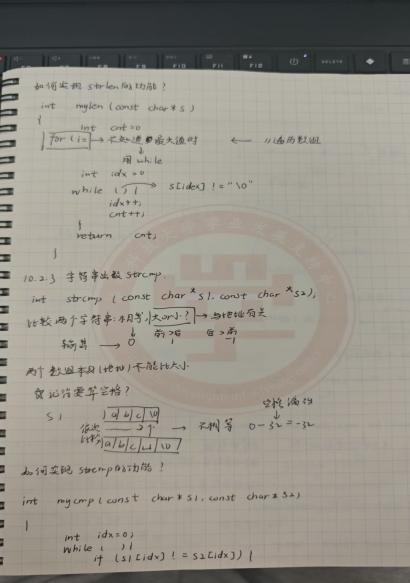
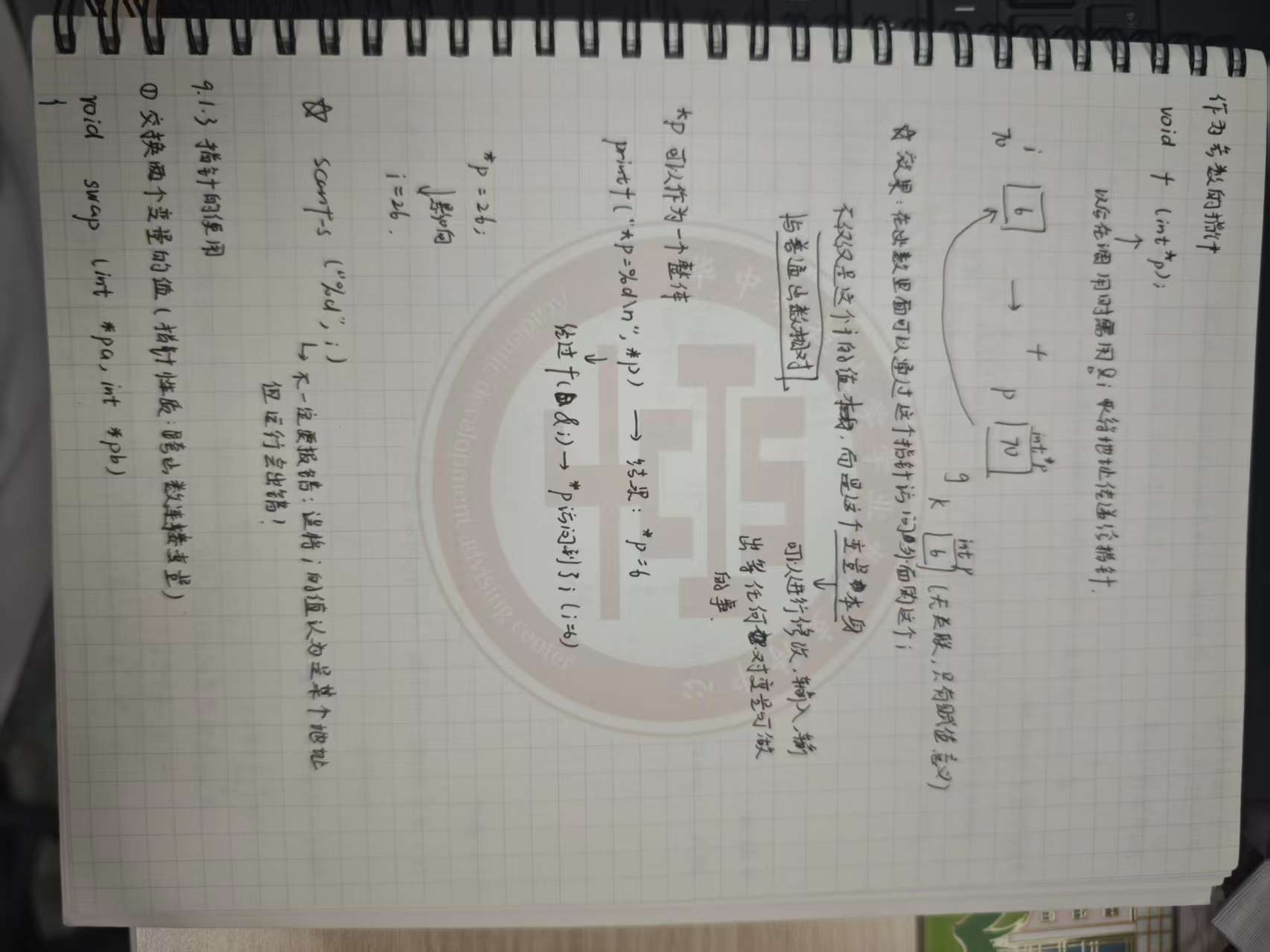
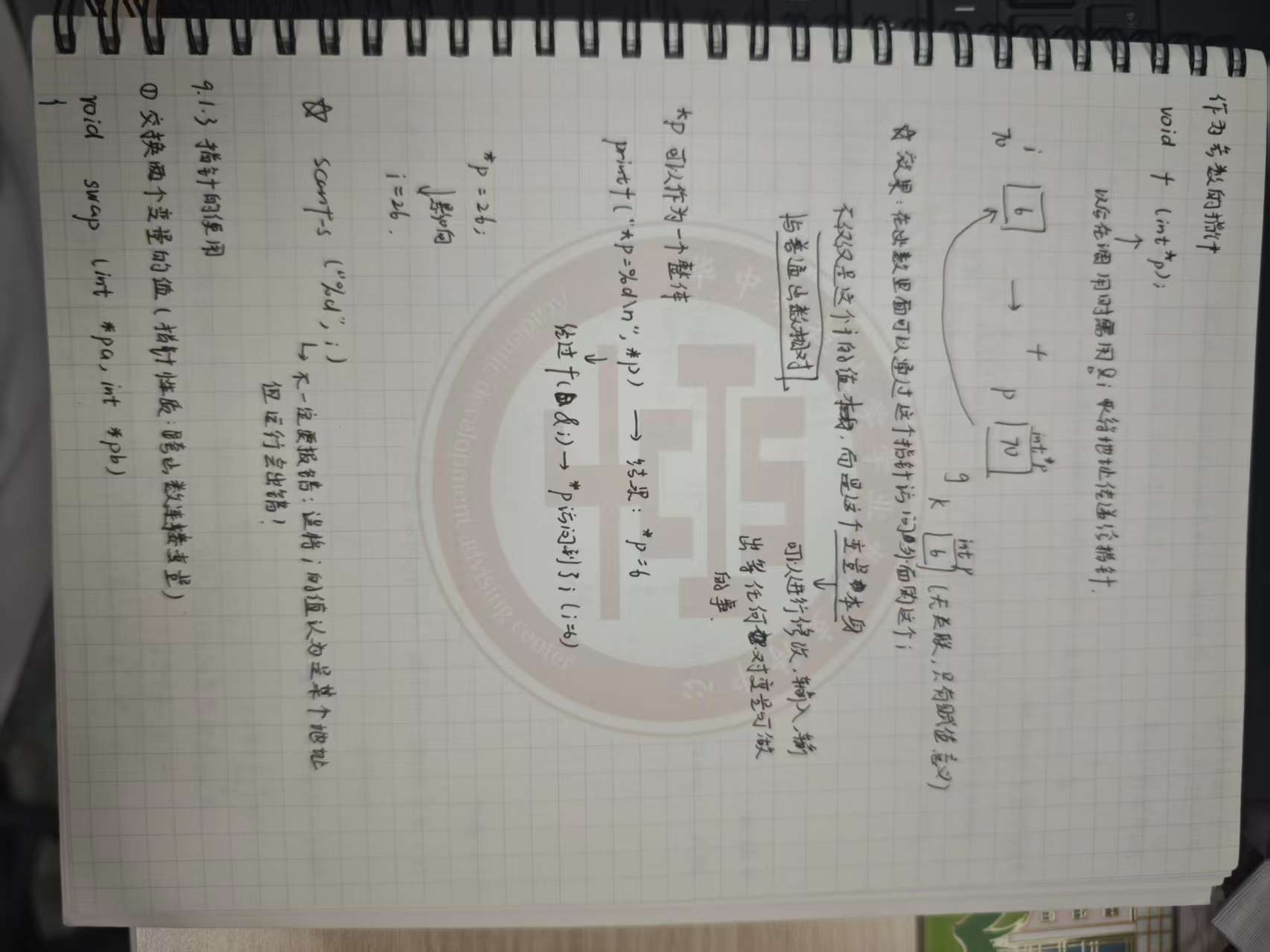
