C语言从小菜鸡到老司机

原创 科岩 科岩成果 2021-06-28 08:59

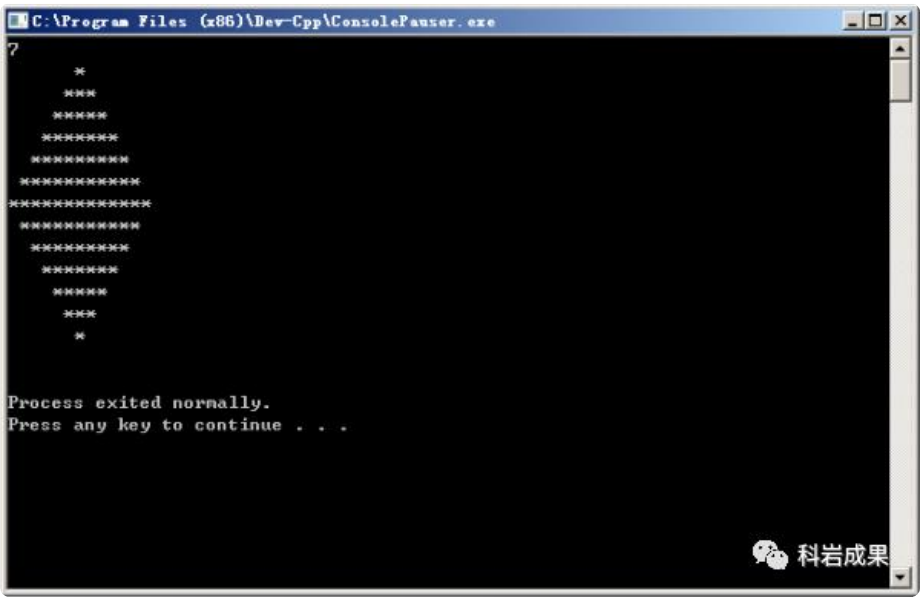
收录于合集

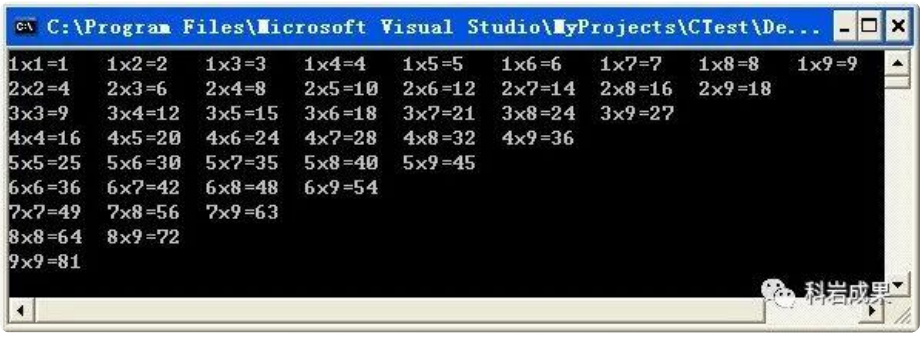
#C语言 1 #C语言学习 1 #C语言路径 1 #C语言入门 1 #C语言学习路径 1

大家好我是科岩。

提到C语言呢，大家第一时间头脑里边想到的是什么？

是不是在黑乎乎的命令行里，用星号输出一个菱形，是打印99乘法表，是杨辉三角？





如果你想到的是这几个例子的话啊，恭喜你，没错，你就是那个C语言小菜鸡。

小菜鸡往往会有各种各样的疑惑，这C语言不是说特别强大吗，会画个菱形，会打印个杨辉三角，就能当饭吃了？

有了这样的疑惑就会导致觉得学习C语言没什么用。然后就厌学，然后就学不好，学得不好就一点成就感都没有，逐渐的会失去了学习的动力，然后就更厌学，厌学又厌学，厌学何其多，不如放弃算了，所以在考试中挂科，在补考中又挂科。

造成这样现象的原因是什么呢？理论与实际的生产环境，有非常大的距离。

一个是象牙塔里边的C语言，除了画画小星星，什么都干不了，一个是打工人眼里的C语言，简直无所不能。

那么到底真实的C语言是什么样？我们真的有必要学习c语言么。

这篇文章我们就来聊一聊，**C语言到底能做什么，还有没有必要学习C语言，该怎么学习C语言。**

为什么要学习C语言

我们不说它那些语言特性的优点啊，不说什么速度快呀，可移植啊，代码紧凑啊，等等啊，这些我们不说。因为语言特性和我们自身的关系也不大对吧？

那么为什么要学C语言呢？我给你三个理由。

1．C更偏底层

C是一门造轮子的语言，更加贴近操作系统，有多贴近？连windows、linux操作系统都是C写的，学习c语言，能非常近距离的**「看到」**一个程序的运行流程，编译、汇编、链接、运行。

内存的申请和释放，每一个变量是怎么存储于内存中的，他们又分别存在内存中的什么区。

学习C语言要直面操作系统的特点，这个过程能学到很多底层相关的知识，相比C语言，学习其他语言可能就没有这么底层啊，很多东西都会有一种雾里看花的感觉，无法深入的理解本质。

