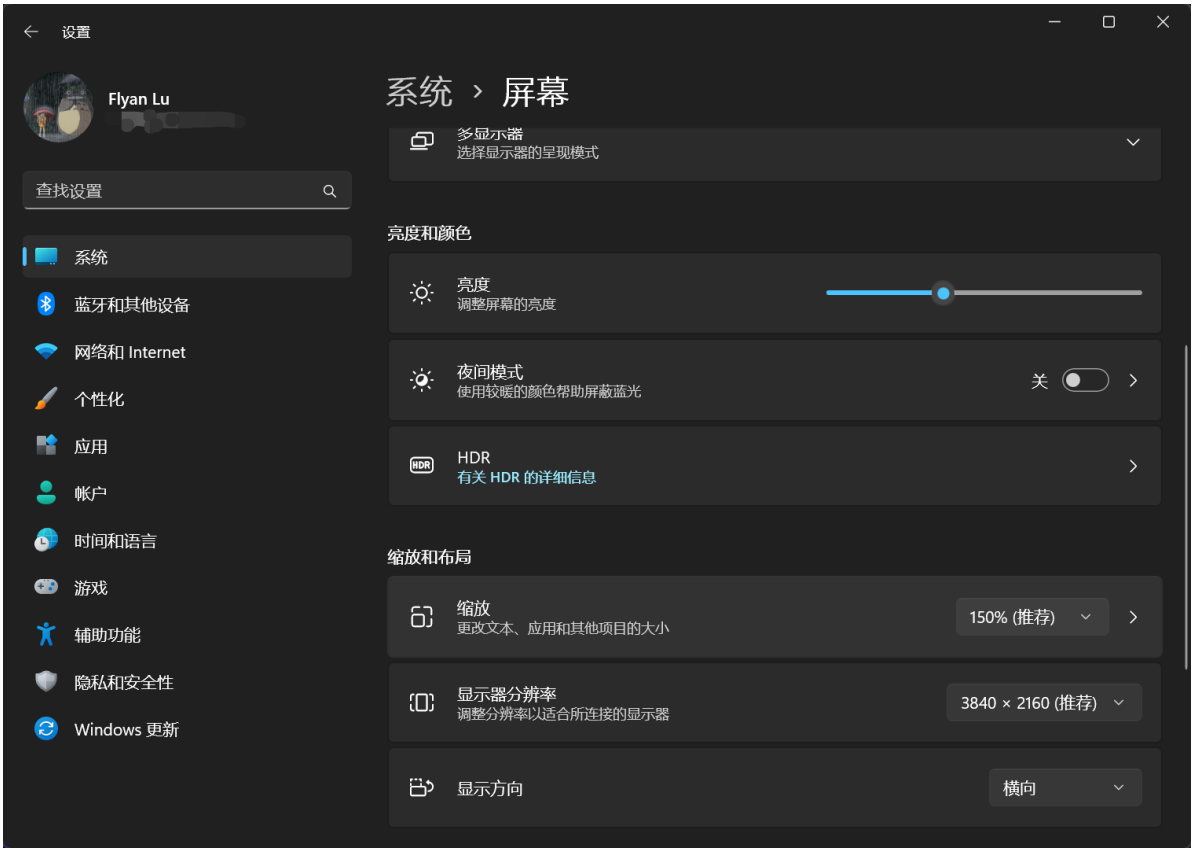
回头看

首先，给同学们说一声抱歉。主要有几个方面。

一，回去以后，我想到，其实外接投影的事后，我没有考虑到外接投影的缩放，所以我在我使用的软件 Obsidian 中，即使把文字的大小调到了最大，在投影那里还是显得比较小。这对于在展示的时候，是一个致命的问题。



二、我没有充分考虑到笔记本供电的情况，以及机房的插座的供电情况。这导致我在讲完课程内的一些个人体会之后，笔记本的电量直接罢工。这在后面又给大家带来了另外一个不好的体验——直接就没有东西看了，只能听我空口无凭地讲。如果我是听众，那么，自己也会感觉，这人谁呀，讲得太差了。对于我，最后则是漏掉了一个重要的部分——答疑。我更希望听到有困惑的同学提出困惑，然后我可以就我目前的短短几年的经验来给大家一点微薄的想法，或者说，启发。

三、我其实没有准备充分。我其实是下午的时候才去准备这个分享。具体的别的原因我也不去讲，归根结底，我确确实实是有没有做好的地方，这是不得不承认的。实际上，我今天对着笔记本上的内容，只是一个大纲性的内容，更多的是临场发挥，这对听众的体验是影响很大的。十分抱歉。所以，我希望以文字的形式，给大家作一些弥补。这毕竟也是我很期待的一个机会，希望可以和类似于过去阶段的“我”去作这样一个对话，如果可以有当面的形式，那我真的可谓是诚惶诚恐。我希望我过去踩过的坑大家可以不必踩，可以把时间花在更多的自己感兴趣的问题上面。当时学院的老师和欧阳老师说，能不能请这样一种普通一点的学院的毕业生给大家作一点分享呢？我当时内心是很欣喜的，“这真的是可以的吗？”，所以，从我个人的角度来看，我非常珍惜这次机会，我也知道，以我的水平给大家能够带来的东西也非常地有限，绩点绩点不高，甚至可以说是很低，八十来分（顺便说一句，我是八十分主义者），技术技术也不过关，至今没有可以拿得出手的像样的作品，即使是目前我开发给自己使用的输入法，其实也只是满足了我个人的日常的使用需求，离真正地贴合更多的用户的需求还有很长的一段路要走。所以我相对来说是比较珍惜这次机会的，希望大家可以谅解。每当到了这种时刻，我总会想起 Vamei，我的 Python 路上的引路人，更准确地讲，是我编程路上的引路人。也正是他带给我的影响，使我可以厚着脸皮来给你们作这样的一次不伦不类的分享，我希望多多少少可以影响到一个两个如我这般曾经迷茫

