Diagnostic Recall

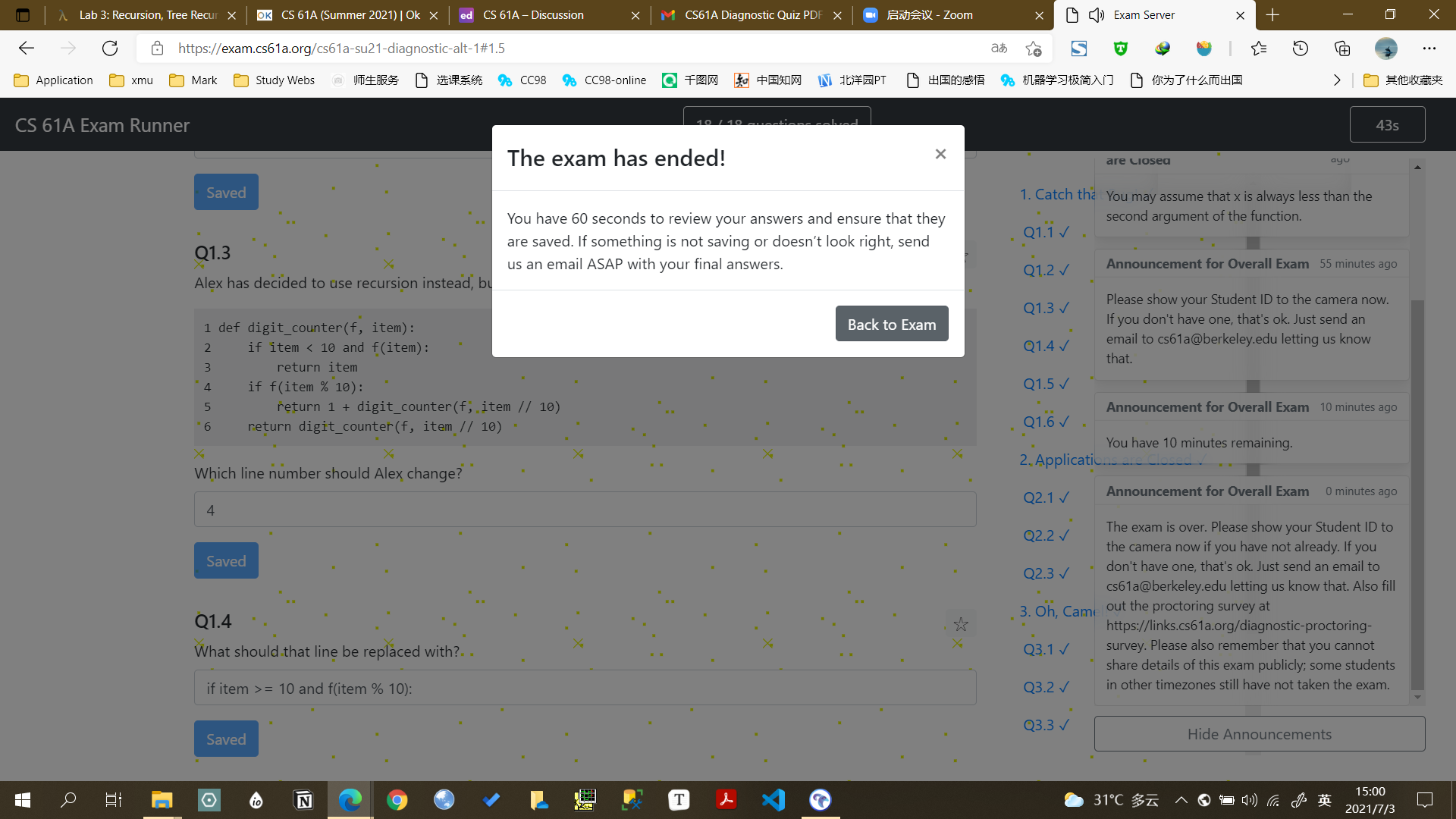
‎2021‎年‎7‎月‎3‎日，CST，‏‎16:41:01

Diagnostic 一共就三个problem。digit\_counter(f,item); application and closed (太简单以至于忘记了题目考的是什么函数和什么考点); is\_camel\_number(n)；

但是每个problem都有local difined function和recursion，我不是很熟练。

而且都是填空题或者debug题。对方给出一段代码。我首先要花时间去理解函数的意思与题目的要求，然后要阅读理解对方的思路！之后再找题目中的问题，去debug！

我发觉自己就是过于忙于code interpreter里测试了，导致很多小的question没有梳理思路。经常反复去读题干和读代码去理解思路，然后发现了一段不对劲的代码，就修改一下看着，然后疯狂test samples。这样耽误了很多时间去试错了！这样的问题在于没有带入具体的例子，去测试他给出代码的各行的意义！我应该带入一些例子，然后人脑逐步编译的！



