feature 2
- Doesn't require as much coordination across team
- Takes approximately half the time to implement as Feature 1

© 一亩三分地

Q1 选出4个feature1的优点

Q2 选出2个feature2的优点

两个都不难 比gre阅读简单

Q3只能implement一个 你选哪个 应该是排序 我选了feature1 应该是对的 因为最后决定做1了

Q4 有个senior engineer说我们要implement feature 2 问这时候你怎么做

A 和他还有high level decision makers开会 B 坚持做feature1因为已经讨论过pros and cons了 C 和这人还有external team开会 D 全组投票 E 因为他比你更有经验所以听他的 F 问manager

Q5 manager过来说我们要做feature1, 而且要在social media C D E上实现, 但是ddl很紧张, 问你怎么做 排序

A 直接跟manager说做不了 需要更多resource B 因为ddl紧张所以不test了 C 排序这三个social media的重要性 先做重要的 把不重要的留到ddl之后 D 先做着 过几周再找manager说你做不完了 需要additional resource (这个不太好吧) E 自己找team member寻求帮助 F 直接驳回 建议只work on 1-2 social medias G 自己work overtime

Module4:

亚麻的product page出问题了，有customer无法查看product info，你manager让你解决

Q1 What additional info would be helpful? 排序

A customer region B URL C Screenshot of error page D session ID E Browser versions

Q2 你怎么fix this bug? 排序

A search the logs B look at metrics(latency, error counts, # of requests....) C 手动去看code然后debug (好猛) D 分析有可能会fail的request的log E 跟technician说这是偶然现象 自己会fix F 只分析failed request不看valid request G 因为太urgent了所以立刻去向大佬请教

Q3 给了log data 让你分析到底哪出错了

这题应该选unequal number of rate and review

Module5: 这个module好好笑

你要debug一个东西，manager跟你说大概需要三天，但PM让你今天就交因为他觉得是个小bug 可以不用test

Q1 你怎么办 排序

A 跟senior eng讨论一下risk再决定 B 还是先test过后再release C 跟PM manager开会再决定 D 都有理所以重新问问你manager E 相信PM 不test了

Q2 一周以后你同事也遇到了相同情况，他想push without test，你怎么办 排序

这道题选项记不太清了 大致就以下几种： A 把你上次情况跟他说，劝他别这样 B 不关我事 C 建议他找senior eng讨论下 D 找他manager告状（当时看到这个选项笑得停不下来）

Q3 马上下班时候收到同事msg，他没听你的，直接push code了，结果翻车了（哈哈哈哈哈哈哈哈哈哈）你怎么办 排序

A 早跟你说了你不听 B ask for more info about the issue C 帮助他revert code change D 帮助他push另一个code把问题立刻解决（这真的能做到吗。。） E 找senior eng求救 F ignore the message and sign off (这个也好好笑)

Q4 manager给你发邮件说让你come up with a plan to make sure this doesn't happen again 而且希望你把这个任务prioritize 你怎么搞 单选

A 跟他说我做完手头的再去做（作死啊） B write up document详细描述 C set up automations来预防 D 规定以后每个code change都必须有至少一个peer review E 和manager, PM再好好探讨下不test的风险（打PM脸啊这是） F 让之前翻车的同事跟你一起弄

Q1: Schedule the design review meeting (1)

- 1 - We can take our best guess at an estimate on our own
- 2 - We should work for a couple of days to gauge our progress, and then complete our estimate from there
- 4 - We should consult a coworker who has more relevant experience on this type of task
- 3 - We should conduct our own investigation utilizing online research materials and internal documentation
- 5 - Let's ask our manager how we should go about developing an estimate

Q2: Schedule the design review meeting (2)

- 3 - Ask all parties to identify a back-up person who could meet during a designated time
- 3 - See if there is a backup person on the Localization Team that can meet
- 5 - Set-up videoconferencing to include all POC's regardless of their physical location
- 1 - Agree to postpone the design review for two weeks when all parties have more availability
- 2 - Discuss the design review over email
- 4 - Agree to schedule the meeting at Xavier's location an hour away

Q3: Response to Ravi (1)

- 3 - We should miss the conference and increase the timeline to four weeks because we have four weeks of work
- 4 - Take a day to investigate whether adding additional resources would allow us to meet the original timeline, and re-evaluate afterwards
- 1 - Tell the Localization team if can't be done in the timeline, so we should go ahead with the US launch and delay the global launch even though it means adding an additional week of effort to the four week estimate
- 5 - Take two weeks to create a prototype of the feature to demo at the conference, then take the additional two weeks needed to fully complete the feature
- 2 - We can still hit the two week deadline without any changes by working harder and putting in overtime

Q4: Response to Ravi (2)

Begin your investigation using the old error logs, but tell Ravi he will need to run the new logs if the old logs aren't useful

Q5: Response to Aaron and Jacob (1)

Can you tell me more about what you're talking about?