的人，也多少是对我所受到的 Vamei 和其他 Vamei 这样的人的帮助的一点点小小的交代。还有就是，有人可能注意到了，我自己在分享的时候并没有说太多关于自己的信息，甚至连名字也没有提，其原因也是如上所述，我只是代表了软件学院更多的普通学生中的一员，我不希望给在我这个层次之上的同学们留下太多的记忆，而只是作一些事实上的描述和分享，并希望可以由此给同学们带来一定的启发。

四、我个人的表达能力。我个人也有感觉，我在公众面前总是无法自如地表达。我曾尝试过无数次的改变，现阶段的成果就是现在这个样子，很紧张，所以说话可能会给人一种断断续续的感觉，我自己录音回头听的时候发现这个问题尤其明显。也是因为这个原因，我说话的语速也会受到影响，只有慢下来，我才可以把每一个字给表达清晰。

此外，在我这里没有《浣花洗剑录》里万老夫人那种"伸手不打笑脸人"的不好的说法。如果你们对我表达的内容、对我当前表述的方式和风格有所不满，那么，大胆地表达出来，不一定要表达给我听，可以说给身边的好友听，纾解一下被我侵蚀的情绪，我是举双手赞成的。当然，也是可以说给我听的，期望通过你们的意见，我可以变得更加符合自己的期待。因为我以前在受到批评或者打球遇到挫折的时候，都是这么和身边的几个宿舍这样宣泄的，推己及人，不求谅解。

前言

首先很感谢欧阳老师的邀请，让我能够有机会去作这样的一个分享，类似于这样的分享，从我个人的角度来说，我是非常地乐意去实施的，因为我个人在软件工程这样一个专业经历了大学这几年的学习，其中遇到了不少的困难，踩过了不少的所谓的"坑"，有些解决掉了，有些至今也没有办法解决，但是经过这么长时间的专注，在经验上总归是有进步和积累的，而在这样的情况下，我就很希望能够将我在软件学院获取到的经验能够多多少少地给别人也带来一点帮助，让别人少踩一点我曾经踩过的坑，那么，我的内心会感到我曾经遇到过的种种问题，它带来的价值就得到了进一步的提升。

然后，说明一点，今天分享的主题主要是"技术"这一条路线，这一点可能和实训的关联稍微紧密一点。这里再多说一点关于今天我有这个机会来给大家作分享的因由，那就是前几天我们学院召开了一个关于大三实训的意见座谈会，最后提到了一点，能否在特别优秀的同学之外，抽取一些能够照顾到范围更广的学生的这样一种相对来说更加贴近大部分软件学院的学生的高年级同学来给大家作一点关于过来人的经验分享，所以，我现在作为其中的一员，就出现在了这里。

以下的我尽可能地从细节入手。以及，必须要说明的一点是，本篇文稿可能更多的是和过去的我以及过去的我身边的一些同志对话，所以，对有的同学可能缺乏适配性，关于这一点，请选择性阅读。

个人介绍

软件学院大四的一位普通的学生。

简要介绍一点我个人在大学之前的背景。来自于教育偏落后的地区，苏北乡下。农村教育。没有信息竞赛背景。在大学之前接受的是封闭式教育，没有太多的机会接触电脑。

之前可能大家听过一些启明那边的技术团队的我们软件学院的前辈的分享，或者是保研、出国的同志的经验分享。我今天则是从另一个视角——一个软件学院普通的学生这样一个角度来给大家作一点大学本科阶段纵览性的分享，希望可以给大家带来一定的启发。

毕业去向：就业。关于这一点，涉及到保研、考验和工作的抉择。后面我会给出我个人的一点见解供大家参考。

课程内

关于上课

有些人可能不太喜欢老师的上课，可能会选择去自学这条路线，这当然是很好。我在这里也仅仅是给出我个人的另一条思路。

为什么要上课，我的回答是：答疑。我们学院的老师在答疑这一块我觉得至少从态度上来讲是非常好的。如果把他们和我大学之前的教育环境里的老师相比，那么，我个人的感觉是从盆地来到了天上的云端。在大学之前，我是不敢想象我可以去无拘无束地去询问老师那么多问题的，而且更不可想象的是，我都可以得到相应的回复。我不知道别人的想法，在我这里，我觉得我有太多太多的问题，尤其是关于本专业方面，有太多的我无法弄清楚的问题。

在之前的座谈会里面，王书记（在这里提王书记仅仅是因为是他引出了这个概念）提到翻转课堂的概念，我其实是第一次真正去了解这个概念，

> **翻转课堂**（Flipped Class Model）是学生在正式学习过程中，课前利用教师分发的有关材料（视频、音频、PPT、电子教材等）自主学习课程，再到**课堂**上参与同伴和教师的互动活动（提问、解惑、探究等）的一种教学模式。 与传统教学模式相比，**翻转课堂**更突出了"以学生为中心"的教育理念。

以上是复旦大学给出的概念，可能稍微会专业一点。我觉得，这在某种程度上是解释了答疑解惑在我们学习过程中的必要性（如果你觉得不对，而且你也有合适的方法，那么当然要承认，每个人的思想和学习思路是不一样的，每个人有每个人的对法）。

大一

大一的专业课程主要是有 C 语言。而且我之前是和大一的同学打球时聊到过一个问题，2022 级似乎是没有写 C 语言的课程设计——聪明的小蛇。从我的角度来看，不管是出于什么缘故，把这个内容砍掉都是很可惜的。因为它在某种程度上算是我们第一次正式接触软件开发这一个领域，而且在大一上就可以接触到这样的机会，它对于我们的基础有很大的提升，我们学完一门编程语言之后，其实心里对它的印象还是有很多模糊的部分的。