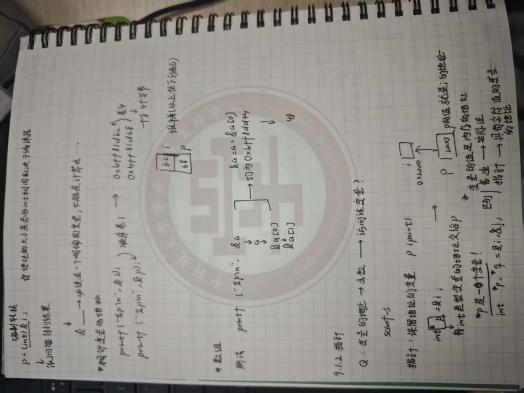
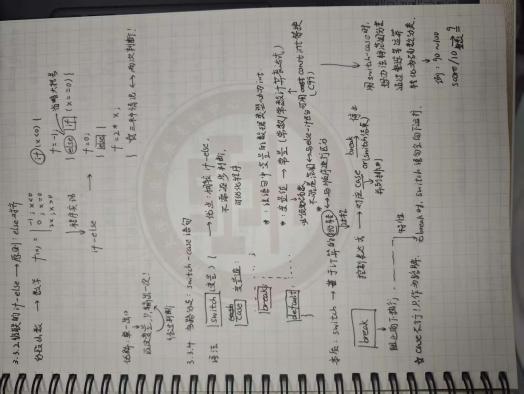
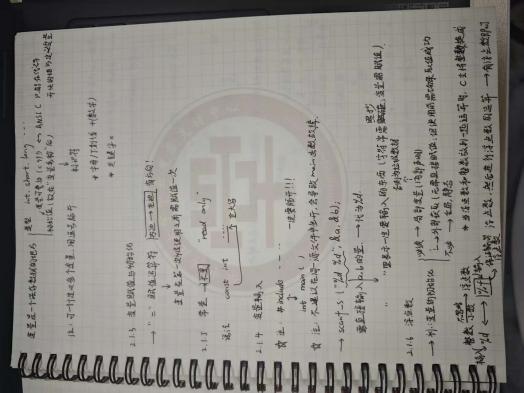
学习文档