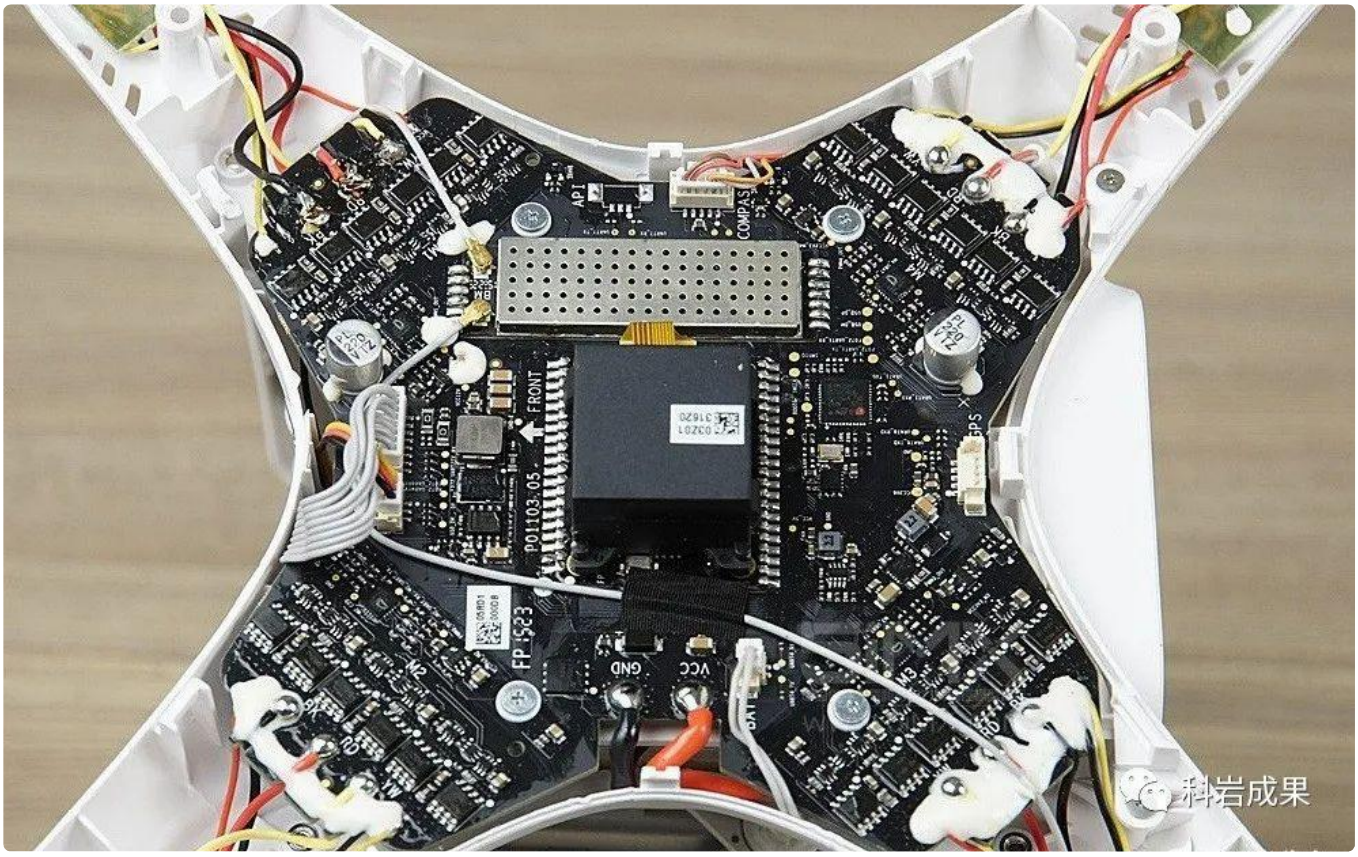
学习这些底层知识与机制，对于你程序里排查问题、做性能优化，非常有好处，会帮你开阔思路。

2．就业面更广

说实话，在我的认知里，没有C语言做不了的事情，只有C语言不适合做的事情。我举三个必须使用c语言的例子，三个完全没有一点关系的行业。

- **嵌入式，就是做和硬件相关的行业**

小到一个儿童玩具、大到洗衣机电冰箱，无人机、汽车，甚至航天飞船，都有嵌入式系统的身影，而这些系统里面的代码，基本上都是用C语言编写的。

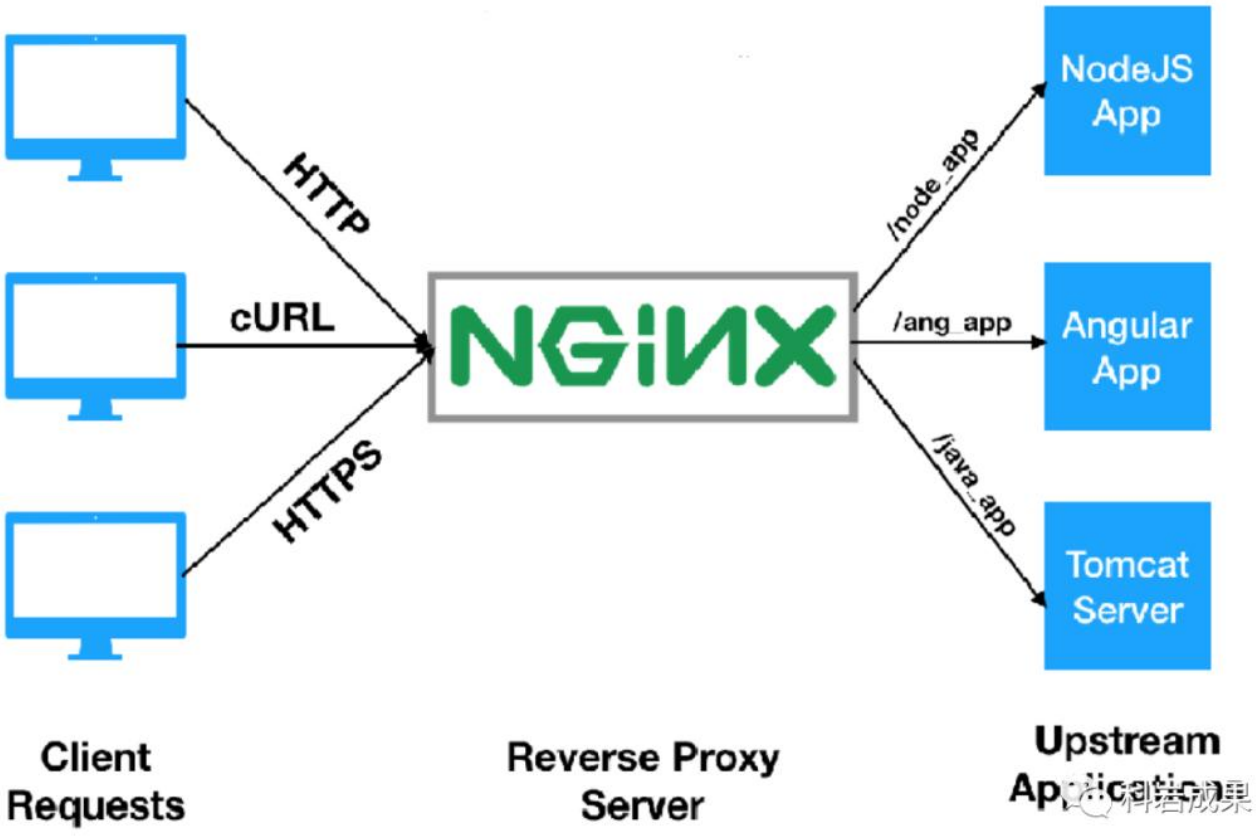


- **网络游戏**

可以这么说，几乎所有你听说过的网络游戏都有C和C++的身影，比如王者荣耀的后端服务是C和C++写的，其他语言的性能满足不了游戏的实时性。

- **互联网web服务**

著名的高并发服务器Nginx就是C语言写的，你现在手机上装的APP，他们的服务端架构基本都会用到nginx，像淘宝双十一这样的海量并发请求，它背后所用的支撑的技术体系，也有nginx的身影。