Q6: Response to Aaron and Jacob (2)

You said we have an internal database of both digital and physical books. How did we get the physical book data if the Book Database API doesn't give it to us?

Q7: Response to Aaron and Jacob (3)

I recommend you go with Jacob's solution. We should miss the deadline to build our own service and meet all the requirements.

Q8: Roadmap

Since you know more about the programming language than anyone else, you revise the estimate for porting to Java.

Q9: Response to Nadia

What were the internal test case results?

Q10: Most likely cause of German language issue

Site is using proxy server location to determine displayed language

Q11: Most likely cause of invalid recommendation issue

Database field storing username is too short

Q12: Log trace investigation success

- 5 - Increase time allotted for testing in overall lifecycle
- 5 - Update automated end-to-end tests to include broader data coverage
- 3 - Write more unit tests to include edge cases
- 3 - Have team members perform more manual testing before checking code in
- 1 - Increase the size of QA team
- 1 - Have more user testing in beta phase

Q13: Response for meeting the deadline

- 2 - Work on the project on your own, putting in extra effort to finish on time
- 3 - Work on the project on your own until Priya is available, then continue to work on it together
- 4 - Work on the project with Ben, being sure to watch his work closely because of his lack of experience
- 5 - Tell your manager you will not be able to complete the project in the time available
- 1 - Cut features from the product so you will be able to meet the two week deadline
- 3 - Start working on the project right away with Ben. Then ask Priya to contribute what she can when she is available

Q14: Response for completing this work on time

- 4 - Work with the Customer Incentives Team to identify the critical features that they need by the deadline, and focus on those
- 2 - Push the timeline back another week to ensure there is enough time for all work to be completed accurately
- 3 - Ask your whole team for help, explaining the urgency that another team is blocked
- 5 - Ask your manager for help in determining the best approach to meet the new deadline
- 1 - Put in extra hours yourself to make sure everything gets done on time

Q15: Upgrade

- 4 - We should not perform this upgrade at this point in time. We promised the Retail Website Team we would have their new features complete by the proposed deadline. Let's postpone the upgrade to another time
- 2 - We should not perform the upgrade because it will not have a significant impact on the Retail Website Team's experience. We should focus on the Retail Website Team's requests
- 3 - We should not perform this upgrade at this point in time. Our top focus is meeting our agreed-upon commitment with the Retail Website Team, so we should finish that first. We can focus on the upgrade afterwards by pushing our deadlines for some of our other projects
- 1 - I think we should perform the upgrade. The right thing to do is push back on the Retail Website Team because it will keep our team from having to do the same work twice
- 5 - I think we should perform the upgrade. As a compromise, we can include the gift recommendation feature the Retail Website Team wants by the deadline and then complete the upgrade. We can finish the seasonal-based gift recommendations feature after the deadline
- 2 - I think we should perform the upgrade. The right thing to do is push back on the Retail Website Team because it will allow us to more efficiently serve the customer and the customer will be helped in the long run.

Q16: New product design

- 2 - A, C, D, G
- 1 - A, C, D, G, H
- 4 - A, B, D
- 3 - A, C, F
- 5 - A, D, F
- 3 - F, G

Q17: ?

?

Q18: Problem with Product.wasPurchasedByUser()

It has performance issue

Q19: Most effective way of improving ShoppingCart()

Change the design of ShoppingCart by removing ShoppingCart user and making shopping cart a property of User instead

Q20: Five tests within ShoppingCartTest()

Fail - Test1
Pass - Test2
Fail - Test3
Pass - Test4
Fail - Test5

Q21: Ask Jacob a question

- 3 - Do any other projects depend on fixing this problem?
- 5 - How many customers is this affecting?
- 5 - How does this affect customers?
- 4 - Are we receiving complaints from customers?
- 2 - How long will it take to solve this problem?
- 1 - If I help you with this problem, will you help me finish my work today?