U202513732 电子信息与通信学院 王博宇

9.1-9.20知识准备

了解到dian团队，并开始据b站翁恺c语言课程零基础开始学习c语言，并且记录了一些笔记如下（部分）：



总计50+页。

当然，光有笔记是不够的，我还在听课与实践写代码的过程中总结了部分经验和教训。

例如常见的错误：

1.scanf\_s(vs2022中不能用scanf）后的整型变量未用&来读取地址；

2.;未加带来的错误往往在下一行体现；

3.一定要把运行窗口关闭后（调试，运行过程完全结束）再修改代码，否则文件会无法打开，只能重写程序；

4.不能在同一源文件多开，否则会导致main函数故障；

5.scanf\_s中一般只写需要读入的变量，不要把不相干的文字输入进去（和printf区分）；

6.在做嵌套循环中，一定要关注变量的值的变化，不能随便拿来用，如果需要一个变量的初始值，需要提前用另一个变量储存起来；

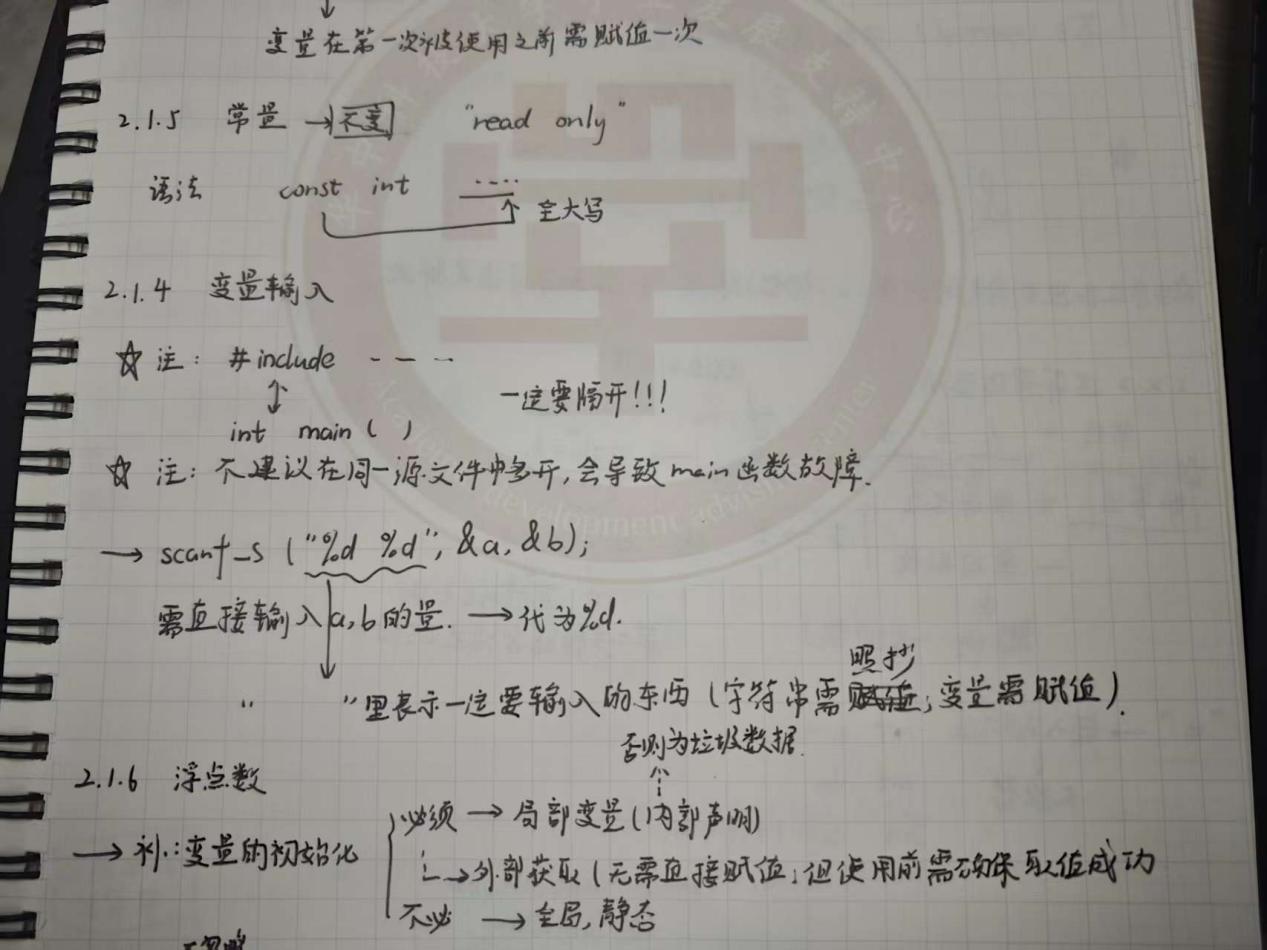
7.写条件时，=和==不能混淆；

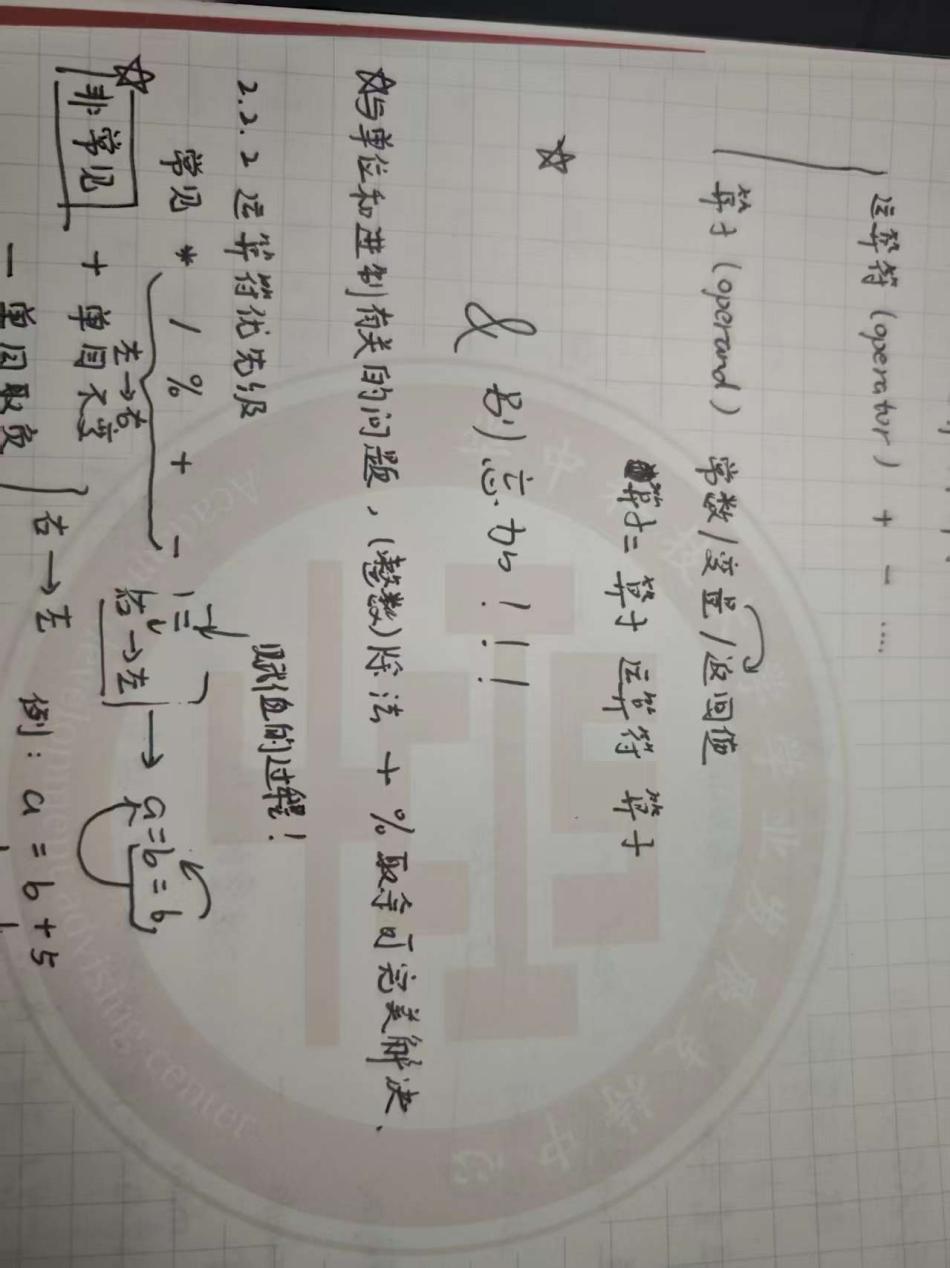
8.用变量的范围来作为条件时，不能写成1<x<5（因为比较运算不是同时进行，而是依次进行），而要写成1<x&&x<5

