**算法书签**

**您当代生活的算法说明书**

**适用读者：**

本书适合作为编程爱好者，想学习编程的学生，也宜用于专业程序员。从事项目编写，框架设计，算法研究人员也可以从中得到许多算法知识，并扩大知识面。

**区别：**

上课用书主要是利用Pure C语言进行编写，可读性较面向对象语言相差甚远，并且特化性太强，而本书则是根据元编程思想进行编写，对于模板跟抽象思想有着一定的深入讲解，并且本书摈弃了上课用书时的枯燥乏味的说明，转而多用常见的例子亦或是游戏算法来进行一定的解释。

**本书特色：**

本书通过简单的数据结构中的算法入手，通过STL中的部分源码进行解析，使读者能够不仅复习数据结构知识，还可以对于元编程的思想有着一定的了解。同时，本书还尽量会拓展一些可视化算法以及图形学中的一些优化算法，用以说明算法优化的重要性。并且本书会通过大量的小游戏实例来说明这些算法的实用性。同时，在课程深度方面本书也会由浅到深，从并非先涉及递归等高级语法，转而先从贪心，蛮力这种思想简单的方法开始写起，然后涉及分治与动态规划，但这些章节并非主要章节，更重要的是根据前面这些章节进行展开的后续课程，具体从线性代数到高等数学算法实现，从游戏思路实现到赌博概率计算，最后到机器学习初涉，这些章节才是本书重要的部分。

**本书目录：**

1. **导语**
   1. 程序 = 算法 + 数据结构
   2. 如果算法很好“玩”
   3. 算法与数学不可分割的关系
   4. O与N
2. **算法其实很好想**

2.1 贪心法思想及举例

2.1.1 思想解释

2.1.2 怎么能在一定时间内赚到更多的钱

2.2 分界线

2.2.1 开胃菜

2.2.2 STL初涉

2.2.3 容器

2.2.4 简单的算法函数

2.3 蛮力法思想及举例

2.2.1 思想解释

2.2.2 vector中的蛮力操作

2.4 分治法思想及举例

2.3.1 思想解释

2.3.2 sort函数的实现思路

2.5 动态规划法思想及举例

2.4.1 思想解释

2.4.2 走台阶 (注：这里开始结合之前的STL知识)

1. **生活处处有算法**

3.1 找零钱问题

3.1.1 蛮力法复习

3.1.2 动态规划法复习

3.1.3 贪心法复习

3.2 数据压缩

3.2.1 LZ77算法压缩

3.2.1 哈夫曼编码压缩

3.3 地图寻路问题

3.2.1 如何用

3.2.1 深度优先搜索

3.2.2 广度优先搜索

3.2.3 拓扑排序

3.2.4 关键路径

1. **数学算法好兄弟**

4.1 导读

4.1.1 数学的历史

4.1.2 数学与计算机的相似性

4.1.3 当代计算机中数学的应用面

4.2 线性代数中的算法

4.2.1 矩阵连乘

4.2.2 非线性方程组与牛顿迭代法

4.2.3 线性方程组的求解

4.2.4 超定方程组与最小二乘法拟合

4.3 高等数学中的算法

4.2.1 极限

4.2.2 微分

4.2.3 曲率

4.2.4 利用迭代法求定积分

1. **游戏动画遍布算法**

5.1 如何让你的算法看得见

5.1.1 线

5.1.2 多边形

5.1.3 曲线

5.1.4 圆

5.2 推箱子

5.2.1 迷宫问题的变体

5.2.2 A\*算法

5.3 五子棋

5.3.1 规则问题

5.3.2 估值算法

5.3.3 决策树、博弈树与行为树

5.3.4 实现一个简单的人机五子棋游戏

1. **算法让你生活更加舒适——人工智能算法初涉**

6.1 KNN 与 鸢尾花分类

6.2.1 没有模型的算法

6.2.2 拟合

6.2.3 实现分类

6.2 垃圾分类与贝叶斯公式

6.2.1 朴素贝叶斯公式与极大似然估计

6.2.2 分词与词向量

6.2.3 训练与模型

6.2.4 垃圾邮件分类

1. **结语——算法让生活更美好**

7.1 我们还需要学习算法吗？

7.2 AI算法真的很必要学吗？

附录：

附录一 部分可视化库推荐:

1. Easyx
   1. 介绍
   2. 安装教程
2. OpenGL
   1. 介绍
   2. 安装教程
3. Matplotlib
   1. 介绍
   2. 安装教程

附录二 一些学习算法/算法可视化网站

**举例说明某一重点章节编写内容：**

**第七章 结语**

有人说“结语不重要，大家又不会看，为什么要写呢？”其实不然，大家不会看结语这一章节大抵是因为这一章节内容枯燥又无味，顶多是将算法思想从新灌输一遍，并且加上一些劝说。但是个人认为，结语这一章节才是全书的重点，是一本书的点睛之笔，是一本书是否能够虎头虎尾的结束的象征，如果能够将一位读者从开头看到最后结语都能够接着读下去，那这本书个人感觉已经是非常成功了，如果读者能够从结语中吸取知识，能够活用于今后生活，那这本书的意义便已经升华了，它便不仅仅存在于某一时间段知识的获取上了，而是作为人生中的一个中继站而存在了，这便是在下写这本书结语的中心思想。

