**★★★☆☆**

**題組：Problem Set Archive** **with Online Judge**

**題號：11026: A Grouping Problem**

**解題者：楊昌彪**

**解題日期：2017年4月6日**

**題意：**

 給定一正整數N (2<=N<=1000) 和一正整數M，以及一串數列，總共有N個，求出在這數列任意取出K個數的乘積總和並 mod M的值，最大者即為答案。(K = 1 ~ N)

舉例:

N = 4, M = 10, case integers = 1, 2, 3, 4

首先，C(4, 1) 取出{1, 2, 3, 4}; 總和為10, mod 10 後為0

第二，C(4, 2) 取出{1, 2}, {1, 3}, {1, 4}, {2, 3}, {2, 4}, {3, 4}共六種，乘積總和為1\*2 + 1\*3 + … + 3\*4 =35; mod 10 後為 5, 以此類推, 推到C(4, 4)。

**題意範例：**

#input

4 10 # N = 4, M = 10

1 2 3 4 # N positive integers

4 100 # second case, N = 4, M = 100

1 2 3 4 # N positive integers

4 6 # third case, N = 4, M = 6

1 2 3 4 # N positive integers

0 0 # end of input (N = 0, M = 0)

# output

5

50

5 # three cases, three results, end of output

**解法：**

dynamic programming

**解法範例：**

 可以參考經由組合遞迴式 推演出來的公式

 /\*recurrence\*/

 /\*

 S(n, k) = an \* S(n-1, k-1) + S(n-1, k)

 It's from:

 C(a, b) = C(a-1, b-1) + C(a-1, b)

 \*/

使用一個二維陣列 exArray[n][k], 兩個的index分別是S(n, k)裡的n和k

由於k永遠<=n 所以以迴圈去跑 一層一層填滿 之後就會有所有在N種情況下 取1~N個物品組合數的大小(包含取出元素的乘積和)

最後找出誰是最大即可找出答案

**討論：**

公式怎來的?

由組合遞迴式推導而來

先看C(a, b) = C(a-1, b-1) + C(a-1, b)

可以用C(4, 3)舉例，在四個物品任意取3個的組合，可以把它拆成兩個小問題，假設四個物品分別是m, n, k, l, 那就可以拆成兩個問題:

取到的三個中有包含m (組合數為C(a-1, b-1) )

和沒包含m(組合數為C(a-1, b) )。

接著再看 S(n, k) = an \* S(n-1, k-1) + S(n-1, k)

 an 代表的是m, n, k, l, 這些元素，S(n,k)代表的即是在C(n,k)下的乘積總和，因為要找的是乘積和，所以在有取到該目標數C(n-1, k-1)的這項乘上元素(an)，以此透過遞迴的方式即可找到答案。

 Ex: 延續上面的例子，假設現在an = l, 那S(4, 3)就會是

 l \*{ {m, n}, {m, k}, {n, k} }的乘積在加上S(3, 3)即可得知S(4, 3)

 recurrence 的初始條件?

 exArray[0][0] = 1;

 exArray[i][0] = 1; (1 <= i <= N)

**程式：**

#include <iostream>

#include <algorithm>

using namespace std;

long long sum [1005][1005];

long long compute(int box[], int N, int M);

int main ()

{

 int N=0;

 int M=0;

 cin >> N >> M;

 int box[1005];

 while(N!=0||M!=0){

 for(int i=1;i<=N;i++)

 cin >> box[i];

 cout << compute(box, N, M) << endl;

 cin >> N >> M;

 }

 return 0;

}

long long compute(int box[], int N, int M)

{

 long long ans=0;

 sum[0][0]=1;

 /\*recursion\*/

 /\*

 S(n, k) = a\_n \* S(n-1, k-1) + S(n-1, k)

 It's from:

 C(a,b) = C(a-1, b-1) + C(a-1, b)

 \*/

 for(int i=1;i<=N;i++){

 sum[i][0]=1;

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

 sum[i][j] = box[i]\*sum[i-1][j-1] + sum[i-1][j];

 if(sum[i][j]>=M)

 sum[i][j]%=M;

 }

 }

 for(int i=1;i<=N;i++)

 ans = max(sum[N][i], ans);

 return ans;

}