大家还知道哪些行业只能是C语言来做，欢迎留言，让其他的小伙伴也看到。

所以把C语言学好。你至少可以从事我刚刚说的这三个行业。那么如果你实在是不喜欢做硬件，不喜欢做游戏，也不喜欢做后端的服务，你说我喜欢做APP或者前端，那么我就要说第3个理由了。

3. 比较容易转到其他的语言。

由于c语言更偏向系统底层，可以把它看作是内功的修炼，很多语言在语法上都和c语言类似，而且要比c简单。

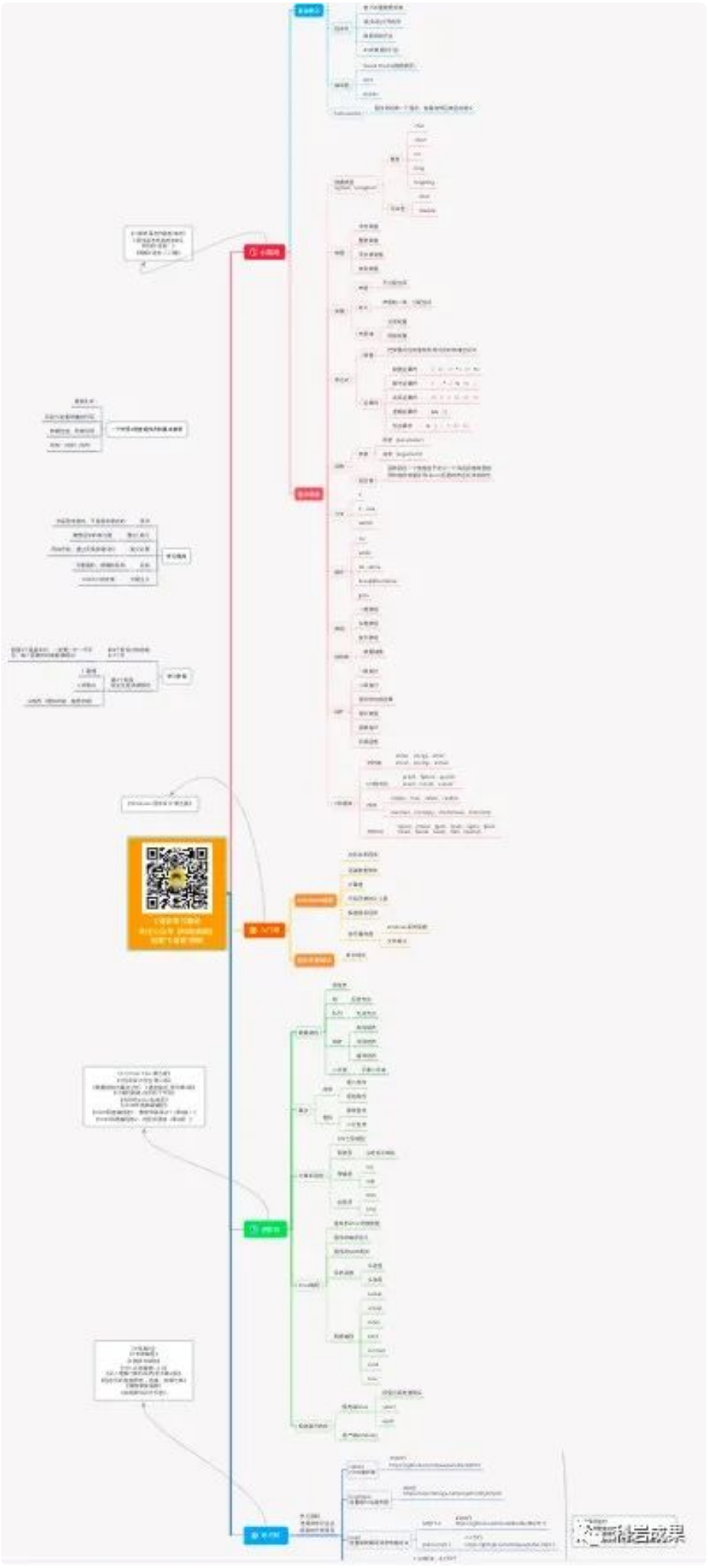
所以如果你不想做C语言相关工作，转向其他的语言会很容易，而如果你想反过来先学其他语言，再转C语言，不是不可以，只是相对就要困难一些了。