7.1 我们需要学好算法吗？

首先，学好算法这个定义对于每一个人都不一样。对于有一些同学来说，学好算法，就是能应付面试中的算法问题；对于另一些同学来说，学好算法，是能在算法竞赛中拿大奖；还有一些同学想要学好算法，是因为自己当下在学习人工智能；或者编译原理；或者操作系统；或者数据库；或者图形学；或者图像学，等等等等的时候，遇到了阻碍，发现算法是自己的薄弱环节；看到了吗？同样是学好算法，其实大家的定义完全不同。而且，一旦我们明确了自己的定义，自己的目标，其实，向着这个目标该如何去行动，并没有大家想象的那么复杂。很多时候，我们不知道如何去行动，关键是，不知道自己的目标是什么。

所以，首先要根据你自己想要将自己算法提升到什么程度想清楚。这一步是很重要的，有些人总是在盲目的学习，却不知自己真正所需求的，正如《论语》所言，“学而不思则罔，思而不学则殆。”学习一定要思考，这样才会有真正的提升。但如果你跟我说：“我的目标就是向上攀爬，直到顶峰”。那你的目标便是顶点，伸手摘星，纵使不得，亦不会满身污泥，请务必准寻自己的梦想。

说了这么多，可我们真的需要学好算法吗？虽然在计算机的世界里，算法无处不在，但这并不代表我们一定要掌握他们。业务上的“分级”，其实是一种很常见的现象。成功的打造一款网站，不代表你需要开发出浏览器，开发出运行在浏览器上的各种语言，以及一个底层的网络模型；成功打造一款app，也不意味着你必须先学会开发出一个iOS或者Android系统。在绝大多数情况下，确实不需要什么算法。即使需要，也真的很简单。那么我们是不是真的不用学好算法了？

或许是，但是请一定要记住，学习算法对你来说不一定有用，但与此相对应的，要想取得成功，就一定有别的什么，是有用的。算法不是技术领域的唯一的核心竞争力，但无论是一个人，一个企业，还是做一份事业，都需要有核心竞争力。什么都没有，肯定是不行的。有的人高中毕业却靠着写外挂的本领入职大公司，有的人iPhone 3代便开始开发iOS程序，3个月便坐到经理一职，这些都是个人的核心竞争力外显的结果，如果你没有像他们那样子耀眼的光芒的话，那请务必先将可以抓住的抓住。比如说，算法。

7.2 AI算法真的很必要学吗？

不知从何时开始，AI成为了时代的代名词，甚至“人工智能元年”等概念层出不穷，如今，这个行业正在飞速的发展着，甚至很多高中生便开始写机器学习，深度学习代码，可这真的是好事情吗？AI算法真的很必要学习吗？

如果你还是学生，或者将来有读研或者读博的想法，对于大部分理工科领域（不仅仅是计算机），近乎一定都会用到AI。学习是绝对有用的。

如果你在工作，但不是AI岗算法岗，我的看法是，学AI不是必须。不过如果你有充电的时间和意愿，对于技术工作者来说，多少触一下AI的基础知识，有百利而无一害。

为什么这么说？现在主流的AI技术——机器学习，解决问题的方式，和传统的计算机解决问题的方式是完全不同的。机器学习解决问题的方式是”基于数据“，而传统的计算机算法解决问题的方式是”基于逻辑“。这两种思想之间有巨大的差异和鸿沟。

在这里，我们不讨论那种思维方式好。实际上各有千秋，适用的场景不同。但接触机器学习，在我看来并不是一定为了找一个AI相关的工作，而是多一个思考问题的角度。同时，对很项目，事情，新闻或者人，都能多一分属于自己的判断力，而不是人云亦云。这也一定程度减少了盲目跟风（对某些项目或者投资）所带来的风险。

也正是因为这个原因，我认为从教育的角度，小学中学的教育不说，但是本科教育，尤其是计算机教育，AI一定会慢慢成为一种”必修课“。因为它不仅仅是一个技能，更是一种思维方式。AI如何解决问题；能解决什么问题；有什么坑；怎么看待AI给出的结果，这些思考将慢慢成为一个技术人的基本素质，这些都和找到一份具体的工作没有关系。

看到了这里，你因该是把所有章节都看完了吧，那这本书应该算是成功了吧。陌生的读者啊，请你一定要记住，所谓的教育，就是忘记了在学校所学的一切之后，剩下的东西，你所学习到的东西，并非里面各种各样的代码逻辑，而是思考问题的方式与方向。