离散数学

在后面其实可能用到的不是很多，但是它是我们重要的专业基础课。因为没有涉及到很具体的上机实验部分，这里就简要说一点点，比如，我们可以使用一门编程语言去尝试一下里面提到一些问题，比如哈密顿环问题。

数据科学导论

本来这是在大三下选修的一门课程，现在学院的教务把它提到了大一来上，我从中看到的是改变，学院在改变，但是，改变的可能不是很好，这一点我们在毕业午餐会中也是有人提到过这个问题（其实就是我）。我这里谈一下在具体的实践方面，这一门课需要注意的地方。这门课我觉得老师给的实验的指南是非常详尽且具体的，我们只需要按部就班地完成即可，可能其中会有一些环境配置的问题，比如 MacOS 和 Windows 中遇到的路径问题等等。但是总体而言，这门课对于提升我们的 Python 水平很有帮助，同时也不会给我们带来太多的关于实验最终能否顺利自主完成方面的困扰。

大二

大二的专业课是比较多的，从某种程度上讲，它对我们的就业的意义是很大的。

大二上

面向对象和程序设计

其实就是 Java。Java 平时的上机实验，不管怎么样，最后总是可以做出来的。我们需要关注的是它的大作业。它的形式其实和实训的形式类似，给出的任务是让我们使用 Java 来实现一个项目，可以是游戏，也可以是其他，比如坦克大战、Todo 应用等等。关于 Java，我想说的是，它需要我们在平时就花费大量的时间去学习，不管是看网上的视频课程还是去阅读一些经典的书籍，再结合实际去实际地编写程序，去不断地尝试，这对后面能够自主高效完成大作业是很有帮助的。关于 Java 呢，我的记忆里是有这样一个片段，当时身边有几位同志在暑假的时候就抱着《Java 核心技术》在那里拼命地看，我当时对此表达了我的疑惑，他/她也表达了他们的疑惑，你为什么不看，不提前去学？我问为什么？他们说，下个学期，也就是大二上，有一门 Java 课程，如果不提前学，会很吃力的！尤其是它的课程设计！我心下了然。后面其实也还在学期初，在别的不太重要的课程里面看见过班上的同学在课上抱着 Java 书在那里阅读，状态十分投入，似乎是在为即将到来的考试作复习，交流之后，发现也是类似的道理。

还有一点可能需要描述一下，我的印象里，有很多同学完成的课设可能并不是自主独立从零开始构建的。以及，有一些同学完成的内容最后是使用 js 去写的，但是没让老师知道。我觉得这里涉及到一个叫作“技术信心”的问题，所谓的“技术信心”就是，我事先在某一门技术上面没有很深的沉淀和积累，但是我以前有很多成功解决别的问题的经验，所以，遇到新的技术，就会觉得可以完成。这门 Java 课的课设，如果做不好，我觉得可能会对我们的技术信心造成一定的影响。我能给出的参考依旧是，多学、多看、多实践。

数据结构

数据结构这门课程的重要性我不去作过多的赘述。相信大家已经在很多地方提到过好多次了，比如什么 `程序=数据结构+算法` 等等。我在这里多说也只会讨人嫌。那么，继续从实践的角度来看这一门课程。数据结构课程里面的每一个知识点我们其实都可以使用实际的 C 语言或者 C++ 去实现。上机实验要求的也大概就是这个方面的内容，问题的关键在于，我们能否对所学的数据结构和算法有清晰透彻的掌握。我的脑海中有一个回忆，那就是，老师让我们去研究汉诺塔的问题，后面上机的时候，在楼梯口，就有人追着老师问，究竟什么是递归，如何去理解递归？还有的同学会问关于子序列的问题。这里我举出的例子其中的主人公可能是我，但是由于我更想以毕业生的身份回头以类似于一个第三者的视角去看这些问题，所以就不作具体说明，下面也是一样。

当时我就想，我能够提出什么样的问题呢？由此，我又回想起在一门也是不太重要的课程当中，我看见一位同学在阅读《数据结构与算法分析 C 语言描述》，也就是我们当时的教材。我清楚地记得，那时才是学期初，但是他/她的整本书已经看完五分之四了。我问，为什么？回答的大概意思是，为了在课上向老师提出更好的问题，为了把这一门课、这一本书吃得更透，只有自己先实现读一遍，尝试一遍，记录下问题，回头再去上课的时候，才可以理解得更加地好。稍微多提一句，这一位同学的加权是年级的前五名水平，有需要的同学可以结合自身作一点点参考。

我们当时这一门课程的课程设计是实现一个图形化的数独游戏。主要是巩固一下大家对回溯和递归的理解。这个可以使用 Python 里面的 pygame、C++ 的 sfml 或者 sdl、或者 easyx，当然，使用前端来绘制也都是可以的。如果没有前置的经验，那么，这里可能需要注意一下。其实如果使用控制台来实现，那也是可以的，老师更多关注的可能是对算法实际的理解，这就是我们现在回头看可以看到的一些课设的“变通”之处。

汇编原理

汇编原理的课程内容就是 80X86 汇编语言。我们这门课实际上是使用了两门教材，一本是黄色的我们学校出版的教材，另一本是王爽的汇编语言。实际上我们在上机实验中更多使用的是王爽的汇编语言。

前面的实验照着书上依葫芦画瓢即可，可能对一些同学来说，因为此前没有什么基础，会有一点点困难，但是只要肯花时间照着书上实际去看，以及多去网上作 research，那么，问题就不是很大。

稍微有点困难的是最后的课程设计，我们当时是有两个任务，第一个是完成一个学生成绩查询的程序，另一个是完成书上的一个实验，第二个是加分项，如果只完成第一个的话，那么，最后的分数上限好像是只有八十分。