比如这个recursion-question

最后10分钟也没搞清楚那个recursion的思路，实在想不出来digit\_counter(f,item)这第一个problem的recursion-question的代码是什么思路，为什么有两个return一个return 1+digit\_counter(f,item//10)，而另一个是直接不要1了，recursive case不就应该是func(item)=1+func(item//10)吗？这个对应iteration-question的count=count+1啊！

考前5分钟一直在想这个问题，时间到了我还是没想出来！考完5分钟内，我把f换成is\_even，即is\_even=lambda x: x%2==0；再把item换成4343这个数了，然后突然就懂了！

item%10是3！所以这个3肯定不会让digit\_counter+1的！而是直接返回digit\_counter(f,item//10)，所以Line 6是没错的！

然后到最高位数字4的时候，item=4，这个base case应该return 1！因为位数是1，肯定不是return item!

# Problem1 Catch that Bug!

第一个Problem有三个小question，分别是iteration, recursion和local defined function。

Recursion的问题如之前所述，不多总结了。第一个problem的local defined function也遇到困难了，虽然写出来了，但是是我瞎试出的答案。

那里的helper(0,0)改成helper(1,0)，我也只是抓住了(item//x)%10这一个细节去理解代码的，完全没有去代入例子理解代码书写思路！纯粹是在online coder里test各种samples，这样是非常有问题的！

# Problem2. Applications are Closed

第二个problem倒是很简单，填空还是debug我给忘记了，直接5分钟内搞定了problem2

# Problem3. Camel Sequence

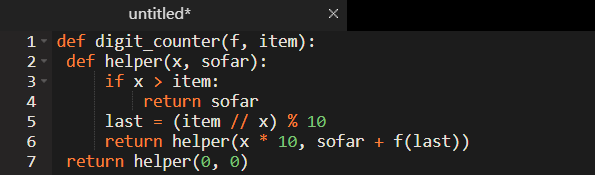
第三个problem的is\_camel\_number(n)也用了helper的func，helper(n,thank)，我直接给看懵了，不知道thank是啥，为什么传入一个thank，后来纯看代码发觉thank像是True or False的Boolean Type。而且也花了不少时间，有10分钟才搞清楚挖出的坑是什么类型的语句，然后再用10分钟才填写完毕并且online coder测试完毕。

而且这个is\_camel\_number(n)我得出来的时候还是带入了n=12这个例子才理清楚思路的！那是考试临时起意代入的一个数，没有发觉其他题目也应该这样，所以其他题目继续死磕以至于时间到了还没写完。

看来带入例子进行对方代码理解和填空，真的极其重要！不能光看字母代码去理解思路！更不能随便修改个别看着不对劲的代码，然后疯狂test samples！

# Review

对于瞎蒙的Q1.3 digit\_counter(f,item)的local defined function version，现在利用考完反思Q1.2的代入数值分析出题人思路的方法，来重新解决这个Q1.3的思路



需要debug的代码是这样的

我代入f=is\_even，“>>>is\_even=lambda x: x%2==0”，然后”>>>digit\_counter(is\_even, 4343)”

然后7分钟就理清这个题目的思路了！具体过程如下：

4343是item之后，先是helper(0,0)

If x>item无效，进入last那句，然后发现//x出错，而且item%10显然是要取余数，然后后面的f(last)显然是要判断这个digit是否f(digit)==True，所以初始的x是1。因此last=3，is\_even(3)是False, 之后就return helper(10,0+False)了，可能sofar+False很费解，不过在terminal试了一下0+False=0, 0+True=1。

所以返回为helper(10,0)，last=(4343//10)%10，显然是取第二位的4，last=4。return helper(100,0+True)即helper(100,1)新的x=100, 新的sofar=1

Helper(100,1)中，x=100>4343仍然是False，所以这个Base Case跳过，last=3, return helper(1000, 1+0)

Helper(1000,1)中，跳过Base Case，last=(4343//1000)%10=4%10=4，最后一个位了。因此return(10000, 1+1)

对于helper(10000,2)，“if x>item: ”的判断即为“x=10000>item=4343” 结果是True，因此return sofar，而sofar就是helper传入的第二个argument，2！这就是item中满足is\_even的digit的数量，也是digit\_counter(f,item)的返回值！

至此成功解决问题，而sofar+f(last)传入以sofar为参数的helper函数，其实就是sofar=sofar+f(last)，而f(last)就起到了如果last is even那么count+1的功能！