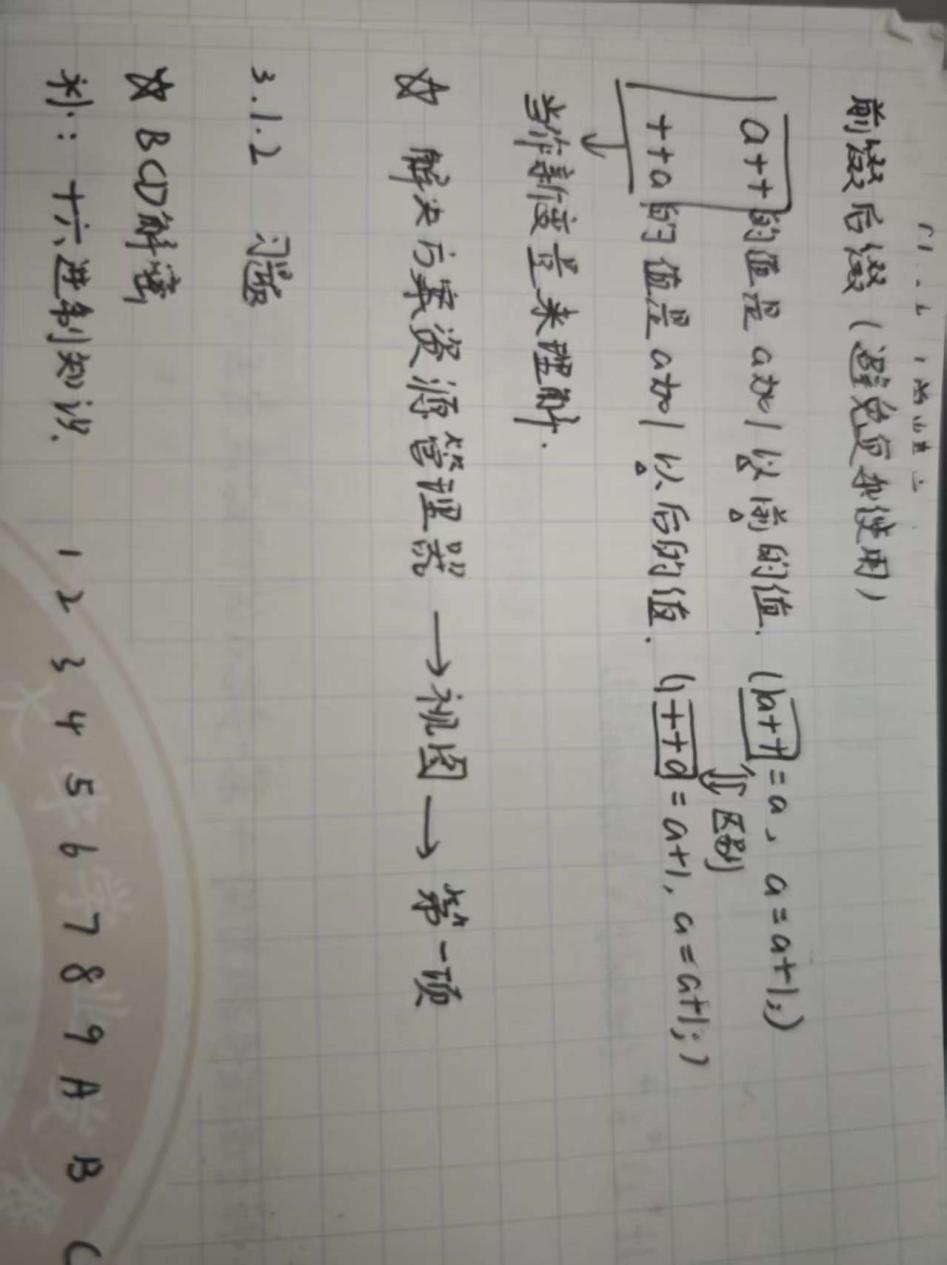
......

