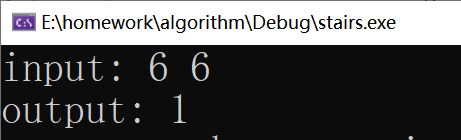


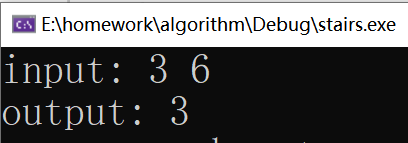
Q1：

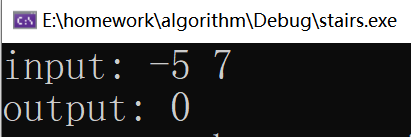
·方法1：

1. 首先判断台阶数目m是否为1或者2，单独判断m == 1与m == 2的情况，即若m == 1时，判断卡路里n >= 1，则返回方法数1（利用1点卡路里走一步），否则返回方法数0；若m == 2时，判断卡路里n >= 3，则返回方法数2（可以利用3点卡路里走两步，也可以利用2点卡路里走两次一步），否则判断若n >= 2，返回方法数1（尽可以利用2点卡路里走两次一步），否则返回方法数0。
2. 若m >= 3，设置达到每层总共方法数num\_of\_ways以及可以有的走一步的次数、走两步的次数以及所消耗的卡路里的组合，每个组合再映射到该组合的组合数。求解第i层的参数时，第i层的方法数num\_of\_ways可以从第i - 1层走一步上来的方法数加上第i - 2层走两步上来的方法数；然后通过i - 1层以及i - 2层的走一步的次数、走两步的次数以及所消耗的卡路里的组合对第i层的进行更新，若出现相同的则进行合并；同时更新这些组合到组合数的映射。
3. 重复II中步骤，直到计算到第m层台阶为止。
4. 第m层所对应的num\_of\_ways即为所求。

测试：

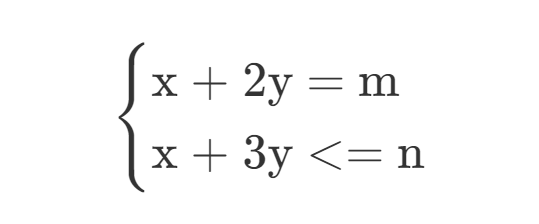






·方法2：

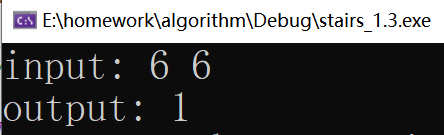
1. 设爬上m个台阶需要x次一步，y次两步，则有关系

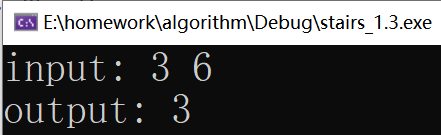


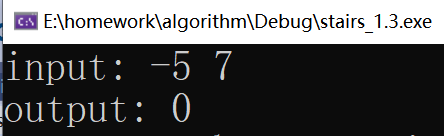
并且可得出关系0 <= y <= n - m

1. 若n - m < 0，则说明卡路里不足以爬上m个台阶，放回方法数0
2. 否则，遍历y从0到n - m，通过y求解出x，保证x >= 0的前提下，由此得出x次一步与y次两步的组合，然后通过x和y求出该组合对应的组合数。
3. 重复II中操作，直到将所有组合数相加，其和即为所求。

测试







Q2：

·方法1：

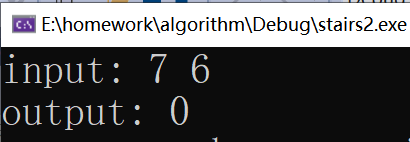
1. 首先判断台阶数目m是否为1或者2，单独判断m == 1与m == 2的情况，即若m == 1时，判断卡路里n >= 1，则返回方法数1（利用1点卡路里走一步），否则返回方法数0；若m == 2时，判断卡路里n >= 3，则返回方法数2（可以利用3点卡路里走两步，也可以利用2点卡路里走两次一步），否则判断若n >= 2，返回方法数1（尽可以利用2点卡路里走两次一步），否则返回方法数0。

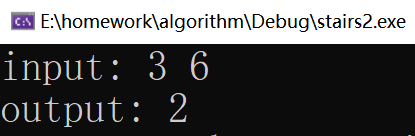
ii. 若m >= 3，设置达到每层总共方法数num\_of\_ways以及可以有的走一步的次数、走两步的次数以及所消耗的卡路里的组合，每个组合再映射到该组合的组合数。求解第i层的参数时，第i层的方法数num\_of\_ways可以从第i - 1层走一步上来的方法数加上第i - 2层走两步上来的方法数；然后通过i - 1层以及i - 2层的走一步的次数、走两步的次数以及所消耗的卡路里的组合对第i层的进行更新，若出现相同的则进行合并；同时更新这些组合到组合数的映射。

iii. 重复II中步骤，直到计算到第m层台阶为止

iv. 在最高层m层对应的组合到组合数的映射中，寻找走两步的次数最多的组合，该组合映射的组合数即为所求。

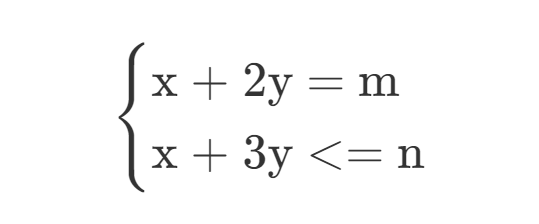
测试





·方法2：

I.设爬上m个台阶需要x次一步，y次两步，则有关系



并且可得出关系0 <= y <= n - m

II. 遍历y从n - m到0，求解出x的值，在x >= 0的前提下，由此得出x次一步与y次两步的组合，然后通过x和y求出该组合对应的组合数。

III. 跳出循环，所求组合数即为所求。

测试

