# 小Q的歌单（腾讯2018秋招）

小 Q 有 X 首长度为 A 的不同的歌和 Y 首长度为 B 的不同的歌，现在小 Q 想用这些歌组成一个

总长度正好为 K 的歌单，每首歌最多只能在歌单中出现一次，在不考虑歌单内歌曲的先后顺序的情况下，

请问有多少种组成歌单的方法。

输入描述：

每个输入包含一个测试用例。

每个测试用例的第一行包含一个整数，表示歌单的总长度 K(1<=K<=1000)。

接下来的一行包含四个正整数，分别表示歌的第一种长度 A(A<=10)和数量 X(X<=100)以及歌的第二种长度

B(B<=10)和数量 Y(Y<=100)。保证 A 不等于 B。

输出描述：

输出一个整数,表示组成歌单的方法取模。因为答案可能会很大,输出对 1000000007 取模的结果。

输入示例：

5

2 3 3 3

输出示例：

9

解题思路：

可以首先把这个问题分成两步：

第一步是求解用多少个长度为A的歌曲和多少个长度为B的歌曲组合可以得到长度为K的歌单，也就是求出长度为A的歌曲的数量，和长度为B的歌曲的数量。记录每一种情况。

第二步，针对每一种A和B的组合。我们来求它有多少种组合可能。假设m个长度为A的歌曲和n个长度为B的歌曲可以得到一个长度为K的歌。那么我们只需求出从X首长度为A的歌曲种选出m首歌的可能情况个数p，以及从y首长度为B的歌曲种选出n首歌的可能情况个数q。p\*q就是该组合的可能情况总数。

那么我们如何求出从x首哥里选出n首歌有多少种情况呢？

首先有如下组合公式：=+

根据这个公式，我们可以利用动态规划的方式来求解这个问题。因为x<=100。那么我们只需要求出100以内的组合数。

定义一个dp[100][100]的二维数组，dp[i][j]就表示的数值。很显然，dp[i][0]为1。

dp[i][j]=dp[i-1][j-1]+[i-1][j]。

具体代码如下：

**package** Tencent;

**import** java.util.Scanner;

**public** **class** SongList {

**static** **int** *mod*=1000000007;

**public** **static** **void** main(String[] args) {

Scanner in=**new** Scanner(System.***in***);

**int** k=in.nextInt();

**int** A=in.nextInt();

**int** x=in.nextInt();

**int** B=in.nextInt();

**int** y=in.nextInt();

**long** dp[][]=*permutation*(100);

System.***out***.println(*CountOfOption*(A, x, B, y, k, dp));

}

**public** **static** **long** CountOfOption(**int** A,**int** x,**int** B,**int** y,**int** k,**long**[][] dp) {

**long** count=0;

**for**(**int** i=0;i<=x;i++)

**if**(i\*A<=k&&(k-i\*A)%B==0&&(k-i\*A)/B<=y)

count=(count+(dp[x][i]\*dp[y][(k-i\*A)/B])%*mod*)%*mod*;

**return** count;

}

**public** **static** **long**[][] permutation(**int** n) {

**long** dp[][]=**new** **long**[n+1][n+1];

**for**(**int** i=1;i<=n;i++) {

dp[i][0]=1;

dp[i][i]=1;

**for**(**int** j=1;j<i;j++)

dp[i][j]=(dp[i-1][j-1]+dp[i-1][j])%*mod*;

}

**return** dp;

}

}