我对这个的实验其实没有什么特别好的办法，做总可以做得出来，但是，我觉得教材没有讲解得到位，也可能是层次对我来说太高。我个人觉得我更适合学王爽的教材。某些老师则会说，哎呀，王爽的教材太简单啦，对你们来说肯定啥问题都没有。我觉得可能还是稍微要照顾一下学习方式和速度和大家稍微有点不同的学生？以及，我觉得老师并没有把那本黄色的汇编讲清楚，但是，其实这个东西并不像算法那样会涉及到很多思考的东西，这是死死的东西。也是因为如此，我不知道对过去的我说些什么，是让自己多花时间去看？还是建议老师讲得更细致一点？不好说。

大二下

操作系统原理

操作系统原理这一块，这一门课可以称得上是艰深。如果我们想要把它吃透，在一个学期内是不可能的事情。

关于这门课，我依然是建议可以多和老师沟通交流，不厌其烦地去询问老师我们不解的问题。从

我这里还是只谈一谈我个人对实践方面的一些想法和我曾经遇到的困难。平时的上机任务是编写一点 shell 或者 batch 脚本，或者就是用 C 语言模拟一下哲学家就餐问题等等。最后的课设是完成一个简易的操作系统。更具体一点是实现《Orange's: 一个操作系统的实现》这本书里面的一部分。这个对我们的 C 语言能力要求是比较高的。所以，我们平时可以注意把 C 语言的知识带着去看，去实践。

我当时有幸帮助老师整理一些他正在编写的教材的实验方面的内容，而老师也就把他那里收到的所有同学的课程设计和实验文档都给我们作为整理的参考。但是看完不同年级的很多同学的文档，我个人的感觉是，实际上能够把这个课设做到可以让我们自己觉得满意的程度的，只有寥寥的几份。这里面的原因当然有很多，其中一个就是，那个时候虽然是大二年级，但是有一些同学要开始奔波于实习了，这也就造成了当时的一个比较尴尬的事情，老师让我挑我觉得可能会做得比较好的同学上交的作业，然后他帮着给我看看要点，我当时脱口而出我觉得是当时我们软件学院在技术这一块可以在整个四个年级排上前三的一位同志，结果，人家并没有把这个课设放在心上，并不是九十五分主义，我和老师当时都打了个哈哈就过去了。实际的情况也确实是当时人家早就已经去往我们学院的大部分学生都很向往的一个公司实习去了。

这里其实就也冒出了两个选择，依旧是在我的角度看来，第一个选择，我们可以选择把课设做得尽善尽美，那么，那就需要我们花费大量的前置时间，这个可以给我们带来的好处是，对这一门课的理解可以更加地深入，对于以后想读研进行更加深入研究的同志，这是有一定的好处的，对于分数当然也是有很多的好处的。第二个选择，我们可以不去追求那么完美，我们可以把时间分出来，去学习对我们的就业很有帮助的技术，这个选择的好处最终就是转换成一份自己满意的工作了。如何取舍，最终还是靠个人去思考。这里仅仅是提供文字供阅读参考。

另外，这里稍微多讨论一个问题，那就是在学习这一门课程的时候，如何去选择和使用 Linux 操作系统。课上老师给出的方案是使用 VMWare 虚拟机去安装 Ubuntu 的发行版。我从个人的角度给出我觉得可能可行的另一个方案，那就是在自己的物理机器上安装双系统，因为这样一方面可以锻炼自己的动手能力，另一方面，这种方案可以完全地把硬件的性能发挥出来，让我们体验到真实的 Linux 是如何地风驰电掣，而虚拟机的方式则会损失很多的性能。对于 Linux 的发行版的选择，我和我的室友目前更青睐的是基于 arch 的 manjaro plasma(kde) 版本，它对硬件的兼容性更好，出问题的概率更小，对程序员更友好，更容易让我们可以去体会到 Linux 的魅力。如果有大把的时间和精力，那么，去折腾一下 arch 也是一个不错的选择，身边的在技术这一块**非常**高明的同志有些会选 arch，仅供参考。

数字逻辑

这门课没有什么特别的实验。我唯一的小想法是，写作业的时候可以使用编程语言辅助一下，Python 是一个我看来比较好的选择。

算法设计与分析

这门课的教材是《算法导论》。

我记得我们当时是只会抽取其中的 6 个章节来进行讲解。实际上，这本书除开附录有 35 个章节。如果想连带证明一起看透吃透的话，很不容易。所以，这里就拣在上机的过程中会遇到的问题进行叙述。

首先，我个人建议在对书中的伪码作具体的实现的时候，可以使用 Python 或者 C++，可能实现起来难度稍微低一点，当然，如果 Java 或者 js 学得比较好的话，那么，也是没问题的。

然后就具体讲一下实际的上机过程中可能会遇到的问题。这门课我们当时很多时候是白天讲两个算法，然后晚上就要对照着教材，把算法实现一遍，当堂去到前面给老师看、演示，给参数作不同的修改，讲一下关键点，如果是纯粹的照搬别人的代码，那么，得到的评级可能不会达到 A 或者 A+。关于这个算法的实现，如果事前没有相当的准备的话，那么，晚上的上机时间可能会比较仓促。有些算法老师讲完之后我们可能并不能立马理解，而我们当时上课恰好更多是下午的七八节，刚上完，吃完晚饭就要去上机了，如果课上没有理解，那么，上机就是不太妙的。此外，这门课课上也有小测验，比如，讲一个图方面的算法，那么，会让大家在课上根据一个题目构造一个图出来，我个人觉得可能不太适合我，但是老师的规矩是这个样子，那就去尝试改变自己。这个和上机放在一起，我能够给出的参考意见是，可以利用平时的时间，多看多想，也可以提前去实践，关于算法导论，网上的材料可以说是相当多了。