以上就是学习C语言的三个理由，不知道有没有打动你。那么接下来就是学习C语言的路径了。

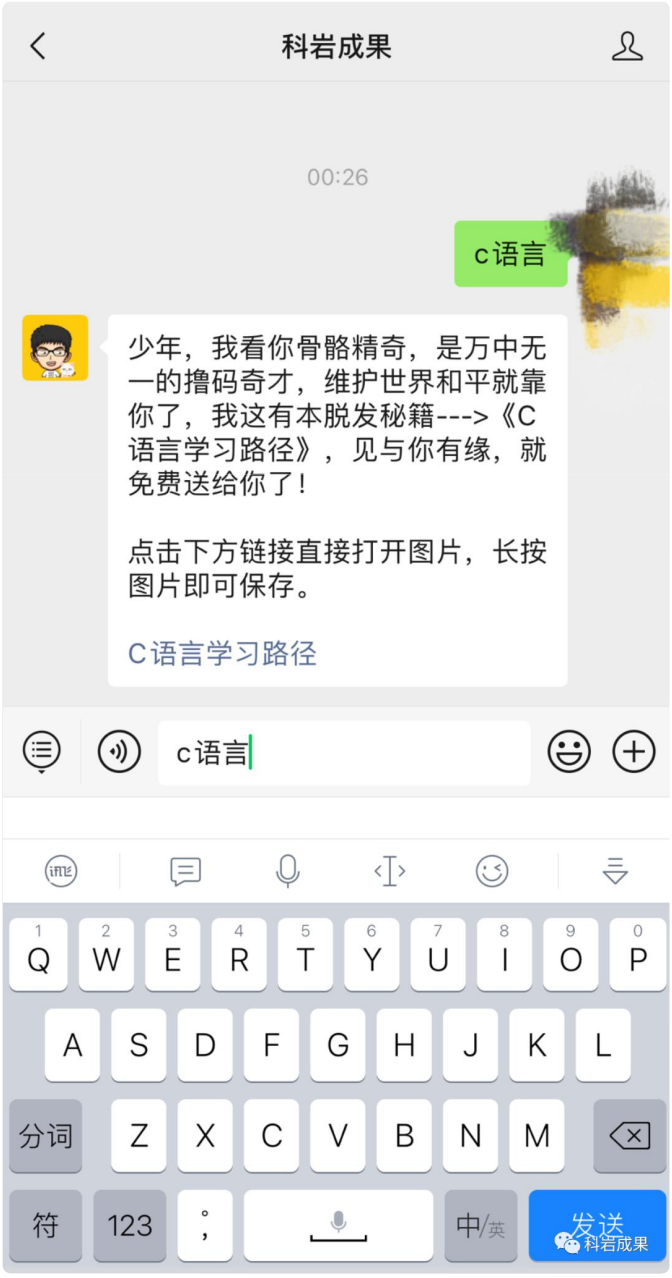
C语言学习路径

如果把学习c语言比作是攀登一座巍峨耸立的大山，那么我要说的这条路径呢，就是最快的爬到山顶的路径，我会帮你把那些路边的野花野草全部通通干掉，把那些可能导致你迷失方向的小路也都干掉，只留下最简单直接的垂直的一条直达山顶的路线，让你爬山爬的非常过瘾。

为了能够清晰的表达我想说的这个路径呢，我还画了一个脑图，看我是不是很用心。



脑图文件有点大，清晰的图片文章放不下，如果有想要下载这个脑图的小伙伴，可以关注后回复“C语言”，即可下载。



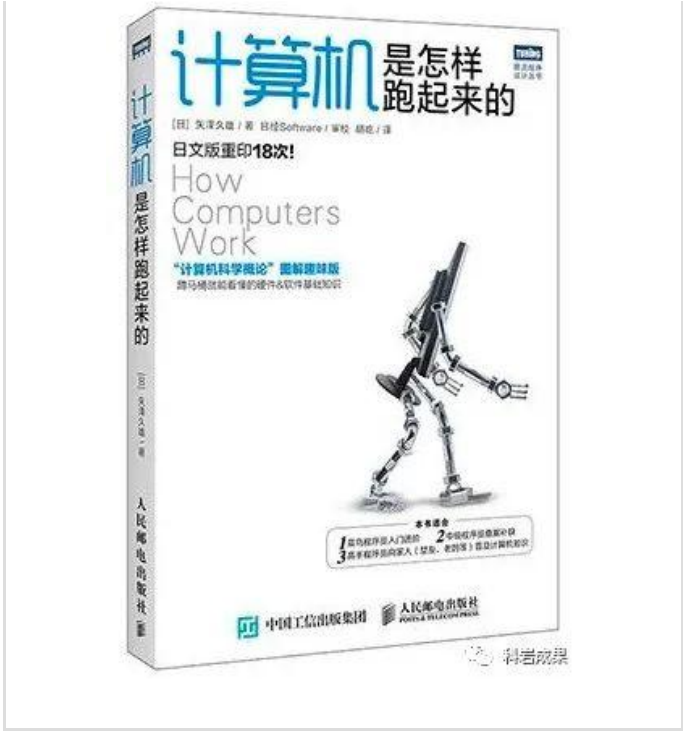
我把学习C语言的路径分成了4个阶段，**小菜鸡、入门鸡、进阶机和老司机。**

这4个阶段的学习过程中，要遵循一定的学习原则，我也总结了5个学习原则分享给大家，这样才能保证质量又保证效率。

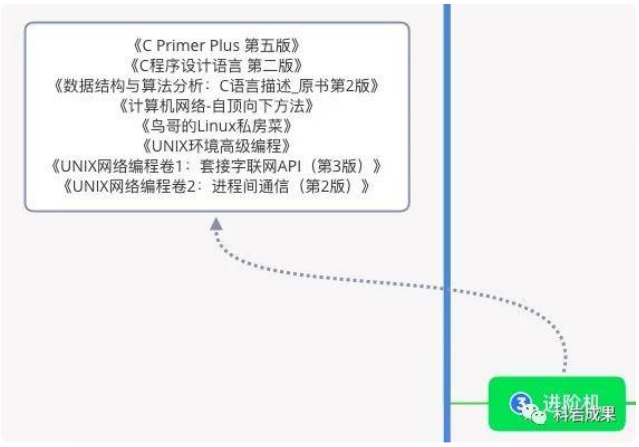
一、小菜鸡

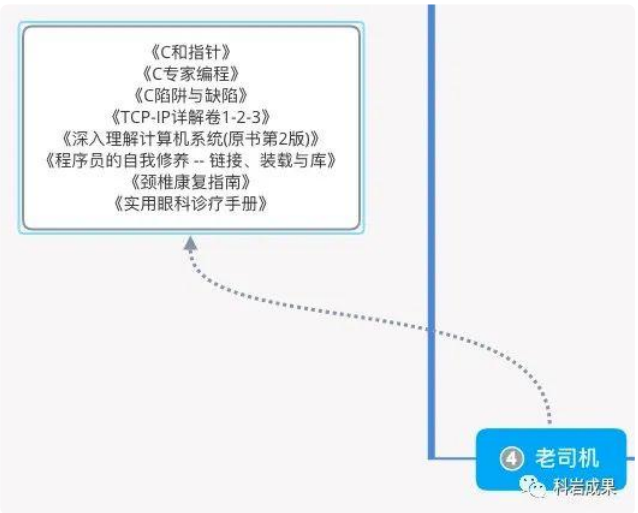
首先是小菜鸡，小菜鸡指的就是纯小白，作为一个小白呢，你要对计算机，对程序有概念。

这部分可以看看日本作家写的几本计算机知识相关的科普书，《计算机是怎样跑起来的》、《程序是怎样跑起来的》，这两本书对新手都非常友好。



说到看书呢，脑图中每个阶段也都会有对应的图书推荐，大家可以买来或者通过某种手段白嫖，我不推荐白嫖，还是买原版支持一下，作者不易啊！





看书有一个原则，就是千万不要把”读完“当做目的，计算机相关的图书很多都是大部头，等到你读完，黄花菜都凉了。所以最好是带着问题读，把它们当做一部字典，只读你关心的部分，找到答案就可以把书放下了，不要纠结读没读完。