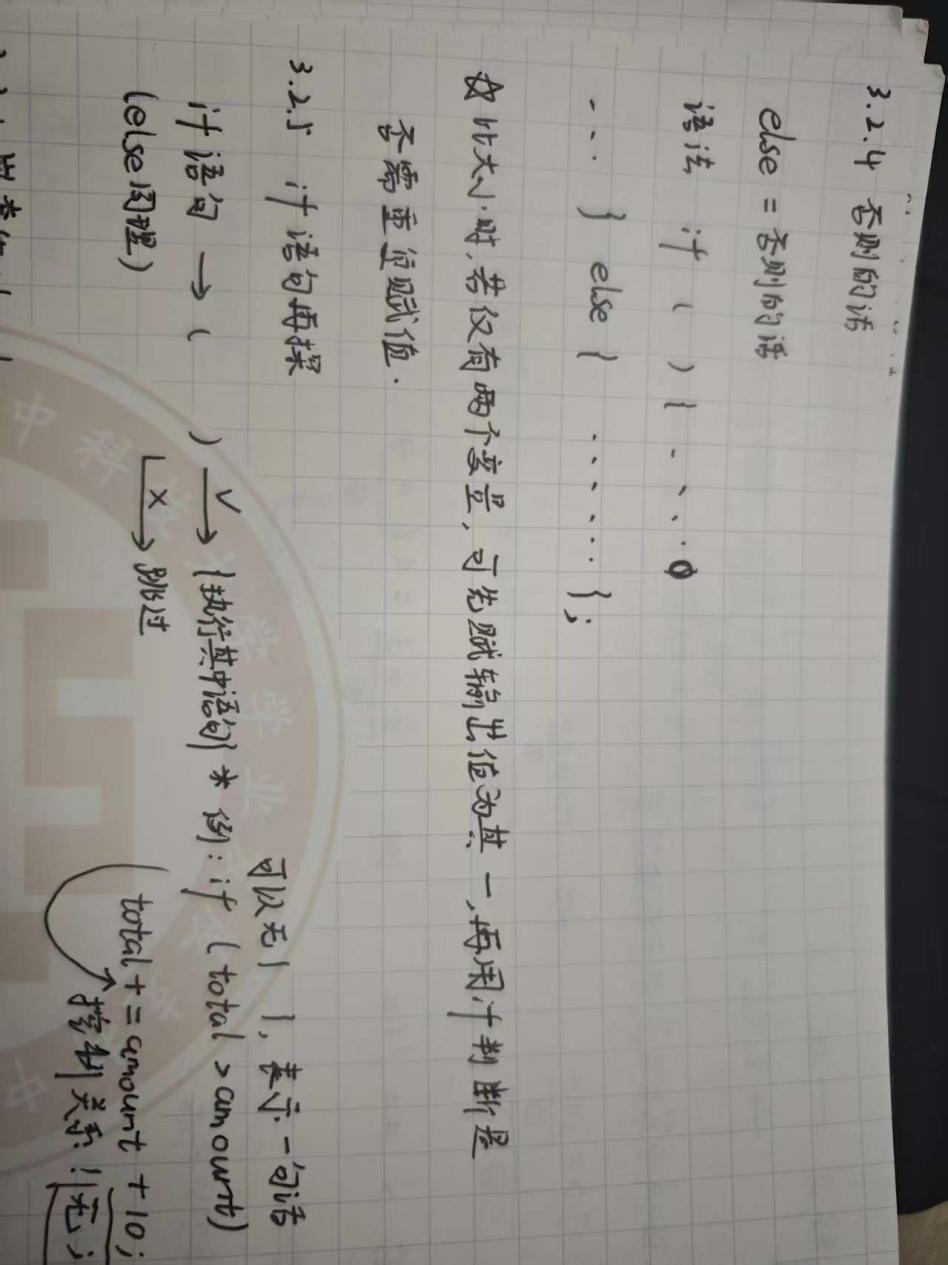
这些总结的错误虽然看上去很简单，但是往往背后是无数个运行失败，删除并重开新文件，输入输出异常，有时照抄视频里的代码也会出错（和代码在不同编译器和更新有关）但是每当练习题在我到处询问之后终于输出正确的时候，我依旧会重拾对这次学习的热情；