还有一个点要注意，有的实验会要求我们把算法进行动画可视化，这个要有一点心理准备。

对，一定要提前学。我当时的印象里，打过 acm 或者有过信奥基础的同学，在这门课上是何等地意气风发，好吧，在其他实践课上也是一样，而他们那种程度，随着我们其他的大部分的同志把时间花上去之后，是可以缩小差距的。

软件工程理论与实践

课本就是机械工业出版社的《软件工程》。

这门课没什么很难的实验，可能也会使用到一点不同编程语言的测试框架。像 Java 就是 junit，Python 可以使用 pytest 等等。

这门课可能没有什么具体的坑，如果是对加权比较注重的同学，就一定不要错过每一节课，这门课的老师治学态度非常严谨。有问题老师也非常欢迎同学们去和他探讨。我现在还记得我当时问了一个算是有点“奇怪”的问题，当时讲到软件工程的最佳实践，关于函数命名方面的，课上讲函数的命名不要太短，一定要把含义给表示出来，我就问某一门编程语言函数名最大的长度可以是多少？这个问题在我现在看来可能问一下搜索引擎那是很容易解决的，可是那个时候，我问了，老师也给了合理的解释，那是和问搜索引擎不一样的体验。老师不以我的问题为幼稚，这是我觉得很不容易的地方。不同的人有不同的体会，这只是我个人的体会，仅供参考。

大三

大三上

微机接口与原理

选修课。我们当时有很多人为了在加权上获得更多的优势，放弃了这一门课。因为这门课想要拿高分确实是比较难的。这门课的内容也是比较难的。这个完全是看个人的选择。授课老师和操作系统是同一位。对硬件有兴趣的同学可以尝试。

这门课的上机其实不难，根据老师给出的材料，最终把实验的结果做出来不是很困难，而且，这门课的实验可能更多是给我们体验，所以我们最终提交出相应的实验报告能解释清楚实验的过程即可。

数据库

数据库这门课的老师和数据科学导论是同一位。因此，数据库的实验材料依旧是比较完备，根据实验材料完成实验问题不是很大，而且也是有助教，有些疑惑的地方，其实可以稍微多麻烦一下助教，只要它有利于我们把问题搞得更加清楚。

这里多说一句关于上课方面的我的体会。这门课的课程，我觉得很值得线下去听课。到了大三下这个阶段，我想，很多人应该是已经对数据库有了很多的了解，相应地，也会有很多的困惑，同时，也会有不少的同学在准备大三下的暑期实习了，毕竟在当前这个环境下，有些公司是从大三的寒假开始招聘大三下的暑期实习生的，比如微软中国。那么，有些公司在数据库方面还是会问得很多的，有些原理，可能在老师讲解之后，或者在我们和老师探讨之后才能够理解得更加清楚，这样在面试的时候也可以应对得更加自如。

编译原理

这门课，依旧是专注于最后的课设。

最后的课设是要求我们利用 flex 和 bison 来实现一个简易的 C 编译器。使用的语言是 C 语言。C++ 应该也可以。这门课最大的我觉得可能会有点问题的点是老师无法给我们过多的我们想要的指导，因为实际授课的老师平时主要的研究方向并不是这个领域。这样的话，我们就需要自己去多学、多看。也可以去多看看网上的别人的实现。

我们当时关于实验推荐的教材是这一本，《flex与bison（中文版）》。有时间的同学，我当然是建议可以尽早地阅读和做一点关于书里面的实现。

这一门课应该也有变通之处。这一门课的实现其实可以自己使用纯粹的 C 语言来作语法分析和词法分析，而不是借助 flex 和 bison。以及，这门课可能使用 Linux 下的 C 语言环境要更好一些，在 Windows 中可能会遇到一些兼容性问题，而根据我当时的经验，就这个问题询问老师可能不会有能让我们太满意的结果。

计算机组成原理

这一门课程的实验需要我们在暑假里面把老师给的材料给预先去阅读学习一下。尤其是想要取得高分的同学。因为这一门课程的老师和数字逻辑的老师是同一位，所以，在数字逻辑快要结课的时候，老师就会给出相关的提示。

关于这个老师对实验的要求，我这里也多说一句，那就是，这位老师打分的标准当然是实验的完成度越高越好。而对于完成可能有些困难的同学，老师的意见是这个样子的，尽可能去完成，如果实在无法给出很完整的报告，甚至做的内容都不是很正确，那么，其实最后给出的分数也是不会故意让学生为难的。 所以，我们放开手去做、去尝试就好了。

数学建模

这里提一下最后的上机的题目。我们当时是要在晚上上机的三个小时（可以适当延时一到两个小时）完成一道题目的建模与求解，可以使用任何方法，如果使用计算机求解，那么需要给出代码。

对这个上机的建议是，事先了解一点 scipy 或者 matlab 之类的数学方面编程的知识，最后做起来可能会更加地得心应手。

数字图像

大概会让我们去调用别人已经实现的模型、库等等。比如 PyTorch、yolo 等。

计算机游戏引擎

没上过。

软件体系结构

一门偏文档性的课程。

大三下

计算机网络

计算机网络的重要性我想我是不需要介绍的。我这里谈一下在实际的实践过程中可能会遇到的问题。

对于平时的上机实验，有让我们使用 Wireshark 来进行抓包的，也有让我们使用 C/C++ 来调用 Winpcap 进行抓包的。然后就是，Winpcap 其实有点老旧了，而 npcap 是它的一个现代化替代。这些其实是可以变通的地方，老师都是可以理解的。

稍微讲一下最后的大作业部分，我们当时的课设是做一个流量地图，就是对流量抓包之后使用 whois 识别其 ip 地址，然后做成可视化的一种流量地图。这个可以利用前端来绘制。这个其实也就要求我们前面的技术基础要打好。