对计算机和编程有了基本的概念后，就要学习C语言的语法了。C语言的语法其实非常简单，里面所涉及的概念也不多，都是一门编程语言中最基础的东西。


① 小菜鸡
基本语法
数据类型
signed、unsigned
整型
char
short
int
long
longlong
浮点型
float
double
常量
字符常量
整数常量
浮点数常量
枚举常量
变量
声明
不分配空间
定义
声明的一种，分配空间
作用域
全局变量
局部变量
表达式
赋值
把变量对应的值存到其对应的存储空间中
运算符
赋值运算符 = += -= *= /= %=
算术运算符 + - * / % ++ --
关系运算符 == > < >= <= !=
逻辑运算符 && ||
位运算符 & | ~ ^ << >>
函数
参数
形参（parameter）
实参（argument）
返回值
函数返回一个值相当于定义一个和返回值类型相同的临时变量并用return后面的表达式来初始化
if
科岩成果

数据类型、常量、变量、表达式、函数、分支、循环、数组、结构体、指针、C库，一共就这11个概念，对新手来说，前面的简单点，后面的可能稍微难理解点，平均每个概念学习3到4小时左右就可以了。

这些基本概念该怎么学习呢？

下面说第二个学习原则：一定要有合适的练习题。

什么是合适的练习题，合适的练习题有3个特点。

一是有明确的训练目的。比如说学习函数，那合适的练习题就会把练习重点放在函数的各个特征上，而不应该有其他的一些知识盲点。

二是不应该需要花费太长时间。这个花费的时间应该刚刚好能够使你保持专注，一旦失去专注力，浪费时间不说，还可能完不成练习。

三是合适的练习题的难度应该是能够稍稍高出你自身的水平。使你每次练习都有一点新的收获，如果每次练习都在画菱形、画正方形，画了也白画。

有了合适的练习题的训练，你就会比较快的掌握基础知识了。

我建议这个阶段快速突击，**小菜鸡的阶段不宜久留，宜速战速决**，1周时间搞定，最多2周。否则啊，学习很容易变得枯燥乏味，就会产生这玩意到底有没有用啊，这种负面情绪。

还是拿登山做举例，小菜鸡阶段就是在山脚下听老师讲理论课，老师讲了半天，登山鞋有哪些品牌，如何选择，登山杖该怎么使用，登山时姿势是什么样，全身肌肉如何发力，遇到雨雪天气怎么保暖，遇到山体滑坡怎么自救等等，一直讲这些理论。

你说我报名了登山运动练习班，光理论讲了半年，学员当然要问，会选登山鞋到底和会不会登山有什么关系？

学登山么，懂了最基础的理论知识，就赶紧是骡子是马拉出去溜溜了！先登个几百、一千米看看，把登山实践过程中遇到的问题点记录下来，再翻书本找教练学习理论。

学习编程语言也是一样，懂了最基础的之后，就可以做些与实际生活相关的小项目，遇到不会的再回头学就可以。

这里我插入第三个学习原则，叫做「最少必要知识」。

就是说刚接触一个领域，一门知识，先把最关键的那些知识学会，你就可以做些简单的东西出来了，这时就应该尽快开始实践，特别是那种需要动手的学科，没有必要纠结要全学完再开始，而且有些高级的东西必须需要时间的积累才能理解，就算你学会了考试考过了，没有时间打磨，那也是纸上谈兵啊。

