整数划分实现报告

王资 18214668

# 实现的基本思想

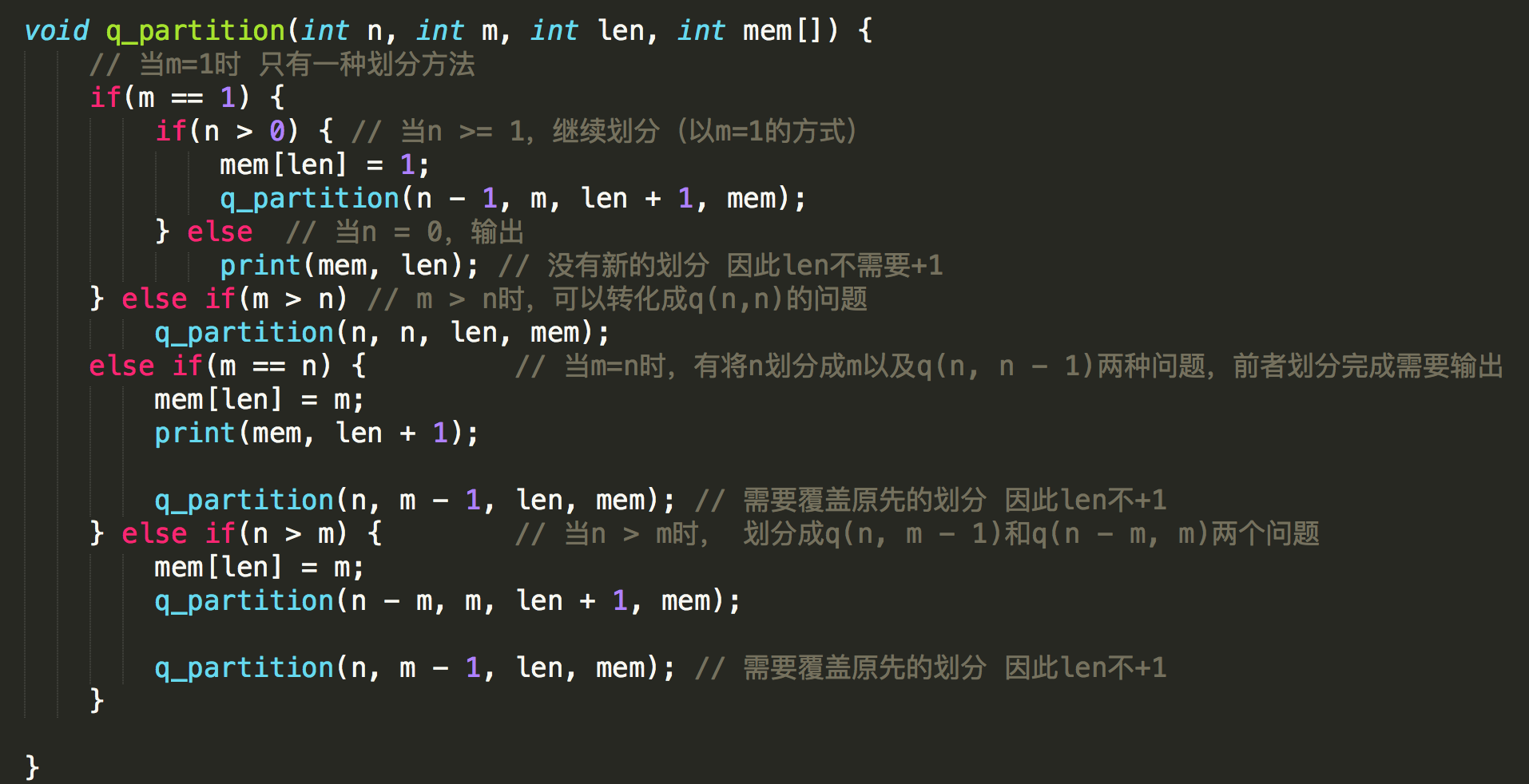
依据整数划分的递归关系，如下图所示，现要实现输出，则在复杂度为1的地方进行输出。



为了输出，程序中的实现函数除了上图n, m两个参数外还有len和mem两个额外参数，n表示需要划分的整数，m表示最大划分，mem表示一个储存器用于记录将要输出的部分划分结果，len表示储存器已储存的整数数量。

# 主要流程

程序的主要部分如下图所示，



与整数划分的递归关系基本相同，即

(1) q(n,1)=1,n≥1;当最大加数n1不大于1时，任何正整数n只有一种划分形式；

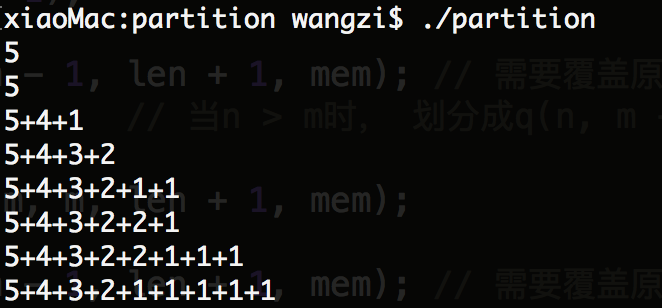
(2) q(n,m)=q(n,n),m≥n; 最大加数n1实际上不能大于n。因此，q(1,m)=1