多说一点，这门课老师讲得很好，课下同学们不管是线下还是邮件和老师交流，老师都非常地耐心给出细致的答复。但是，这个阶段已经是大三下了，很多同学忙于找实习，这个课程就来不及去细细消化，

大数据与云计算

可能偏理论和文档性的一门课程，正常地上课、和老师交流、完成任务就应该问题不大。

我当时最后的大作业是小组合作买一个服务器，部署一个 Web 项目，然后我们可以通过浏览器正常访问即可，项目也不要求很复杂，大二实训时做的那个简单的 Web 项目就可以。关于这个网上的博客、教程其实很多，应该不用去多讲。

软件质量与测试

可能偏理论和文档性的一门课程，正常地上课、和老师交流、完成任务就应该问题不大。

除此之外，我们当时还写了一点点 Java 的测试代码，使用的是 junit 这个框架，如果是之前对于 Java 花了工夫的同学，这里问题不大。

软件过程管理

可能偏理论和文档性的一门课程，正常地上课、和老师交流、完成任务就应该问题不大。

ERP

可能偏理论和文档性的一门课程，正常地上课、和老师交流、完成任务就应该问题不大。

大四

大四基本一节课都没有了，如果前面选修课学分修满了的话。因为我是在这个阶段就处于无课的状态，所以，就没有什么特别的经验可讲，虽然这个学期可选的课还是有很多的，比如机器学习、移动 App 设计、计算机图形学等等。以及，由于大三的暑假末期保研的结果就已经定音了，所以这个学期的课对加权也没有影响，上课的人自然也不多，除非是兴趣使然。

课程外

前置：我们需要尽早地去获取可以没有限制地访问国外的技术文档、技术社区内容的能力。这在我看来可能有些人已经听过很多次了。我不知道我在这里再去这样说一遍会不会引起大家的反感，但是，不管怎样，我觉得我还是再复述一遍可能会更好一点。

小的习惯

多逛一下我们专业相关的论坛，多关注一点我们这个行业的前辈，多阅读一点我们这个专业的同侪的博客，多去尝试我们这个专业的新鲜的技术。

我们专业的论坛或者说网站我这里就简单列举几个，希望能给同学们带来一点启发，

- stackoverflow
- v2ex
- Hacker News
- Reddit Unity 社区
- Reddit Godot 社区
- Reddit VSCode 社区

其他的内容，其实我们也可以去 twitter 去探索，比如关注一下 vue 的作者尤雨溪的账号，或者一些其他在这个行业已经沉浸了多年的前辈的账号，我们可以在打基础的同时去见识一下上层建筑究竟是长什么样的，这或许可以给我们的未来带来一定的启发。

检索信息

相对来说除思考问题之外的最重要的问题，信息检索的能力。

Google，StackOverflow，GitHub 可能会慢慢成为我们检索信息的主流。以及现在比较新的工具：ChatGPT。

这里的话，Google 换成 baidu 和必应其实也是可以的。对于技术方面的问题，它们几个各有千秋。但是如果涉及到需要安装开发套件的情况下，可能 badidu 不是一个好的选择。

StackOverflow 是我们在遇到实际的技术问题时可以求助的对象，这个其实结合 Google 来使用就好。对于 Google 搜索，我们也应尽早尝试和习惯使用英文检索，其效率和命中率可能会更高一点。

对于 ChatGPT，它当然是现在的一个非常好的用来辅助我们编程的工具。除了平时在实际的开发过程中遇到的问题，我们也可以在阅读技术书籍的时候，把书里面我们觉得不清晰的问题拿过去问它，举一个具体的例子，如果书上给出一段不完整的核心代码，但是我们想实际的把那个代码跑起来测试一下效果，那么，我们就可以把这个问题交给 ChatGPT，让它给出一份完整的可运行的代码供我们测试。

实训

我们很多大四的学生，不管是在院长午餐会，还是后面的座谈会，院长和书记都表达了让我们临近毕业的学生大胆地给学院的建设提出一些诤言。其中，我们提到的一个很重要的问题就是实训的问题，尤其是实训的时长和内容的问题。

可以预见的一件事情，有一些小组或者说同学可能最终完成的实训的效果可能不会达到自己预期的效果。造成这种结果的原因有很多。

实训之前，其实我们也是到了后面才知道，关于实训的内容，其实是可以有很多的变通的。比如，这里有提出一些编程语言、技术框架或者说一些项目来供我们选择，我们其实并不需要被这个东西所限制。比如说，大一的实训，有 C++ 的 Qt，但是实际上，使用 Python 封装的 PyQt 也是一样的。对于游戏引擎的选择，其实也有很多，除了 Unity，我们也可以使用开源的 Godot 来尝试一下。甚至我们可以使用 C++ 的图形库来进行一种相对原生的开发，比如使用 sfml、sdl 库来渲染我们的游戏。对于实训题目的选择，我们当然也可以不局限于给出的题目。

实训的过程中，第一次分工可能会遇到各种各样的问题。由于技术和时间的限制，在实际的实训的过程中，可能会比较匆忙。这个是不要紧的，我们只要能够拿出我们现阶段可以拿出的相对最优的结果，自己不会对自己产生不好的怨言就可以了。后面我们依然有机会去进行迭代、去优化。

欧阳老师也提过，每年的实训之后，其实有很多小组会继续接受他的指导，我们可以利用好后面的时间，沉稳地再去学习、再去实践，去迭代出我们满意的结果，这个对于提升我们对技术的热情和信心是很有帮助的。

前面课程内我们其实已经探讨了一点关于答疑的事情。这里就讲一下我对实训过程中问问题的看法。我认为我们可以不厌其烦地去问老师。我们不要担心我们问出的问题很幼稚，因为大一这个阶段本来就是相对早期的阶段，在这个阶段，我们不必对自己有太高的要求。而且，问问题的时候，如果老师们给出的是笼统的答案，比如说，你去用搜索引擎呀，你去查什么什么文档啦，这个时候，我们一定要把老师抓住，让他们给我们写一个实际可以跑起来的 demo，或者，让他们实际去用搜索引擎、查文档的方式实实在在地解决一个问题给我们看，最终一定要跑出结果，不可以纸上谈兵，从我的角度来看，这个对我们的技术的帮助可能会更大一些。