OA2 - Work Simulation

时间非常充足，可以慢慢做，就是问你要不是你你会怎么选，你同意谁的观点和给一下几个做法打分的题

各个员工讨论 **case media network** 服务器最近好多 **complaints**,有德国的，有 **invalid recommendation** 的，给了个列表好多国家的服务器返回什么 **404/german recommendation/ invalid recom/**问是什么原因。还有俩个年轻老白讨论客人要强烈要求有硬皮书的推荐，但服务器里只有 **digital** 版本的，到底要不要加这个功能，感觉后面的视频是根据你的选择来的（有待考证）；里面有个会议室白人，亚裔，烙印在讨论服务器最近好多 **complaints**,然后我选则的要看 **Intenal test**，结果后面会议结束烙印站起来义正言辞跟我说，我已经写了 **20** 年服务器了，不可能有错误的，而且我刚刚才调试过机器，绝对不可能是内部错误。呵呵，里面有个选项问，烙印 **is not helpful...**只能呵呵~~ 大部分跟地里说的一样，类似问卷调查，选 **deadline** 更重要 和用户体验更重要。

第一个情境是给图书馆写图书推荐系统，第一问让两个人继续说，第二问选图书馆的服务器有没有开放关于实体书的 **api**

后面有会议说系统出现 **bug**，该做出什么反应，选看 **internal bug** 记录。

最后是五个 **case** 看哪个可以通过，前人都提示过，注意 **user** 的构造函数没有给 **email** 赋值。

simulation 就是看 **email**，**chat...**大家记得每收到 **email** 就要看看，我当时碰到没有题的 **email** 直接跳过，后来做题的时候做了几道发现信息很少做不出来随便乱选了，翻了翻记录才发现有些信息都在那些没题的 **email** 里了。。看 **log** 得题就找相同错误的规律，我记得有道我选了地点都在德国，有个是因为 **username** 太长没存全， **testcase** 就是地里说的那些 **email** 没有初始化，

找错题有 **5** 个 **unit test** 有一个是 **user** 的 **payment method** 返回的是 **null**，一个是 **user** 的构造函数不包含 **email**， 一个是 **setPrice()**传进去的参数是 **double**，但是 **return** 是 **int**。**coding:** 1 reverse right half linkedlist example:
2->1->3->4->5->6->7->8 变成 **2->1->3->4->8->7->6->5** ； 如果总是为奇数，中间的也要变 **5->7->8->6->3->4->2** 变成 **5->7->8->2->4->3->6** 很简单就不多说了

Work Simulation 一开始两个码农撕逼，一个要用 old API 可以满足 deadline，一个要独自开发 new API 可以满足 requirements，这道题连续让你选三次，每次的视频都是根据你的选择不同而不同的。。楼主纠结很久后选择站在那个颜值更高的码农一边。。满足 requirements。。那仨题其实是一个小测试：第一个选 deadline，因为这时没提出用户。后两个全用户优先。。。。写这里给后人参考下。

其他不这么二选一的绝境，只要坚持 deadline 最好不要拖，自己辛苦一点无所谓，多咨询 manager，找其他有经验的人合作啥的，随机应变吧。。

会有让你安排一个项目的计划，因为有很多不同的 feature 可以实现，但是要在 8 个月之内搞定，每个 feature 会有一个预计的占用时间和这个 feature 的重要程度。。只要坚持在占用时间一样的情况，多选牛逼的 feature。。

Log 里德语我选的 proxy, invalid recommendation 是因为 username 太长，database 的那个 field 定义长度短了。。

ShoppingCartClass 两道题三短一长选最长，之前这么选的拿到 video 了。。

5 个 Testcase 选 1， 3， 5 过不了，2， 4 能过。。

显示德语是因为 proxy 推荐错误因为 username 太长的被简化了。时间很充裕完全不用着急

-
1. deadline 与 requirement。看着选吧。
 2. log 问题。找相同原因就行。我看的 log 是某个 service 出问题了，给了你一个 report。第一问是为什么会出现德语，看 report 发现出现德语的共同点是 locate 都在德国，所以答案选的就是 locate。第二问是为什么有的是 invalid，看 report 发现共同点都是 username 都很长，因此选的 username 很长。
 3. test case。关于 shopping 的代码。第一问是某个 method 为什么不行，答案选的 performance issue。这个不太确定（其他几个选项更不合理）。第二问是 how to improve shoppingcart class。我选的是 add user.id to shoppingcart class。第三问就是 5 个 test case 了。地里前辈说过很多了，应该是 1， 3， 5 跑不过。第一个是 getdefaultpayment 会返回 null。第三个是 user 并没有初始化 email，所以 getemail 会出错。第 5 个是 setprice 的 method 返回的是 integer，而 testcase set 的是 double 。