吃过的亏：【学习方面】
1.为什么要一味地赶进度，不检验不巩固自己学到了什么，最后要用的时候跟没学一个样?
2.一定要以新带旧，才能知其然也知其所以然！更容易记住
2.学一节及时巩固一节，反正迟早要记，而晚复习=重新学
3.不要与知识对抗，它忘它的，我们记我们的，多看几遍就熟了
3.学习拉勾的讲义是怎么写的，将所做的打印出来，有利于掌握
4.打印还有多屏的效果，有4~6屏，方便阅读、做笔记

2.记住一个知识点需要的时间几乎是客观的，不以人的意志为转移
3.广度和深度不行，用的时候都不知道从哪儿去查
4.80%的精力应该放在不会的上面，不会的才是人人都困难的，要攻克只能反复消化
5.不看视频，不完任务而自己看书很容易迷失掉，本质上还是没有个计划

7.谁给你的勇气说边学边玩？你不付出努力、光想一想、谁来给你执行?
8.计算机类不要觉得看视频浪费时间，知道的你快速看一下不就行了，不知道的不仅老师给你讲，还给你演示，这是你零基础看书的效率比不了的
9.学过的天然就在忘，只有顺势而为，及时回顾
10.相似易混的必须放在一起对比着求同存异，效率才高
11.只有你能救自己，救朋友

广深 计划 反复（跟着高手学）
费程（费程：1把每门课的内容做成知识结构脑图，只填

写关键词，重在梳理全课程结构；2看着自己做的脑图，把自己想象成为高水准教授，你的脑图就是教案，打开倒计时，你需要在1小时内复述出所有内容；3复述不出来的内容再做记号重点去背，口述输出才是最能校验是否真正掌握，有多少人想得到却说不出来？不是脑子不好使，是没被训练过，7次以上你会形成思维惯性）
多消化 多练 多完善

如果遇到了困难，可以从哪些角度去解决：
1.
what it is
why it exist
how to use
where to use
when to use
2.倒推：你别管他代码有多复杂，你就看他实现了什么效果，然后去反推
3.零秒思考：
4.先生存后发展：先背下来，用起来，再学着自己写
5.①一切皆是数据，一切皆是操作数据
②类、方法都是对数据、功能的封装
6.对于MVC架构，C—S—D层会拆分即可，并非主矛，重点关注代码如何实现
7.循序渐进、书读百遍其义自见、多看几个作者
8.新的可以不学，旧的绝对不能忘
9.变被动为主动，不打无准备之仗
10.学习的本质就是把不会的筛出来并学会，目的就是用所学的主动解决新的问题、积累经验
11.1.被动变主动 2输入变输出 3.提前做好准备
12.掌握了多少才是衡量的标准，一味地赶进度最后跟没学一样

怎么做一个项目：【类关系图，代码时序图】
1要哪些微服务，他们之间的关系
2一个微服务中要哪些类，他们之间的关系
3要用哪些方法执行什么需求