去年寒假，大三的一个学弟问我大三下的实训内容，说是要提前准备一下，因为之前可能大二的时候做得没有让自己满意。通过这个事情，我了解到可能很多人他是有那样一种愿望的：希望自己做出的项目能够让自己满意。那么，如何使用一种成本不太高的方式来达到自己的预期呢？我能够给出的回答是，提前去了解，平时带着学。

此外，我想说的是，不要给自己太大的压力和负担。我记得当时大一答辩的时候，有一个评委老师提到了这样一句话，“啊，同志们呀，你们将来是国家的栋梁呀，你们现在拿出来的东西，你们自己想想，有没有创新性，有没有意义呀？”，我当时不觉得有什么，但是现在，我同意他前半部分的看法，我们将来确实是建设国家的中坚力量，但是，英雄是一生下来就是英雄的吗？英雄没有弱小的时候吗？你们有没有想过在英雄还很弱小的阶段你们应该做些什么呢？

额外

一些小的特性

特别

我个人的见解就是，我们专业的同志，在专业的发展上面，在后期一定要做到特别，当然，这也是我个人的见解。

什么是特别，以及为什要特别？所谓的特别，其实也就是我们在发展的过程中去发展一种或多种我们独有的不太容易被别人替代的技能。比如熟练地使用 vim/neovim，或者深入地去钻研一门别人可能会因为繁琐或困难而放弃的技术。这很有可能会成为我们的核心竞争力。此外，我们学院的人，其实本质上每个人的心里都是有一股傲气的，每个人都不可能真正地承认我比别人差，那么，我们也可以通过培养特别的专业技能来证明这一点。

专注

前天睡前读到的一篇博客，

https://manateelazycat.github.io/think/2023/05/10/developer-focus.html

> 1. 编程环境的专注： 编程环境的专注主要有两个方面， 一是避免环境打断思路， 一会键盘一会鼠标的操作会因为大脑去思考下一步点界面的什么按钮而容易打断真正重要的逻辑思考，记住我们的大脑其实是协程思路， 切换上下文后容易忘记上一件事情的细节； 二是环境插件的稳定性， 平常折腾一个插件提升效率后， 插件环境应该有长期累积效率提升， 不要因为编辑器 或 IDE 发行商发新版本引入新的 API， 导致我们定期要从头折腾一遍插件环境配置，这么多年我觉得符合专注的编程环境只有全键盘操作的 Vim 和 Emacs；

对于这个观点，我大致是认可的。

这里我就简单列举我自己目前可以提高个人专注力的环境：Neovim 和 VSCode，在输入这一块，我觉得还有一个可以值得尝试的东西就是输入法的双拼加辅助码方案，它可能更符合我们这个专业的同志的习惯，给我们带来更好的输入体验。当然啦，选择什么样的一种编程套件是个人的选择啦，只要有助于专注力的提升，我认为都是值得认可的。

折腾

再怎么折腾也不为过。折腾也可以有一定的取舍。根据时间的多少，我们可以作出适宜的选择。

至于我们可以折腾什么东西，那就太多了。比如，体验不同的操作系统。部署和体验别人的开源项目。折腾一下不同的输入法软件，体验一下不同的输入法方案。体验一下不同的编程语言。

折腾的源动力其实还是兴趣。这里形象化一点地描述就是兴趣驱动学习和开发。

在这里我其实有一点想强调一下，那就是，折腾终究是一种主业之外的一种类似于消遣的东西，我们的核心专注力还是要放在更加重要的事情上。

可持续发展

我们 19 级的学生可能有很多人有这样的习惯，喜欢使用暗色的编辑器背景，有各自喜欢的编程自己，以及喜欢编辑器界面的字体调整得大大的。这个其实是在数据结构这门课上受到了院长的影响。院长对我们说，啊，要注意这个程序员职业生涯的问题，要对自己的眼睛好一点，提出了几点建议。我们很多人深以为然。于是就有了上面的几点习惯。

实际上，我认为这个问题也是有必要去考虑的。如果我们真的对这个行业怀有深切的热情，那么，我认为是有必要培养一些生活中的小习惯的。我这里就抛砖引玉一下。

首先是，上面的编辑器的设置问题。这个点属于见仁见智。每个人都有自己的习惯。能够让自己的眼睛感受到舒适即可。

然后是，我们可以买一块声音小一点的机械键盘，比如说，红轴的键盘。然后，配上一块合适的键盘托，可以在一定程度上缓解我们手腕疲劳。

暂时就这两点小小的建议，可能也是比较常见的问题。

一些经典的问题的个人见解

1、保研、考研还是工作?

七十分主义者、八十分主义者和九十分主义者。依旧是从技术的角度来谈一下我的看法。

在我看来，哪一条路都可以。这个回答可能是敷衍了一点。

我个人大一是九十分主义者，后面就慢慢变成八十分主义者了。这个完全是靠个人选择，我是觉得我在专业课上是没有办法获取太高的分数了，就退而求其次，在不追求分数的情况下，尽量去搞懂那些很重要、很有趣的知识。这是我个人的想法。

而至于保研、考研，因为我没有走这两条路，说出来可能会误导大家，这里就不去多嘴。

2、如何提升自己对技术的热情?