所以我再强调一遍，小菜鸡阶段不宜久留，应当速战速决，1周时间搞定，最多2周。

得马上做点有成就感，有满足感的东西出来。

二、入门鸡

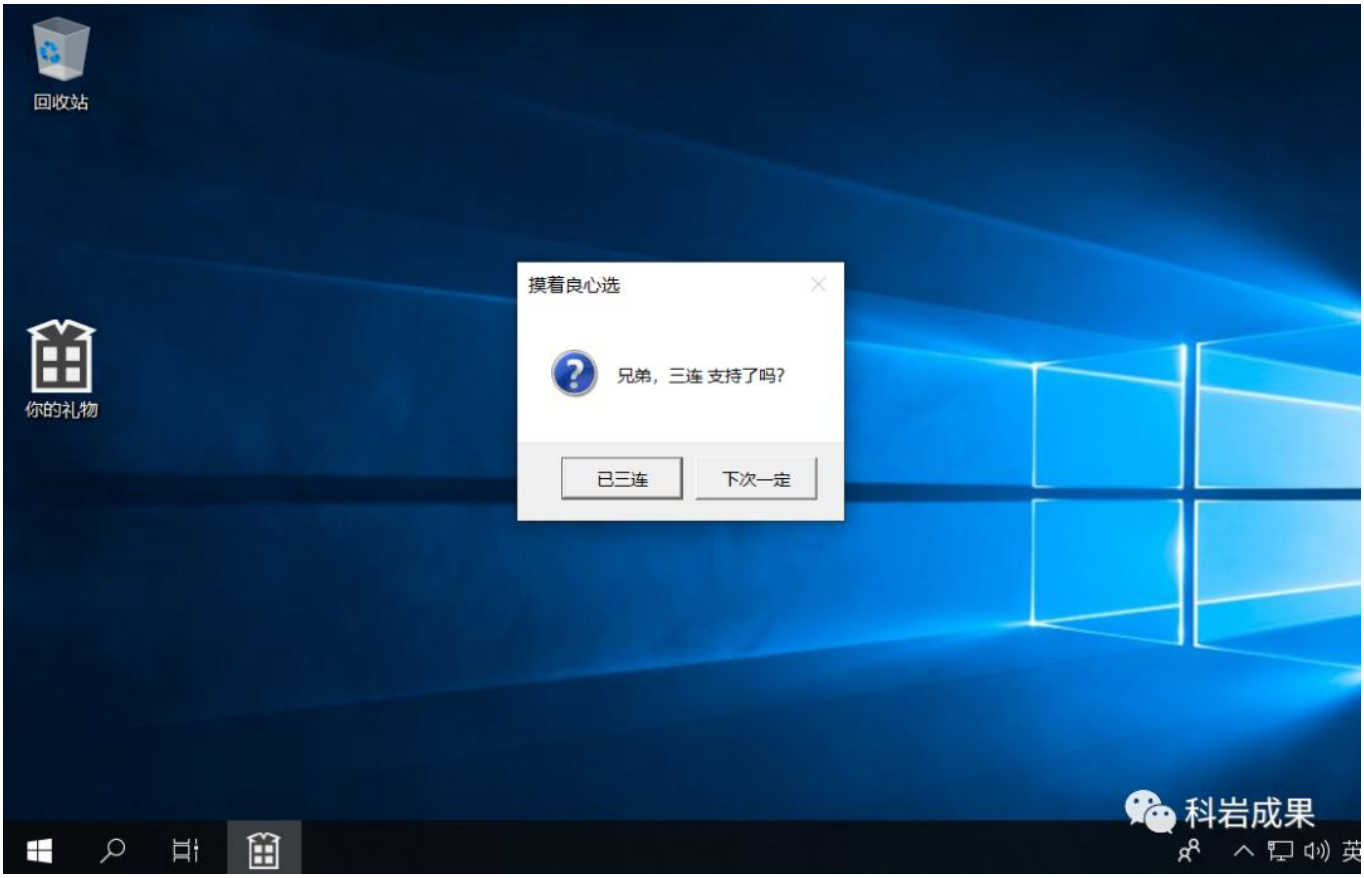
所以，赶紧进入第二个阶段，入门鸡。

这第二个阶段的目的是在对编程有了一点初步认识之后，尝试自我开发自己对编程的兴趣。

我建议用windows系统的接口做一些比较有趣、好玩的程序，比如我简单列了几个适合练手的小程序。

我为什么推荐windows呢？一是因为windows的图像界面编程是非常方便的，拖拖拽拽就能画个界面出来，所见即所得很容易形成成就感；二是因为windows特别适合小白，它把程序背后编译、链接等等在新手阶段不需要了解的知识全部隐藏掉了，只要关注你的代码就好。

可以从最简单的开始，比如说先弹出一个这样的对话框，其实也没有写几行代码。



但是这个东西就很有趣，就要比那种黑乎乎的命令行看着顺眼的多对吧，你可以把这个程序发给你的同学，如果你同学只会写杨辉三角，那你这简直就是高了一个段位。这是什么？这不就是成就感吗？

我上学那会就对图形界面很感兴趣，我到现在依然记得第一次用windows编写了一个最简单的对话框的那种喜悦，我感觉我就是计算机的上帝，我让他干嘛他就干嘛，这才叫编程嘛。

我用C语言写过一个音乐播放器，还用java写过一个扑克牌游戏，当时真的是，课程设计都直接高分。

图形界面编程很容易激发一个人的兴趣，俗话说，兴趣是最好的老师，一旦你学习编程的兴趣和热情被点燃，那这兴趣会驱动着你继续学习下去，你会不断完善你所编写的代码，直到它展现出你想要的样子。

在这些编程练习中，你会反复用到第一个阶段学到的理论知识，遇到不会的，翻翻书，找找视频马上补上，这样的练习有了理论指导会事半功倍。

这个阶段的时间长短呢，自己决定，如果越做越爽，也可以一直在这个阶段玩下去。

这里我再插入第四个学习原则，就是要自己给自己订立目标。不能漫无目的的瞎学，要有个指引，订立的目标最好是可以衡量的，不然你都不知道自己学没学会。

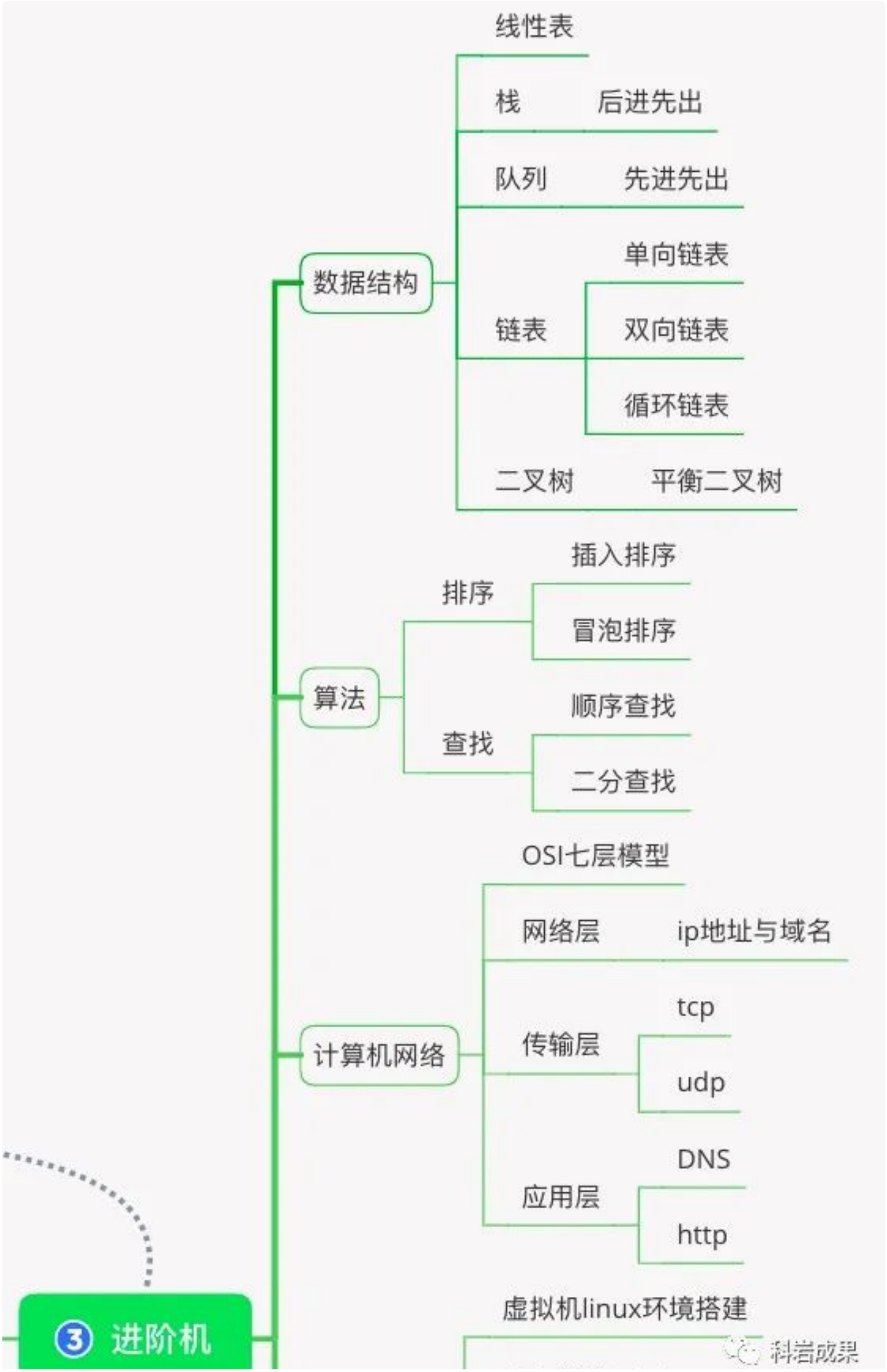
比如立下目标，要在3小时内学习5个C标准库的接口。能够知道在什么情况下用、并且会用5个接口就算达成了目标。