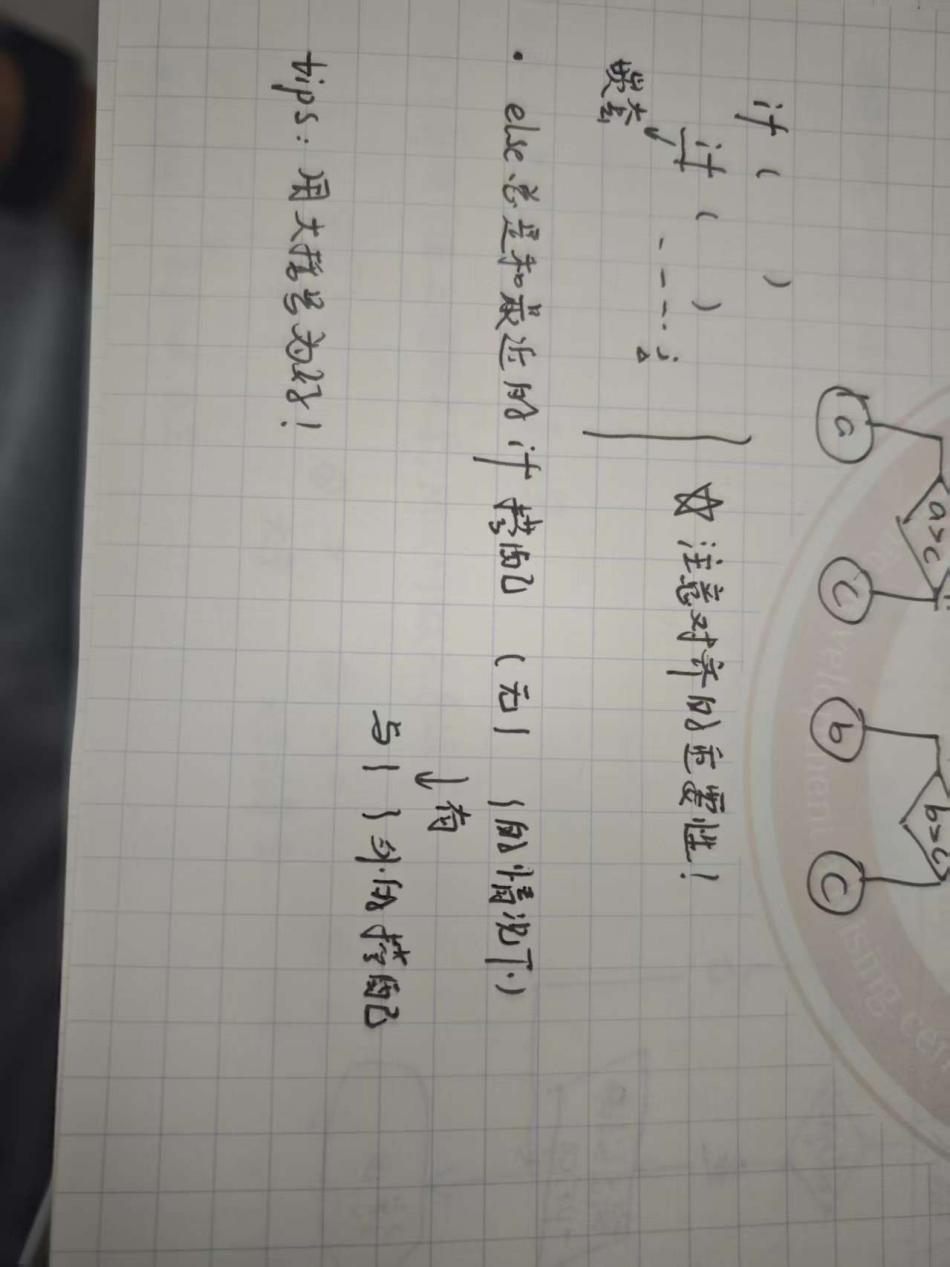
以下是部分练习时恍然大悟后总结的来源：

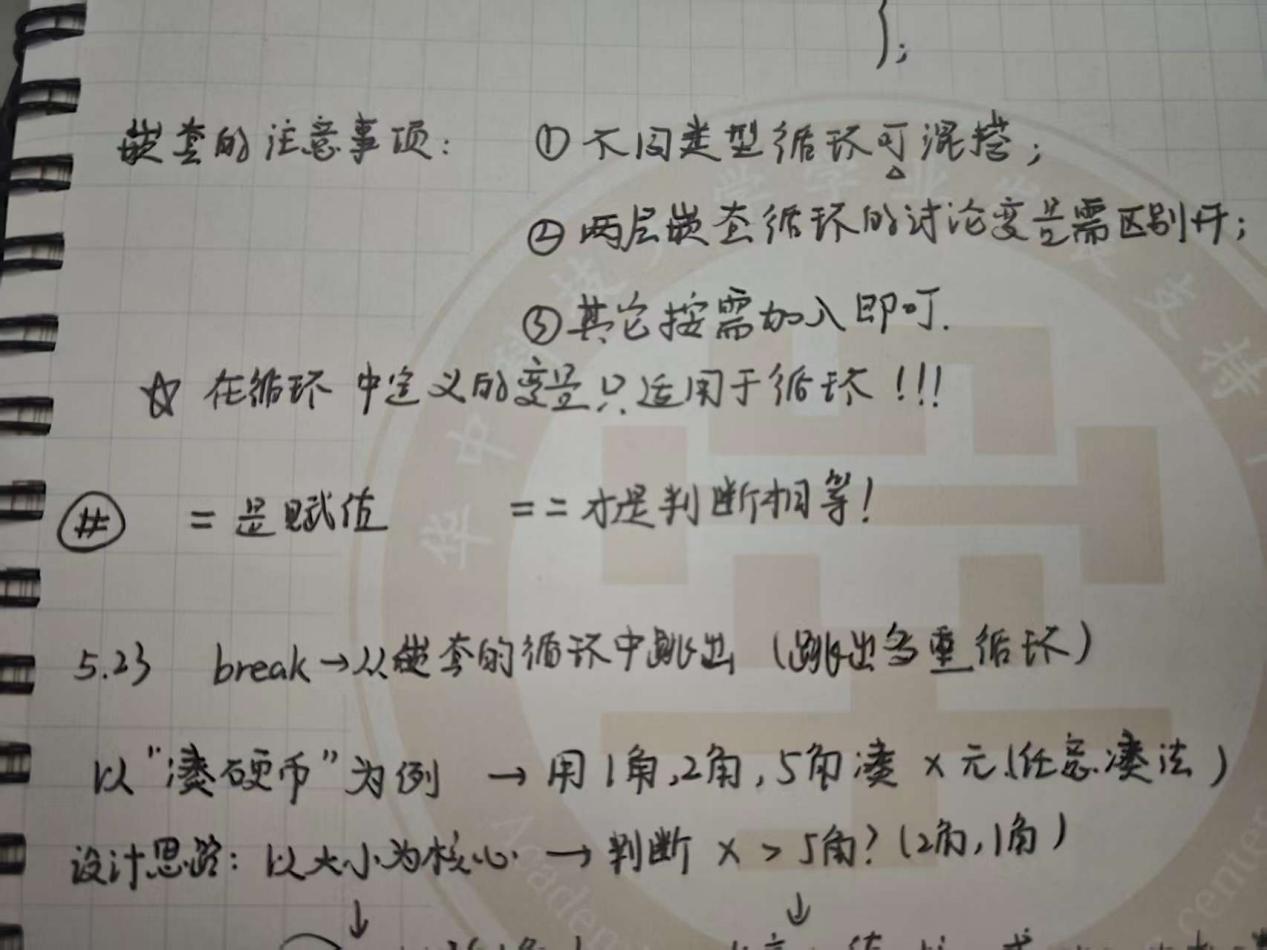


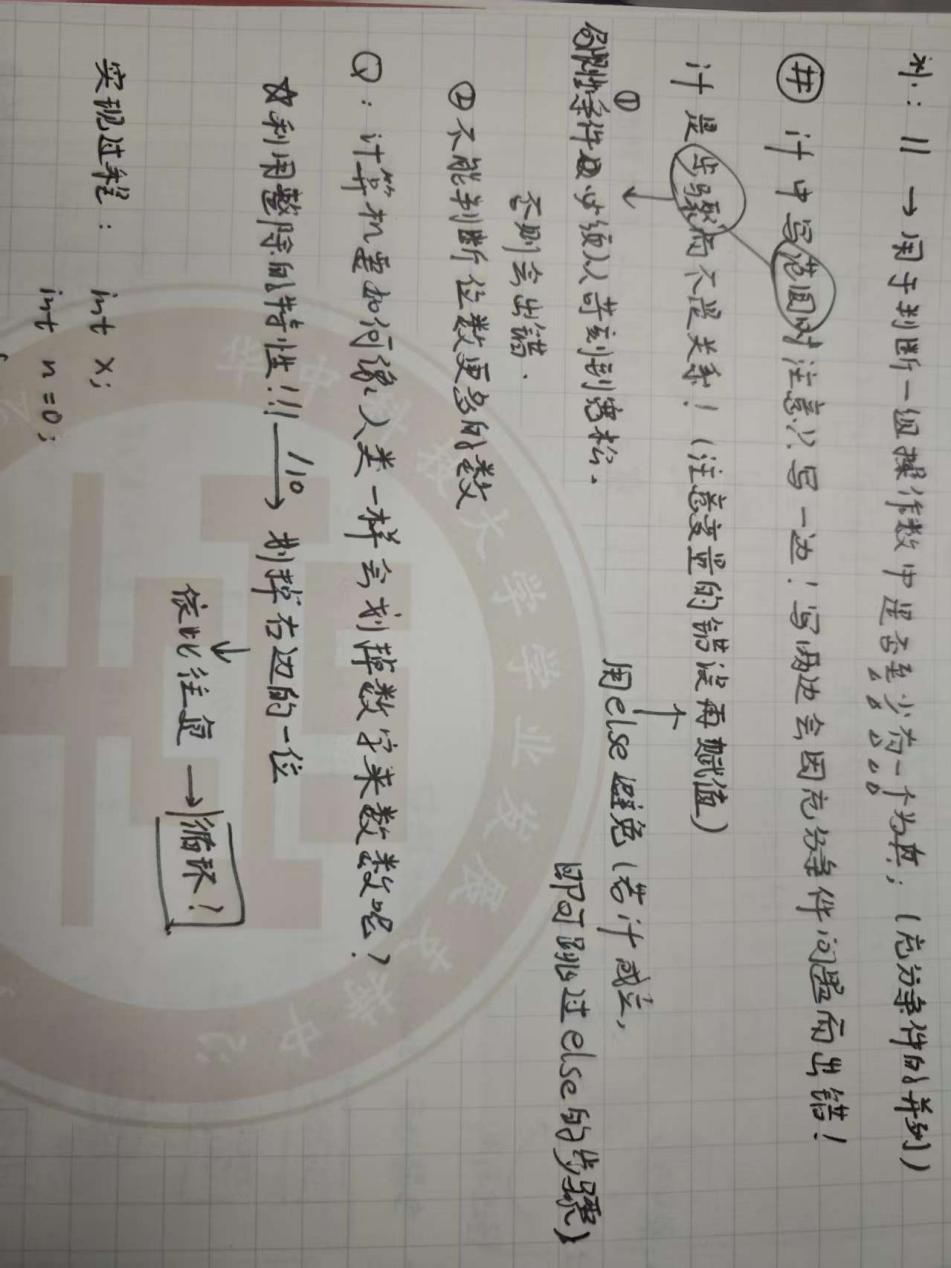












9.20-9.26（下午）level 0-1

20号看了招生题，尝试写一写，发现涉及的字符串相关知识还没学（这部分知识相关的视频在很后面）。原本想直接跳到字符串，但发现其中涉及的数组，指针也没学，于是加快节奏补完了这部分内容，虽然学的比较快而且这部分内容比较复杂，我还是简单掌握了字符串的本质，输入，输出，比较等相关基本知识。

下面是我在完成该任务过程中的一些卡点和解决过程：

1. 对字符串只知皮毛，用char x=’Dian’来定义字符串，然后又用char \*x={}来定义一个用来构造的字符串（输入量）

系统性的学习了相关知识，弄清楚了字符串的本质是数组（指针），也学会分辨了char a[](将字符拷贝到a这个数组中，可以对其进行修改，用于构造字符串），char \*a(只是读取到对应字符串的地址，不能对其进行修改，用于处理比较问题等）；

1. scanf\_s输入字符串时出现问题

通过查找资料，发现我使用的编译器和老师使用的存在差异，vs2022中需要指定缓冲区的大小（不是很明白这是什么，但知道它会导致程序故障，写成scanf\_s(“%99s”,x,(unsigned)sizeof(x))（指向该变量的纯二进制含义，但用在此处的原理我不是很清楚）后规避了这个问题；

1. 在对字符串是否相等的判断时，仅根据以往经验，用==来判断，导致始终判断失败

意识到字符串的特殊性，继续学习字符串函数的知识，了解到专门用来比较字符串之间大小关系的函数strcmp(),并弄清楚了其中的原理就是对两个数组的所有元素进行遍历，逐个单元进行比较；

9.26（晚上）level 0-2

该题开始有点看不懂（可能我理解能力有点差），但是看到输出效果演示后大概清楚了方向，本题个人认为涉及的知识和level 0-1大致相同，新的收获可能在于再次练习了if-else语句的嵌套关系和对strcmp函数的使用更加熟练和优化了。