系统、软件、外设。

就拿上面的三个方面来具体。

系统方面，我们可以选一个称手的系统，有人偏爱 MacOS，比如我的一个室友，有人主力工作系统是 Windows，比如我，有人更喜欢使用 Linux，这都是可以的，在我看来只要有长时间的专注，它们的可用性都可以达到一种很高的程度，我们的系统使用环境变得很便捷之后，也会反过来更好地把我们的时间抓住。

软件方面，我们可以大胆地去尝试，就拿编程相关的工具来举例，我们不仅要去听别人讲哪个工具多么多么好用，而是自己实际动手去尝试，像是 vim/neovim 啦、emacs 啦、helix 啦、VSCode 啦等等。其他一些终端软件，比如 Windows Terminal、iterm2、alacritty、kitty、tmux 等等，我们都可以去实际尝试一下。还有一些独特的特性，比如 VSCode 的光标动画，它在一定程度上是可以提高我们编程时候的幸福感的。这些东西只要不是太过沉溺其中，它都可以给我们带来很多正向的反馈。

外设方面，我大致上有两个建议。第一，就是购置一块称手的扩展显示器，这对我们看视频、文档学习来说很重要。第二个就是购置一块称手的键盘，当然，一定要配置上键盘托，呵护我们的手腕，这对于延长我们的编程生涯很有帮助。

3、系统的选择?

- Windows
- Linux
- MacOS

其实在之前已经提到过，三者都可以，对于我们专业来讲，没有特别的不适配性，使用到后面，在我看来，对我们技术的提升的影响其实不大。如果有可能的话，我觉得可能都尝试一下比较好。如果经济实力允许的话，那么，可以多添置一些设备，多尝试一下总归是好的。

4、peer pressure(同辈压力)?

及早地抽离出来。我知道每一个年级里面都有很多人喜欢在群里面发表一些超出我们一些人理解的东西，可能是一些技术方面的东西，可能是一些网络上面流行的东西，这可能会给一些同学造成一定的困扰，尤其是一些从小地方来的同志，思维可能暂时还没达到跳脱的境界，如果你暂时还做不到像费曼先生那样，看到一个新鲜的东西，“啊，我也要去尝试一下，这太有趣了！”，甚至你可能会产生压力，啊，为什么别人会这么多东西呀，为什么别人说话那么有趣呀？为什么别人可以那么勇敢地大声地批判这个、批判那个呀，我几时才能做到那般？我想，这可能和过去的教育背景、学习态度和风格有关，这些等过几年看，其实都是没关系的。如果群里面有那种类似于不去大声叫嚷，而是以自己的实际的厚实的在专业领域的积淀给同学们认真解惑的人，那当然是很好的（19 级是有这样的同志的，我非常地钦佩），而如果没有，那我们不妨把身子埋下来，我们自主去提升自我，坐一坐冷板凳，不必去理会外面的叫嚣，时间和付出自然会给我们更好的答案。

我的室友中，有一个是从高中开始学习 Java，大学加入联创。也有一个在技术方面过去并没有太多积淀的室友，他走的是保研路线，然后保研之后，专注技术，大四加入 memo 游戏工作室。他们两个都是我非常佩服的，他们的能力当然都是在我之上的，我举这个例子，其实也只是为了给大家带来一点别的视角，看一看我们走不同的路可以达到什么样的程度。

谈到压力，我又想起了另一个出国的球友给大一的同志对于大学生活方面的建议：享受大学生活。在我看来，如果我们怀着一种对技术的憧憬和热情、对知识的渴望，把这些融入到我们的学习生活中，我们的大学生活确实是可以过得很快乐的。

如果有机会，我真的希望大家可以读一读我们专业的大四学生的毕业致谢，可能有人已经读到过部分的大四学生的致谢，其中很多是有对这个专业和大学生活的真切的思考的，甚至我自己在读完好友的致谢的时候，会发现，原来他在当时还有这样的困扰，原来他在那个阶段遇到过那样的问题，原来他还给学院提供了那么多的改良建议等等等等，如果有机会读到，我相信，对大家现阶段可能会产生的一些问题一定会有一定的启发。

5、竞赛类活动的参与?

从自身的实际出发。我们大一大二实在是太忙了，如果有可能的话，可以等技术先积淀积淀，等到大三的时候去参加，虽然可能对找实习来说有点晚了，但是对于保研、秋招、春招来说还是用得上的。

6、是实习而不是兼职。

可能有一些同学会选择在寒暑假期间做一些兼职，对此我个人的建议是，在可能的情况下，把实习排在第一位。我们专业的时间是相当宝贵的，可能有些同学在现阶段会感到一定的困顿，但是从长远来看，现阶段的箪食瓢饮是暂时的，我们不需要等待太多的时间就可以过上我们想要的生活。而在等待的过程中，它可能需要我们去尽可能地利用好这段时间。

答疑

由于我的失误，这个模块被我忘掉了。如果有人觉得我的文字或之前的现场分享有讲不清楚的地方，欢迎给我发邮件。

总结

比较匆忙，很多东西无法说清楚，但是，这并不意味着没有机会说清楚。如果大家有疑问，可以给我发邮件，或者通过我博客里面的社交账号来联系我，我会尽我所能地以我现在的这样一种经验经历向大家给出我可以给出的回复。

最后，给大家推荐两个我个人比较喜欢的人物，一是费曼先生，我们可以通过《别闹了，费曼先生》来了解他；另一位是《清明上河图密码》里面的张用。他们都是极其不在意世俗意见的人物，对于自己喜爱的事物，可以几乎奉献一切，我很喜欢他们的生活和治学的态度，如果有感兴趣的同志，或者说，受限于过去的环境，性格上偏社恐或者说内敛的一些同志，或许可以从中得到一点启发，而大学，恰好是一个很好的实践和尝试的场所，大胆尝试，当然啦，维持现状也是完全没有关系的，以上所有的内容，也只是我一个人小小的看法啦。声明：绝不是价值观输出。

我的邮箱是：lxyfl6688@gmail.com