再比如做个音乐播放器，要把一个音乐播放器做到什么样的效果，有哪些功能，先从整体上考虑好，然后为了实现这些目标不断完善，一旦达到了目标，就可以进入下一个目标了。

三、进阶机

那么下面是第三个阶段，进阶机。

这个阶段要学一些编程语言之外的东西。数据结构、算法、网络的理论和实践都该安排上了。


③ 进阶机
数据结构
线性表
栈
后进先出
队列
先进先出
链表
单向链表
双向链表
循环链表
二叉树
平衡二叉树
算法
排序
插入排序
冒泡排序
查找
顺序查找
二分查找
计算机网络
OSI七层模型
网络层
ip地址与域名
传输层
tcp
udp
应用层
DNS
http
虚拟机linux环境搭建
科岩成果

数据结构可以先学些最基础的，像栈啊、队列啊、链表，他们的概念是什么，有什么区别和联系，树形结构比较难理解，看不懂可以先略过。

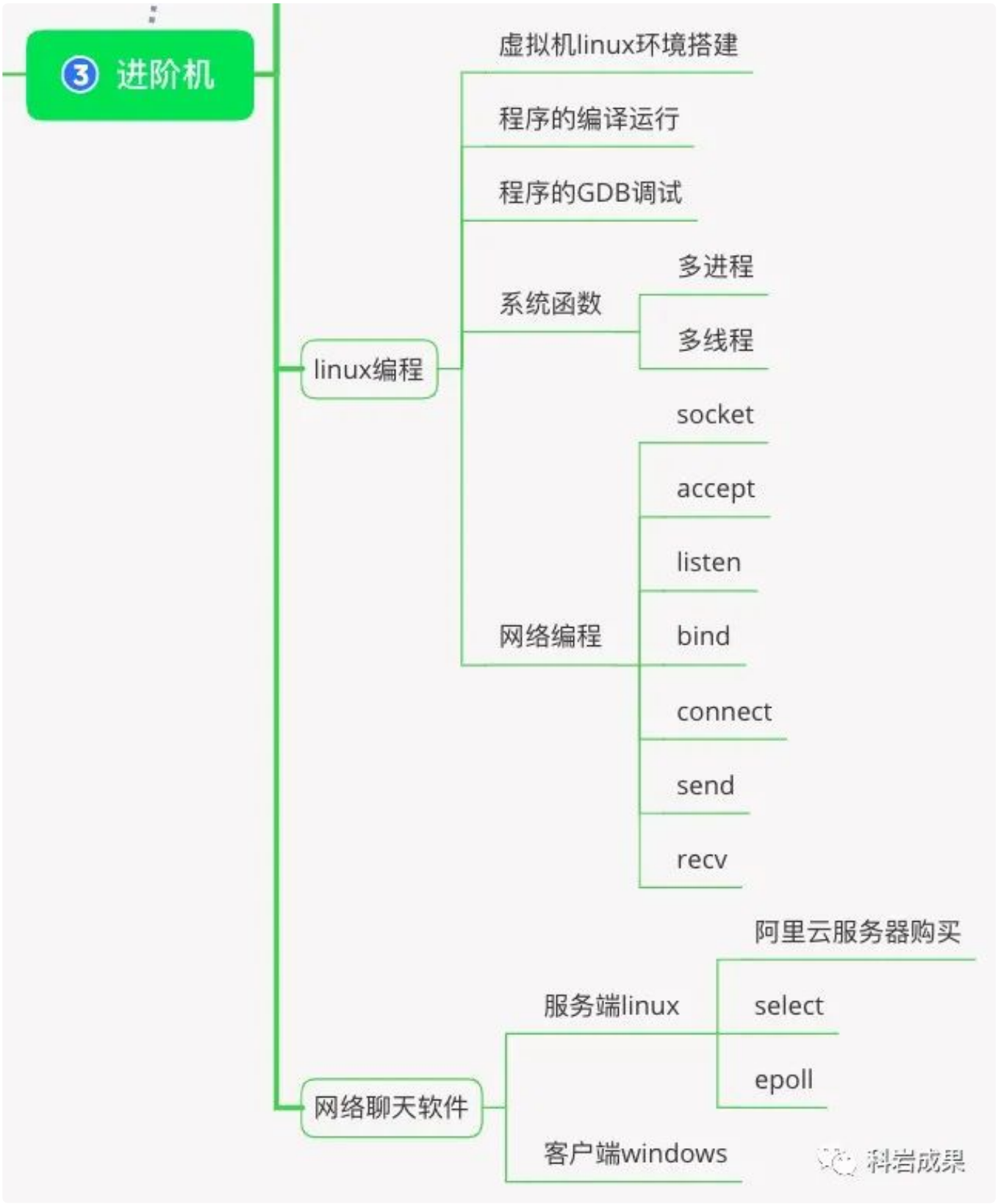
简单的算法也要懂一些，排序和查找是最常用的，必须要掌握。

计算机网络是整个互联网行业网络通信的理论基础，也必须学，物理层、链路层可以不会，网络层、传输层和应用层必须掌握最关键的几个协议。

最后，学习了这些理论知识还是要实践，没有实践的理论就像是盖在沙子上的碉堡，俗称沙雕。如果你不想做沙雕，那就把代码写起来！

这个阶段该学习在linux上写代码了，毕竟在实际的工作中，基本没有人在windows上用C语言开发程序。

我觉得非常合适的一个练习项目就是网络聊天软件，像QQ一样的，可以用linux写服务端，用windows写客户端，这个过程中的练习会加深对操作系统的系统调用、数据结构、网络这些知识的理解。



