NOIP 2020 在线模拟赛试题与题解

信奥题库 NOIP 2020 在线模拟赛已于 11 月 28 日顺利举行，现公开模拟赛试题、参考标程与题解，供各位选手参考。

A 题：有趣的函数

有趣的函数

输入文件名：function.in
输出文件名：function.out
共 20 个测试点，每个测试点 5 分
每个测试点限时 2 秒，运行内存上限 512MB

“阿，这漫天星光的规律，就藏在那自然对数里吧。”**老 Win**感叹道。

这天他发现了一个有趣的函数 $f(x)$，这个函数是这样的，其中 e 是自然对数，**取值约为** $2.718\ldots\ldots$：

$f(x) = f(x-e) + f(x-2)$

其中当 $0 \le x < 3$ 时，$f(x) = 1$

他一眼就看穿了这个问题的简单解。

但是他想考一考你，作为即将要参加 NOIP 的**高中生**选手。

由于老 Win 最近很讨厌高精度算法，所以他打算直接叫你对 998244353 取模，以减轻**蒟蒻**出题人的烦恼。

输入格式

一个整数 T，表示数据组数

T 行每行一个整数 n，表示询问 $f(n)$

输出格式

T 行每行一个整数，表示 $f(n) \mod 998244353$

数据范围

对于 10% 的数据，满足 $1 \le n \le 10$

对于 30% 的数据，满足 $1 \le n \le 1000$

对于 60% 的数据，满足 $1 \le n \le 100000$

对于 80% 的数据，满足 $1 \le n \le 1000000$

对于 100% 的数据，满足 $1 \le n \le 10000000, T = 5$

样例输入

```
5
10
1000
100000
1000000
10000000
```

样例输出

```
12
919123937
824738881
112429956
121400987
```

题解

这题十分的简朴，不过为了恶心人用了e。

大概就是枚举用了多少个e，然后剩下的用2的方案数肯定是$O(1)$个的，然后就可以得出用x个e和y个2可以拼出来答案。

然后看一下最后一步能不能用e或者2来跳就可以了，答案就是个组合数的和。

参考标程

```cpp
#include <bits/stdc++.h>
using namespace std;

const int N = 10000010;
int       T, n;
int       fac[N], inv[N];
double    e   = exp(1);
const int mod = 998244353;

int       ksm(int x, int t)
{
    int tot = 1;
    while (t) {
        if (t & 1) tot = 1ll * tot * x % mod;
        x = 1ll * x * x % mod;
        t /= 2;
    }
    return tot;
}

void ad(int& x, int y) { x = (x + y >= mod) ? (x + y - mod) : (x + y); }

int  C(int x, int y) { return 1ll * fac[x] * inv[x - y] % mod * inv[y] % mod; }

int  main()
{
    freopen("A.in", "r", stdin);
    freopen("A.out", "w", stdout);
    scanf("%d", &T);
    n      = 10000000;
    fac[0] = 1;
    for (int i = 1; i <= n; i++) fac[i] = 1ll * fac[i - 1] * i % mod;
    inv[n] = ksm(fac[n], mod - 2);
    for (int i = n - 1; i >= 0; i--) inv[i] = 1ll * inv[i + 1] * (i + 1) % mod;
```

```
    while (T--) {
        scanf("%d", &n);
        if (n < 3) {
            printf("1\n");
            continue;
        }
        int ans = 0;
        for (int i = 0; i * e <= n; i++)
            for (int j = max((int)floor((n - i * e - 3) / 2), 0); j * 2 + i * e <= n; j++) {
                if (n - i * e - j * 2 >= 3) continue;
                if (n - i * e - (j - 1) * 2 >= 3 && j) ad(ans, C(i + j - 1, j - 1));
                if (n - (i - 1) * e - j * 2 >= 3 && i) ad(ans, C(i + j - 1, i - 1));
            }
        printf("%d\n", ans);
    }
}
```