(3) q(n,n)=1+q(n,n-1);正整数n的划分由n1=n的划分和n1≤n-1的划分组成；

(4) q(n,m)=q(n,m-1)+q(n-m,m),n>m>1;正整数n的最大加数n1不大于m的划分由n1=m的划分有q(n-m,m)个,和n1≤m-1 的划分q(n,m-1)。

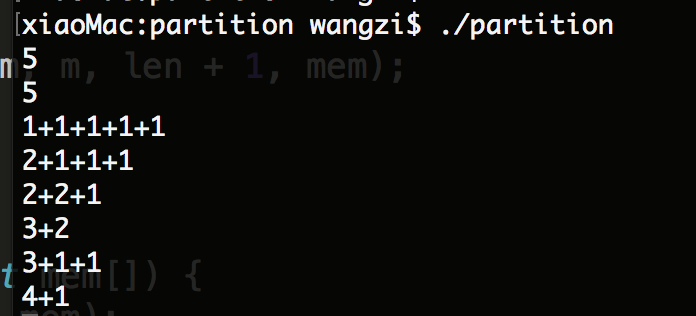
# 出现过的问题

1. 输出混乱，如下图所示



该问题出现是原因是对储存器mem的长度len在递归过程中设定不当。解决方式是深入了解整数划分递归式的细节，确认好流程中什么部分储存器mem的长度len需要增长，而什么部分需要覆盖掉上一次的结果而让长度len保持不变。

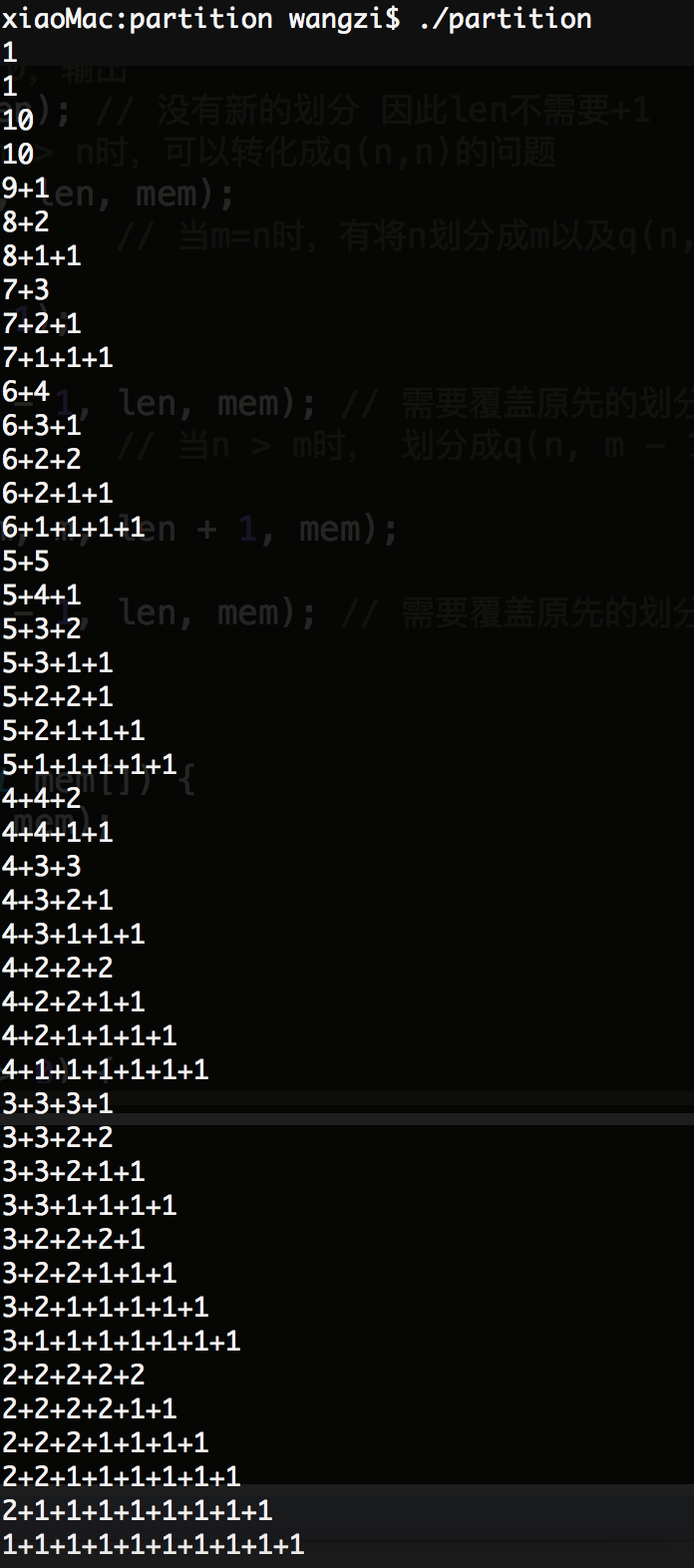
1. 反序输出，如下图所示



该问题出现的原因是按照整数划分递归式，先对q(n, m-1)进行划分，后对q(n – m, m)进行划分造成的，若先对q(n, m-1)进行划分，按照程序流程，最终将递归进行到q(n, 1)，即先划分到最小，因此输出的顺序相反，解决的方法是先对q(n – m, m)进行划分，后对q(n, m-1)进行划分。

