题目要求：<31-P1164-小A点菜-题目要求.docx>

## 分析

这是一道简单的动规题，定义f[i][j]为用前i道菜用光j元钱的办法总数，其状态转移方程如下：

（1）if(j==第i道菜的价格)f[i][j]=f[i-1][j]+1;

（2）if(j>第i道菜的价格) f[i][j]=f[i-1][j]+f[i-1][j-第i道菜的价格];

（3）if(j<第i道菜的价格) f[i][j]=f[i-1][j];

说的简单一些，这三个方程，每一个都是在吃与不吃之间抉择。若钱充足，办法总数就等于吃这道菜的办法数与不吃这道菜的办法数之和；若不充足，办法总数就只能承袭吃前i-1道菜的办法总数。依次递推，在最后，我们只要输出f[n][m]的值即可。

很明显的动态规划。而且有点像背包问题。

不过即使搜索，似乎也能过一些点。

方程 f[i,j]:=f[i-1,j]+f[i-1,j-a[i]];

数组表示在前i道菜中，总价格为j。

你可以不点这道菜（f[i-1,j]），或者点（f[i-1,j-a[i]]）

时间复杂度O(MN)。有的OJ可能会通过不了百万级的时间复杂度，不过洛谷Online Judge就随便了。

空间复杂度O(MN)足够了，但是使用循环队列可以减成O(M)，只是你想不想的问题了。

## 推导

代码和楼上楼下的都差不了太多，我就来解释一下如何推出来下面的这个式子的吧

f[j]+=f[j-a[i]]

### 过程1

f[i][j]表示前i个菜品恰好花费j元的方案数

然后我们来考虑如何转移：

首先，因为f数组存储的是方案数，所以思路可以是它可以由那些方案转移过来

1.第一种可行的方案是买当前第i道菜品，这个时候前i-1个物品所需要的钱应该是j-a[i],\*\* 也就是说前i个物品中所有能凑出j-a[i]元的方案再加上当前这道菜品，就可以变成前i个物品所需的钱为j的方案数。\*\*即f[i][j]+=f[i-1][j-a[i]]

2.不买当前第i道菜品，这时候，也就是前i-1个物品凑成j的方案，即f[i][j]+=f[i-1][j];

注意这里是方案的叠加，因为每一种方案都是可行的。

你可能以为现在已经结束了，但实际上你还没有考虑一种方案，当前第i种物品恰好为j元钱，所以可以只买它自己。这种情况其实包含在1中，但是由于f[i][0]=0，所以不会算入这种情况。所以我们要把所有的f[i][0]更新成1，这样就可以计算上面所述的那种情况。

然后你就可以得到一份这样的代码：

1. #include<iostream>
2. #include<cstdio>
3. #include<cmath>
4. #include<algorithm>
5. #define N 110
6. using namespace std;
7. int n,m,a[N],f[N][10010];
8. int main()
9. {
10. //freopen(".in","r",stdin);
11. //freopen(".out","w",stdout);
12. scanf("%d%d",&n,&m);
13. for(int i=1;i<=n;i++)
14. scanf("%d",&a[i]);
15. for(int i=0;i<=n;i++) //注意这里要从0开始
16. f[i][0]=1;
17. for(int i=1;i<=n;i++)
18. for(int j=1;j<=m;j++)
19. {
20. f[i][j]+=f[i-1][j];
21. if(j>=a[i])
22. f[i][j]+=f[i-1][j-a[i]];
23. }
24. cout<<f[n][m];
25. return 0;
26. }

### 过程2

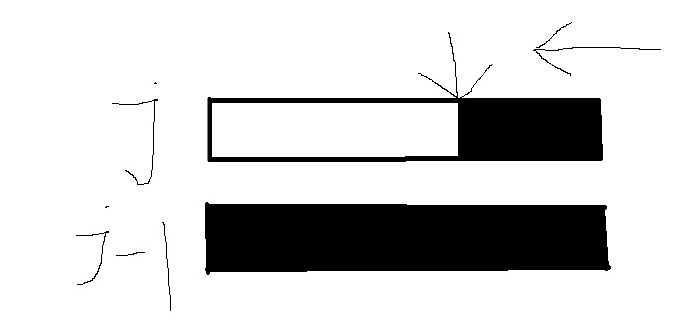
接下来就是如何把数组从2维降到1维了

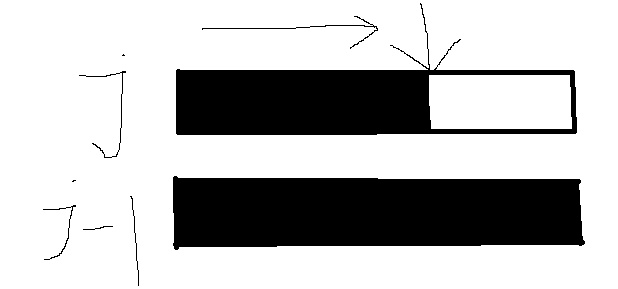
我们先来考虑这样的一个问题，我们要输出a\*b的答案，如果硬要用数组做的话可以用一个for循坏从1到a然后令s[i]=s[i-1]+b;最后输出s[a]。

但是通常情况下，我们都是直接用s+=b来实现这个问题，这就是一种数组降维。为什么可以降下来，因为这个问题我们每一次只会使用到i-1，而i-1就是上一次做完留下来的值，所以根本不需要用数组来做这个问题。

然后回到原问题，我们发现f数组的第一维每次也只是使用到了i-1，所以说我们可以给f数组降维。

但是这个时候需要注意一个问题，我们的第二层for循环不能正着跑了，为什么？我们可以观察一下下面两幅图片





第一个图片表示倒着跑，这个时候遍历到的当前的j就是上一次的j

而第二个图片，遍历到的j是刚刚更新了一次的j

最终代码

1. #include<iostream>
2. #include<cstdio>
3. #include<cmath>
4. #include<algorithm>
5. #define N 110
6. using namespace std;
7. int n,m,a[N],f[10010];
8. int main()
9. {
10. *//freopen(".in","r",stdin);*
11. *//freopen(".out","w",stdout);*
12. scanf("%d%d",&n,&m);
13. for(int i=1;i<=n;i++)
14. scanf("%d",&a[i]);
15. f[0]=1;
16. for(int i=1;i<=n;i++)
17. for(int j=m;j>=a[i];j--)
18. f[j]=f[j]+f[j-a[i]];
19. cout<<f[m];
20. return 0;
21. }

## 我的代码：

附上文件路径：<31-P1164-小A点菜.cpp>

1. *#include<stdio.h>*
2. *#define max(m,n) (((m)>(n))?(m):(n))*
3. int n,m,dp[105][10005],menu[105];
4. int main(void)
5. {
6. scanf("%d%d",&n,&m);
7. for(int i=0;i<n;i++) scanf("%d",&menu[i]);
8. for (int p=1;p<=n;++p)
9. {
10. for (int q=1;q<=m;++q)
11. {
12. if(q>menu[p-1]) dp[p][q]=dp[p-1][q]+dp[p-1][q-menu[p-1]];
13. else if(q==menu[p-1]) dp[p][q]=dp[p-1][q]+1;
14. else dp[p][q]=dp[p-1][q];
15. }
16. }
17. printf("%d",dp[n][m]);
18. }