下面是我在完成该任务过程中的一些卡点和解决过程：

1. 在使用strcmp的过程中开始我严格按照视频中的写法int strcmp(const char\*s1,const char\*s2),然后再将其值赋给一个整型变量，写起来非常繁琐，且经常容易漏掉int 和const 后面的char

经过查资料，了解到在头文件已经有include<string.h>的情况下，可以直接调用strcmp函数来判断并将其值赋给一个整型变量，最终写成int x=strcmp(a,A);的形式，很好地提高了写代码的效率和准确度；

1. 对于循环结束条件的判断时将决定变量写在循环内，导致读不到这个变量

发现后立即调整该变量位置，并总结出以后写do-while循环时需要事先设置好这种变量；

二编（9.28晚）：经过学长的指导，终于正确的理解了题意，但是对于文件的读取和解析知识到今天为止几乎完全空白（慌的一批），好在学长给出了额外一个晚上的宽限，于是我紧急学习了有关文件处理的知识，了解到了读入文件基本形式包括1.FILE\* file = fopen("data.txt", "r");打开该文件名的文件2.遍历每一行；（以EOF为止）3.fclose(file);

在解析文件的过程中，也接触到了一些函数与平时使用的差异：如fscanf用于将文件内容输入到程序中输入需要使用%[]的形式；

并且根据strcpy函数的复制字符串的功能，实现了对于符合要求的键值对进行存储；

读文件结束后就回到了我相对熟悉的部分，在这里我新了解到了对于没有明确控制变量的循环，可以直接在括号中写1，表示条件永远成立，然后在需要的地方使用break结束循环；

由于卡点和查资料的次数太多，这里不方便统计，于是谈谈我的收获：

1.^ 在 [ ] 字符集中表示否定或排除的意思，可以在输入时排除：对前面键的输入的干扰；

2.加深了对conntinue应用的理解，如在函数中如果遇到该行不符合要求（存在空格，键值重复），无需储存该行的键值时，用continue可以跳过该行进入到下一行的判断；

3.对于输入信息，存储有效信息的过程有了了解，输入过程参与循环，然后将其中的有效值存储到另一个变量，然后开启新一轮的循环；

9.27（中午-下午）level 1-1

从这个任务开始，我就发现我有些想实现的功能以我的知识储备无法实现，例如：当无人登录的时候，输入Exit不应该直接输出logout。另一方面这题的要求和限制比较少，于是我在符合基本要求的情况下，做了一些更贴合生活的调整：如在登陆时加入welcome+用户名，在退登时清空用户名的信息。并且我还对输入内容进行了限制，当输入不符合规则时，会进行报错。

下面是我在完成该任务过程中的一些卡点和解决过程：

1. 开始循环存在问题，每次进入循环都会输出一遍Please enter your account在用户退登时，输出这句话显然不符合逻辑

将printf移到了循环之外；

9.27 （下午-晚上）level 1-2

这个任务对我来说很复杂。但我同样对输入的内容进行了限制并且通过循环实现了输入错误的可修正；并且对于不同floor之间的预约问题也进行了判断。

下面是我在完成该任务过程中的一些卡点和解决过程：

1. 为了实现对输出日期的控制，需要大量定义字符串，效率低下

使用ctrl v+ctrl c提速

1. 其中Floor和各个楼层一直无法取等

经过检查和查资料，发现在字符串中，空格的存在也会影响strcmp的判断，于是统一了字符串的空格问题，最终解决

1. 忽略了整型变量和数组变量的差异性，在scanf\_s中忘记加&

报错后立即在定义的整型变量ord前加入&

1. 预约时选择的座位有偏差

经过反复试验，发现座位总是差一个，后查阅笔记意识到数组编号从0开始，于是对相关变量进行-1的处理

总结：这次学习和项目给我带来了什么收获和教训？

收获：

1. 完善了思维的严谨性，让解决问题的逻辑更清晰：写代码的过程中，人会忽略错误，但运行结果不会忽略错误，无论是一个分号没加，还是条件判断中漏考虑一种情况，程序都会出现问题，我也在一次次的纠错过程中，完善了对于代码细节的考量，解决问题时思考也更加全面；
2. 对于学习知识的热情，加深了对于学习意义的理解：在这个的过程中，总有亟待解决的问题驱动着我去学习这方面的知识，一个个贴近生活的代码任务让我感受到知识和生活的紧密联系，将我对生活的热情注入对于学习的热情；
3. 面对难题的冷静心态和积极分析，备案的态度：完成任务的过程并不是一帆风顺的，有时代码没有提示错误，但结果却永远不对，从开始的束手无策反复检查，到之后学会从之前犯的错误中针对性检查，以及寻求外界的帮助，我体会到了我对于问题解决方案的逐渐成熟；
4. 一个更加坦然的心态：Dian作为一个含金量很高的团队，招新任务的难度对我来说很高，并且竞争也很激烈，我当然没有十足的把握一定能脱颖而出。我也不是没有因为题目的门槛和竞争的压力而自我怀疑过，但是我还是希望去尝试，并且对于自己的态度和热情感到自信，希望能将这一个月的态度和体会践行到接下来的生活中；

教训：

1. 对于任务要求的仔细研读：在我理解错test2后，我深刻体会到了仔细研读要求的必要性，无论是眼下的招新题还是未来可能遇到的现实问题，解决过程都应该是功能目的靶向性的，所以我之后再面对相似的情况时，一定会先仔细阅读，并且向他人验证并询问的；
2. 学习的策略性：这次任务的完成过程让我深刻体会到了“干中学”的真正内涵，和中学的系统性，全面性学习不同的是接下来的学习都是有时限并且有明确的目的的，比如想要实现特定的功能，涉及特定领域的专业知识，比起广而泛的学习，弄清楚需要什么，为什么需要，如何实现之后的针对性学习才更加有效。当然这种策略并非百利无弊，可能会出现一知半解的情况，因此需要根据具体的任务情境合理选择策略。