#include <iostream>

#include <cstdio>

#include <algorithm>

#include <cstring>

#include <cmath>

using namespace std;

const int MAXN = 85, Mod = 10000; // 高精四位压缩大法好

int n, m;

int ar[MAXN];

struct HP

{

    int p[505], len;

    HP()

    {

        memset(p, 0, sizeof p);

        len = 0;

    } // 这是构造函数，用于直接创建一个高精度变量

    void print()

    {

        printf("%d", p[len]);

        for (int i = len - 1; i > 0; i--)

        {

            if (p[i] == 0)

            {

                printf("0000");

                continue;

            }

            for (int k = 10; k \* p[i] < Mod; k \*= 10)

                printf("0");

            printf("%d", p[i]);

        }

    } // 四位压缩的输出

} f[MAXN][MAXN], base[MAXN], ans;

HP operator+(const HP &a, const HP &b)

{

    HP c;

    c.len = max(a.len, b.len);

    int x = 0;

    for (int i = 1; i <= c.len; i++)

    {

        c.p[i] = a.p[i] + b.p[i] + x;

        x = c.p[i] / Mod;

        c.p[i] %= Mod;

    }

    if (x > 0)

        c.p[++c.len] = x;

    return c;

} // 高精+高精

HP operator\*(const HP &a, const int &b)

{

    HP c;

    c.len = a.len;

    int x = 0;

    for (int i = 1; i <= c.len; i++)

    {

        c.p[i] = a.p[i] \* b + x;

        x = c.p[i] / Mod;

        c.p[i] %= Mod;

    }

    while (x > 0)

        c.p[++c.len] = x % Mod, x /= Mod;

    return c;

} // 高精\*单精

HP max(const HP &a, const HP &b)

{

    if (a.len > b.len)

        return a;

    else if (a.len < b.len)

        return b;

    for (int i = a.len; i > 0; i--)

        if (a.p[i] > b.p[i])

            return a;

        else if (a.p[i] < b.p[i])

            return b;

    return a;

} // 比较取最大值