Level0-1之前：各类英文缩写的含义不甚明了。一个只用过DevC++的编程小白，从来没听说过gcc,sln之类的缩写。

经上网查阅：

gcc是一个编译器的名称，支持多种语言。gcc的下载是件麻烦事，官网改版，之前的教程都不适用了，好在最后自己不知怎么找到了一个installer，跟随教程完成了gcc的配置

VS指Visual Studio是微软开发的集成开发环境IDE，要付费。

IDE指集成[开发环境](https://baike.baidu.com/item/%E5%BC%80%E5%8F%91%E7%8E%AF%E5%A2%83/10119007?fromModule=lemma_inlink)，是用于提供程序开发环境的[应用程序](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F/5985445?fromModule=lemma_inlink)，一般包括[代码](https://baike.baidu.com/item/%E4%BB%A3%E7%A0%81/86048?fromModule=lemma_inlink)编辑器、[编译器](https://baike.baidu.com/item/%E7%BC%96%E8%AF%91%E5%99%A8/8853067?fromModule=lemma_inlink)、[调试器](https://baike.baidu.com/item/%E8%B0%83%E8%AF%95%E5%99%A8/3351943?fromModule=lemma_inlink)和[图形用户界面](https://baike.baidu.com/item/%E5%9B%BE%E5%BD%A2%E7%94%A8%E6%88%B7%E7%95%8C%E9%9D%A2/3352324?fromModule=lemma_inlink)等工具。集成了代码编写功能、分析功能、编译功能、调试功能等一体化的开发软件服务套。VS code不是一个IDE。

sln是 Visual Studio 的解决方案文件，用于组织和管理一个或多个项目。它可以包含一个或多个项目，以及这些项目之间的关系和配置信息。一个 .sln 文件本身并不包含任何代码或文件，它只是包含引用到该解决方案中所有项目的信息和设置。

Level0-1 ：明明没有科学上网，但是依然非常流畅地打开了GitHub。我最终采用HTTPS形式分享我的远程仓库。

Level0-2 ：我通过自行阅读C Primer Plus学习C语言。群里有人说过看到指针即可，于是我一直读到了第十章 数组与指针

C Primer Plus对于编译过程介绍的并不多。通过上网查询，得知从源代码到可执行文件主要有四步：

预处理：预处理器处理预处理指令

编译：将预处理后的源代码（高级语言）转换为汇编代码（低级语言）

汇编：转换为机器可读的0101

链接：链接器将汇编生成的目标文件以及可能的库文件链接在一起，生成最终的可执行文件

在命令行窗口中输入：

gcc -E hello.c -o a.c //-E表示仅预处理，-o指定了输出文件名

gcc -S a.c -o a.s //-S表示仅编译

gcc -c a.s -o a.o //-c表示仅汇编

gcc a.o -o a.exe

输入a.exe运行程序，显示hello,world

或者直接gcc -o b.exe hello.c一步完成

Level1-1 比较简单

Level1-2 比较简单（wait）

Level1-3 首先我通过文件Level1-3.1完成了第一个要求，即串联前两个功能并允许重复购买（输入q以结束），并将购买过程与投币找零过程封装为两个函数。由于是重复购买，我增加了对购买金额超过最大可购买金额（存货不足）的检测。

然后我通过文件Level1-3.2完成了第二个要求，由输入总价改为输入购买的货物。由于要支持重复购买，我仍然保留了摆放结果展示，并将其封装为函数，毕竟得让客户知道售货机里还有什么。我增加了对误操作的检测，包括货物摆放发生误操作，购买货物时输入格式错误、与实际摆放不符或存货不足，并在投币中检测是否为1、2、5

由于此时仍为1个通道，难度仍然不算大。

Level2-1 这个程序我写了很久，更新了好几个版本。首先，我为通道，货物，数量，价格分别创建了一个数组以支持5个通道，随后define了一个常量为5。我仍然保留了摆放结果展示，并且使用嵌套循环使其按照通道序号顺序显示。我在购买结束后询问了是否继续。

我将结束标志从q改为了更不常用的#来标志结束（因为scanf判断字符更方便），但是后来看到群里出题同学要求使用END，我一筹莫展，在这里卡了很久，最终也未能在level2-1中实现（此时我还不知道有strcmp()函数）。

另一个难点是对输入行剩余内容的处理。我起初秉承着与其之后找bug，不如多写点while（（ch=getchar()）!='\n'）的理念，在程序中写了很多while（（ch=getchar()）!='\n'）。但是运行的时候发现完全乱套。思考了很久，后来意识到如果输入行为空，该代码会等待用户的输入，然后之后所有的输入就乱套了。

遗憾的是缺少同货物不同价格的检查机制、通道数重复检查机制，这些我在最终版本才实现。

另一个遗憾是暂时使用了goto来跳出多层循环，我也是在最终版本才删掉了这个goto。

Level2-2 由于此时我还不知道有strcmp()函数，想了很久，最后想出了一个投机取巧的办法，当scanf（）得到货物种类为E并且返回值不为4或3，即当作输入了END。BACK同理。就这样勉勉强强地写了一个程序出来。

但是我总感觉这样不是很好，于是我回顾群里的聊天记录，看到字符串、结构必用。我大吃一惊，字符串还稍微了解一点，结构又是什么。

于是我快速地回顾了前十章，然后快速浏览了后面每一章的关键概念和本章小结，又发现了很多可以用到程序里的东西。这使我想起那句著名的台词：我需要知道所有事情，不然我怎么知道我需不需要知道这件事？

看来，一定要对书本有一个整体的把握，大致了解每一章的内容，至少在脑中声明一下它存在，等用到的时候再赋值。

于是，我非常快速并有选择性地阅读了第十一章，第十三章，第十四章的部分，并开始着手重写level2-2。

我使用了结构数组，每一个元素包含了一个通道的各种参数。

但是仍然遇到了问题。strcmp（）比较的是字符串，这就意味着我必须将输入保存为字符串。但是，保存为字符串后，又如何为结构体赋值呢？

如果先使用scanf（），将输入格式化读取，又如何再使用strcmp（）呢？

两个函数竞争使用输入，于是我想到用一个缓存数组temp存储输入。但是又回到了刚才的问题：保存为字符串后，又如何为结构体赋值呢？

基于之前的经验，既然我有这个需求，前人一定为我准备好了函数。已经有了scanf（）、fscanf（），为什么不能再有一个别的scanf呢？于是我上网搜索：如何从字符串格式化读取输入，找到了sscanf（）

关键点打通了，代码完成只是时间问题。我使用自定义函数将内容读入temp数组；通过利用toupper（）使程序更友好；检测完整的back和end；优化掉了goto；增加了误操作检测，如再同物不同价时提醒用户，拒绝在同一个通道上摆放不同的货物。

但是我觉得应当允许用户在完成一个通道的货物摆放后，继续向该通道摆放相同货物，但这使回退操作非常棘手。我暂时没有想到解决办法，只能提示用户继续摆放不计为一次完整动作。

Level1-2 然后，我回过头来，尝试为Level1-2实现文件输入。实现了对非法投币金额的检测，如果文档中的投币不够，切换为手动投币。