计算机科学进阶班招生

自从开设计算机科学基础班（集体班，一对一，阅读班）以来，收到了很多学生的好评。我很高兴地看到很多完全零基础的人，真正从零开始，学会了计算机科学最精华的部分。这让我大受鼓舞。从我自己的探索，遇到的各种社会问题，我再次深刻地认识到教育的价值和意义。

在这种前提条件下，我已经有开设计算机科学进阶班的想法一段时间了。目前已经有 13 岁的小白鼠同学试了一节课，我发现一个孩子真能在短时间内学会一些曾经只有博士生才能学会的东西。 我也曾经试过即兴教一个朋友大提琴。之前她从未碰过大提琴，结果只花了一个小时，就能拉出巴赫 Prelude 的第一个小节。当然仍然是不流畅的，但原理已经掌握，自己多练就会不断提高。掌握本质原理，不断提高。师父领进门，修行在个人。我感觉这是我领悟到的教育的真谛。

关于计算机科学进阶班，我还没有非常确切的内容和计划，但最初开设基础班的时候，其实也是一样的情况。除了前两节课的内容，后面的其实都是我根据教学情况，临时想出来的。但我最好还是大概列出一些内容，让同学们心里有个底。

课程大纲

大概的内容范围如下，虽然可能临时根据情况更新或增删：

1. CPS（continuation-passing style）变换基础。从零开始，理解 CPS 是怎么回事。
2. 使用 CPS 实现程序的并发执行。根据 continuation 这个统一的程序语言概念，理解操作系统的进程，线程，以及 Go 语言和 node.js 等系统的所谓“大并发”。
3. Scheme 语言基础。因为 Scheme 语言的干净设计，我们将切换课程语言，用 Scheme 来学习后面的内容。学生会用 Scheme 实现递归函数，然后用它写一个类似基础班的解释器。
4. 理解和实现 call/cc 操作符。理解 call/cc 是什么，并亲自实现 Scheme 语言的 call/cc 操作符，而不只是会使用它。然后使用 call/cc 做一些有趣的事情，比如实现并发。
5. 理解和实现“自动 CPS 变换”。这个就是人们常说的“王垠 40 行代码”，它的作用是自动把普通程序自动转换成 CPS 形式。学生将理解它的原理，并且自己实现出自动 CPS 变换。
6. 使用“自动 CPS 变换”的思想，实现一个微型编译器。这个领域叫做「compiling with continuation」。它使用 CPS 变换，

ANF 等理论基础来设计编译器，而不是传统的方式。虽然这个编译器的语言非常简单，但它揭示了编译器的本质，理解它会帮助学生将来写出任意复杂的编译器。如果有兴趣，学生可以自己扩展这个语言。

7. 逻辑编程（logic programming）基础。理解并实现一个类似 Prolog 的简单逻辑编程语言。由此从根本上理解逻辑编程。使用逻辑编程的思想，实现 OCaml 和 Haskell 的 Hindley-Milner 类型系统。
8. 程序的形式化证明。使用 Coq，学习形式化证明系统的使用。与程序测试不同，通过形式化证明，我们可以完全保证程序的正确。通过简单的例子，讲我将述形式化证明最根本的思路和原理。在某种程度上，这些思想就是数学证明的本质。

课程初步计划，应该有至少 8 节课。课程会有专门的交流群。每节课后，学生可以在群里讨论自己的理解和发现。我会临时想一些小练习给大家做，但这个课程相当于是大学“研究生班”的级别，所以课后主要是开放的研究方式进行思考和探索，而不是做固定的练习。我也许会建议一些论文给大家看，或者自己实现一些以前没试过的事情，或者探索学生自己感兴趣的方向，或者做一些小型工程实践。

收费标准

由于内容的难度很大，所以我不是很确定 8 节课能够完全讲清楚这些内容，可能会超出一些。我打算按照一节课 1024 人民币进行收费。课程的单位课时学费比基础班要低。最初的学费是 8192，包括 8 节课，每节课 2 小时。虽然按我的性格会尽量精炼，但如果无法在 8 节课完成，也许会延续 1～2 节课。延续的课时不另外收费。

由于课程内容会用到基础班的很多知识，课程的招生对象主要针对基础班（包括一对一和阅读班）已经顺利毕业的同学，但如果有自学过 Scheme 或者函数式编程（比如 SICP）的其它学习者，我也可以考虑他的加入。

报名方式

课程只招收基础班已经毕业的学生。基础班已经毕业学生，可以发微信给我报名就行。注意，请勿用微信转账。由于进阶班对基础要求比较高，课程不接受没有参加过基础班，或者基础班未能完成的学生。