而且网络聊天软件很有趣，做起来也没那么枯燥，做好了之后发给你女神一个专属客户端，你们在你自己写的软件里聊天，边聊天边吹牛，岂不美哉！

这个阶段涉及不同课程的知识，应该花费比较长的时间。我建议代码要一行一行的写，就像饭要一口一口的吃，三连支持要边看边点。

项目中每一行代码都要搞明白什么意思，这样你才能写出健壮的、没有bug的代码。

我在写这个稿子的时候专门去找了一些开源的聊天软件，没有找到我想要的C语言的windows+linux这样组合的，并且适合新手的项目。

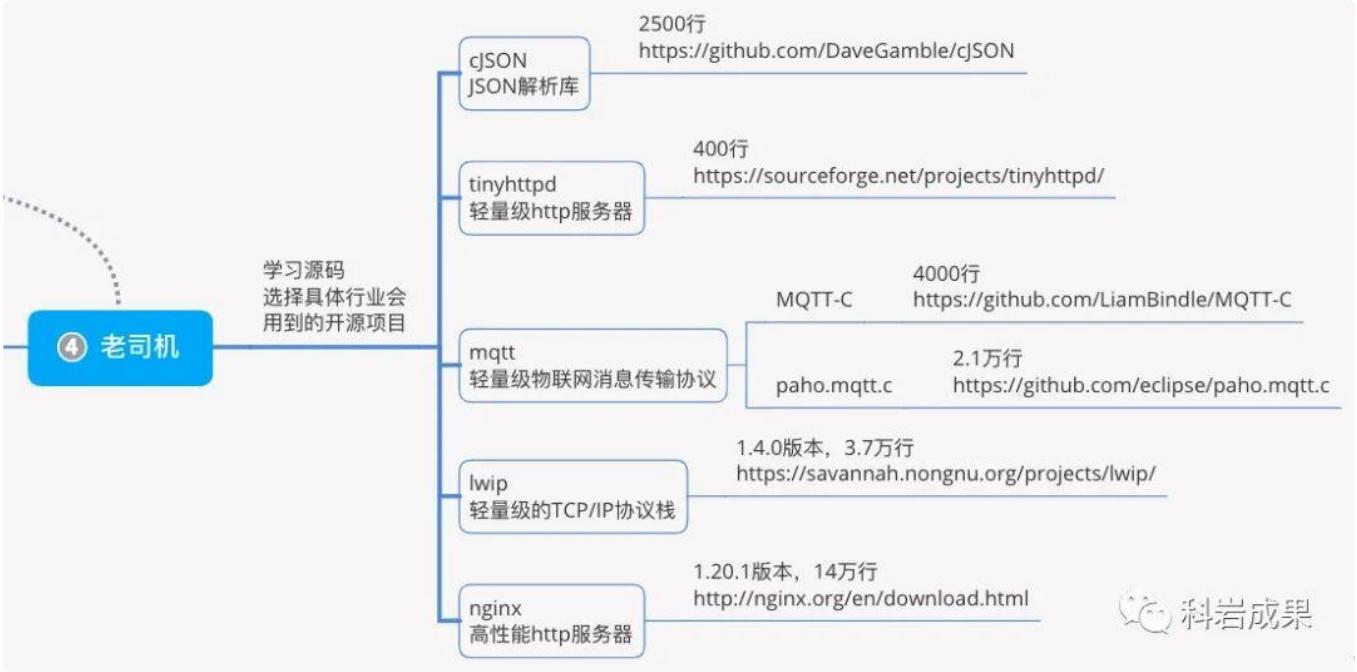
不知道小伙伴们对这个学习路径是否认可，对这样的项目是否有兴趣，我就大言不惭的在此立下一个flag，如果这篇文章的点赞数超过100，我就实现一套适合自学的代码，分享给大家。

四、老司机

那么接下来呢，就来到了C语言学习的最后一个阶段：老司机阶段。

这个阶段不再需要学习基础知识，而是要开始向高手、向大师学习了。

使用广泛、经得起时间考验的开源代码是这个阶段最好的学习资料，我列出了几个网络行业相关的C语言的开源项目，这也是我在工作中经常接触到的项目。

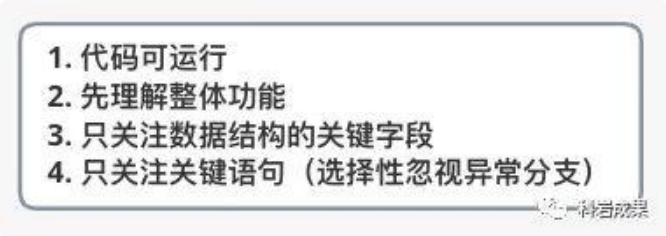


有JSON格式数据的解析库cjson、最小的http服务器tinyhttpd、高性能的http服务器nginx，还有物联网常用的传输协议mqtt，如果想学习tcp与udp的实现，有个协议栈找叫lwip，与linux内核的网络协议栈相比，代码量少了不是一星半点，非常适合学习。

我不知道你是什么行业，你所在的行业又有哪些著名的开源C项目，欢迎留言分享给大家！我相信它们都可以拿来做自学的材料。

刚开始学习开源代码肯定是一脸懵B的，你会惊呼，C语言还能这样写？你会怀疑，我眼前的代码还是那个熟悉的C语言么。

相信我，每个人都是这样的，我总结了几个学习源码的方法也分享给大家，这个脑图后面也有：



首先要保证代码是没有错误可以直接运行的，不然一运行就出错，就没有办法继续下去；

接着我们要先从整体上理解整个项目的功能是什么，它的输入是什么，输出是什么，先在头脑里有个预期；

在看代码的时候可以只关注数据结构中最关键的字段和最关键的语句，与主干逻辑无关的代码可以直接忽视掉。**

进阶机阶段是学习的是怎么使用接口，老司机阶段就是学习怎么实现接口了。能看明白开源代码，能讲明白实现原理，你就踏入了专家之路。

end

好的，以上就是关于C语言的学习路径了，总的来说，C语言简单直接，非常容易上手，但想做到精通就没那么容易了。

不知道大家有没有听说一个「一万小时定律」，是说只要你在一个领域投入超过10000小时的练习时间，你就会成为那个领域的专家。

这就是我要说的最后一个学习原则，长期主义。

其实10000个小时也没有多遥远，如果一天工作8小时，除去节假日，一年的工作时间有2000个小时，所以如果你一直在学习和使用C语言，并按照这个路径和刚刚说的那些学习原则坚持下去，**5年就会成为编程高手！**

不知道你听了有什么感觉呢？欢迎留言说出你的故事。

我把这篇文章用到的脑图图片放在了公众号，关注后回复「C语言」，就能下载了，如果你有需要，可以关注下载。

我是科岩，下期再见。

----------END----------

文章到这里就结束了。

欢迎扫描下方二维码关注「科岩成果」~~~O(∩_∩)O~

你好，我是科岩。我做嵌入式工程师9年多了，每当调通一块板子时，都会有满满的成就感。
我将持续分享个人的成长收获，帮助你提升技术能力与认知视野。另外我还听说，关注我的人个个都是人才，一起加油吧，人才们：）

科岩成果
科岩：嵌入式工程师；成果：喵~
16篇原创内容

公众号