像KMP、Manacher、Splay、LCT这些算法，写得通俗易懂的文章比较少，或者对于新手来说阅读论文有一定困难，可以用这种方法自学：

以下方法纯属个人方法，不推荐已经具有快速阅读论文能力的同学、时间较为充沛的同学或具备较强思维能力的同学使用。

（如果你会读论文、时间够或者能自己去脑补很多算法，那就好多了。）

（一）先上网搜索，大概看看要学习的算法是怎么运作的，理解一下整体思想。

（二）然后在网上找到一道模板题，并且找到别人博客里的对应代码。

找一份好一点的博客，我自己是从hzwer和popoqqq博客找的（见YZOJ左上角或百度）。

（三）接着自己出3~5组数据，做阅读程序写结果。

也就是一行一行看别人的代码为什么是对的，为什么这样写。有些代码不太长，或者你是能理解其中一些函数的意义的，所以读代码是能做得到的。

这么做不是说要抄袭，而是说要透彻理解。目的是增强对算法的深入理解（这样能学到细节处理、边界处理），学习别人优秀的写法（因为自己第一次的写法往往不是很简洁），同时增强出数据的能力（你要出3~5组数据），增强读代码的能力（没什么好说的）。

这个过程会比想象的辛苦，不过至少有个明确目标，比盲目地学，或者只是看看文章口胡一下实际上连模板题都不会写要好多了。

（四）复习检验。抛开别人的代码，自己再写一遍模板题（或者新换一道模板题）。确保你不是在抄别人代码。

（五）做一些拓展。百度这个算法，比方说百度“后缀数组”、“后缀数组 难题”、“后缀数组 好题”，看别人做了哪些题，看题目，自己思考，看别人写的题解。

这一步的目的是拓展、加深理解，增加阅读量。

看完之后，挑选觉得最难写、最有挑战性的题来实践，写成代码。

这样，一个算法差不多5~7个小时就能摸清楚基本知识。当然，进一步思维的提升还要做更多难题。

（六）学习一个算法，要做到：

①会出数据（会叉错误算法，会卡满标算复杂度）

②会对拍（会写对拍用的暴力、数据生成器、checker，比方说欧拉回路、2-SAT的对拍）。

③能自己想出多个拓展（甚至能造题）

因此，在学习的时候还有第六步：自己思考拓展，尝试举一反三。

这一步要做成什么样呢？大家可以以CJK写得next[][26] 为举一反三的参考。

这一步要花多久时间呢？不知道，取决于你的举一反三能力。（比方说差分约束，我们就拓展过判解是否唯一；最短路拓展过01bfs和边权和不超过一定范围的bfs，网络流有各种带/不带上下界的，最大/小，费用/可行流）

直到最后，在考场上遇到这个算法相关的题，不会因为算法的基础掌握不扎实而写不出来，而是更多因为思维难度，比方说题目复杂的包装而做不出来。