# 输入及输出结果

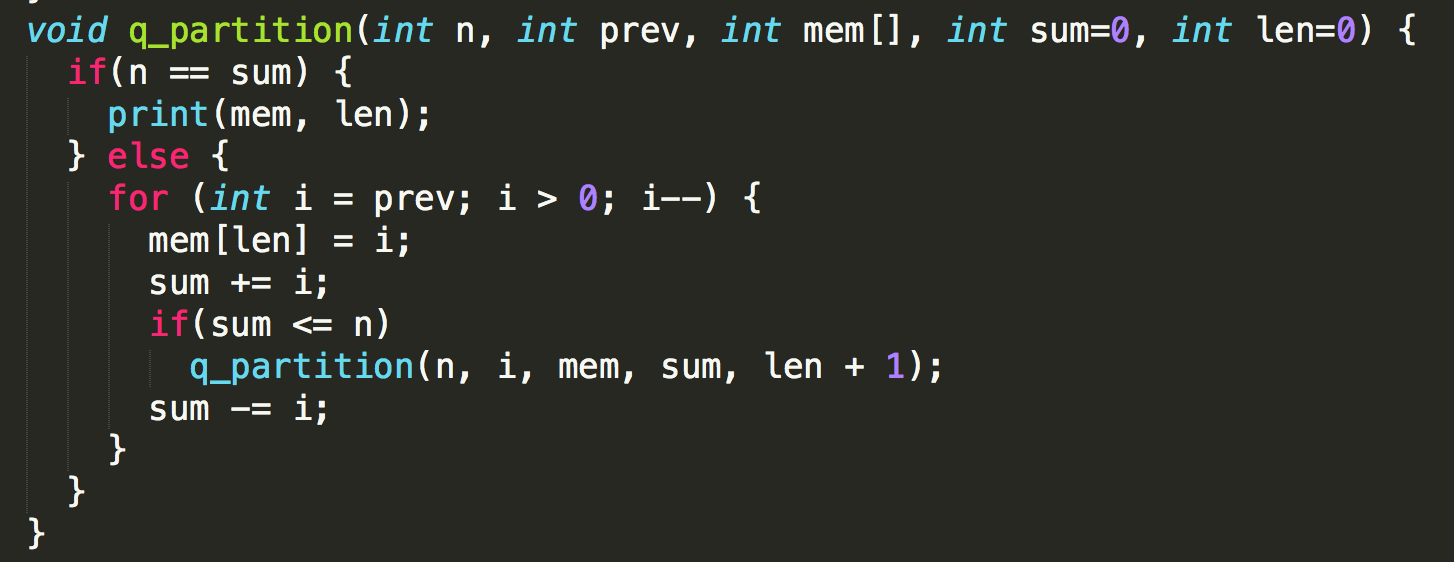
对程序分别输入1和10，结果如下图



由于具有递归关系，输入的10相当于对1-10的全部输入都进行了校验，输入值1是程序的最小有效输入，并且有正确的输出。从上面的输出结果图可以看出程序在输入1-10的情况下都能正确输出，其他情况在本地进行了校验且正确但由于结果输出太长不贴在报告中。

# 其他方法

通过查阅资料，得知还可以使用深度优先算法得到相同的结果，详细代码在partition2.cpp中，代码主要部分如下图所示。



其中n代表需要划分的整数，prev为上一个累加的值，mem为划分输出的储存器，sum为所有累加的值，len为储存器的长度。

该方法的设计较直观，很容易看得出可以正确输出希望的结果，但其复杂度会比一般的递归实现高很多，为

# 心得体会，存在问题，改进设想

虽然已有递归表达式，但要将整数划分的每种划分结果正确输出需要很多的小技巧，一个对于调用栈来说的全局变量储存器mem自不用说，对于何时更新mem中的元素以及何时增长mem的长度len还是需要仔细分析递归表达式，必要时还要用一个实例去展开递归表达式，画成一棵树，以及实际运行程序，查看中间和最终结果，再分析修改代码。

存在的主要问题就是储存器mem的长度len更新的问题，十分不直观，很容易搞错，并且有一处错误输出就会很混乱难以识别到底是什么地方出错。

这个算法经过这么多年的研究，在算法层面应是没有什么我能提出意见的地方，但在实现部分应当要统一储存器长度len的表达，否则时而需要+1时而不需要，在阅读代码的时候就有很大的障碍。