B 题：能量球

能量球

输入文件名：energy.in
输出文件名：energy.out
共 20 个测试点，每个测试点 5 分
每个测试点限时 2 秒，运行内存上限 512MB

“阿巴，我快没有电了。”**老 Win** 的女朋友深情的看着 **老 Win**，呢喃着。

老 Win 的女朋友是一部智能手机，**老 Win** 很爱她，每时每刻都爱护着她，无论发生什么事，他都不会离开女朋友。

可是她的电量并不是普通的，可能是 $(-\inf, \inf)$ 之间的任意一个整数。

现在你手中有 n 个能量球，这 n 个能量球的能量为 $1, 2, 3, ..., n$。

你可以选出若干个能量球，并给每一个选出来的能量球附上对应的能量系数，由于你没有 **老 Win** 这么强大，所以能量系数只能为整数。

如果能量球 $\times$ 对应的能量系数再相加恰好等于她的电量，那么 **老 Win** 的女朋友就可以被救回来了。

由于她的电量是随机的，所以你只要满足选出的能量球存在一组能量系数使得其可以表示成任何一个整数就可以。

老 Win 手中有 N 个能量球，他当然知道如何快速救回女朋友，他只想把前 n 个能量球给你，并且问你，有多少种选出能量球的方案可以把他的女朋友救回来。

而且由于女朋友每天都会没有电，所以他想问你 T 次。

由于 **老 Win** 最近很讨厌高精度算法，所以他打算直接叫你对 998244353 取模，以减轻**蒟蒻**出题人的烦恼。

输入格式

第一行一个整数 T，表示 T 次询问。

T 行每行一个整数 n，表示这次他将会把能量值为 1 到 n 的能量球给你。

输出格式

T 行每行一个整数表示每次方案数对 998244353 取模的结果。

数据范围

对于 20% 的数据，$n \leq 20$

对于 40% 的数据，$n \leq 1000$

对于 60% 的数据，$n \leq 30000$

对于另外 20% 的数据，满足 $T = 1$

对于 100% 的数据，$1 \leq n \leq 300000, 1 \leq T \leq 300000$

样例输入

```
19
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
```

样例输出

```
1
2
11
236
65243
301923282
627961329
552143461
457396877
517491896
844068972
93858167
61592432
272124541
419448881
305862735
472197008
477029493
611084457
```

样例解释

对于 $n = 4$ 来说，除了 空集,$\{2\}, \{3\}, \{4\}, \{2, 4\}$ 都是可行的，所以共有 $16 - 5 = 11$ 种方案。

题解

简单的莫比乌斯反演。

我们先用$vector$把每一个数的因子存下来，时间复杂度$n \ln n$。

考虑构造容斥$f(d) = 2^{t(d)} - 1$，其中$t(d)$表示d的倍数有多少个，$f(d)$就表示有多少个集合的gcd的因子中有d。

那么答案显然就是$sum_{i=1}^{n} \mu(d) f(d)$，因为假若一个集合的$gcd$为$d$，那么它在因子处会被算到，系数是莫比乌斯函数，所以$\sum_{g|d} \mu(g) = [d = 1]$。

将n离线下来维护$f(d)$即可。

如果打了暴力没有加判重优化会被卡成0分，因为没有人规定$T <= n$。

参考标程

```
#include <bits/stdc++.h>
using namespace std;

const int   N = 300010;
vector<int> V[N];
int         mu[N], p[N], Ans[N], T, n, t[N], ci[N], ans;
bool        vis[N];
const int   mod = 998244353;

void        read(int& x)
{
    char ch = getchar();
    x       = 0;
    while (ch < '0' || ch > '9') ch = getchar();
    while (ch >= '0' && ch <= '9') x = x * 10 + ch - '0', ch = getchar();
}

void ad(int& x, int y) { x = (x + y >= mod) ? (x + y - mod) : (x + y); }

void ins(int x) { ad(ans, 1ll * mu[x] * ci[t[x]] % mod), t[x]++; }

int  main()
{
    freopen("B.in", "r", stdin);
    freopen("B.out", "w", stdout);

    read(T);

    n     = 300000;
    mu[1] = 1;
```

```
    for (int i = 2; i <= n; i++) {
        if (!vis[i]) p[++p[0]] = i, mu[i] = mod - 1;
        for (int j = 1; j <= p[0] && 1ll * i * p[j] <= n; j++) {
            vis[i * p[j]] = true;
            if (i % p[j] == 0) break;
            if (mu[i] == 0)
                mu[i * p[j]] = 0;
            else
                mu[i * p[j]] = mod - mu[i];
        }
    }
    ci[0] = 1;
    for (int i = 1; i <= n; i++) ci[i] = ci[i - 1], ad(ci[i], ci[i - 1]);
    for (int i = 1; i <= n; i++)
        if (mu[i])
            for (int j = i; j <= n; j += i) V[j].push_back(i);
    for (int i = 1; i <= n; i++) {
        for (int j = 0; j < V[i].size(); j++) ins(V[i][j]);
        Ans[i] = ans;
    }
    while (T--) read(n), printf("%d\n", Ans[n]);
}
```

C 题：修剪树枝

修剪树枝

输入文件名：tree.in
输出文件名：tree.out
共 20 个测试点，每个测试点 5 分
每个测试点限时 2 秒，运行内存上限 512MB

“阿巴巴，外面那棵树越来越杂乱无章了呢”，**老 Win** 的同班同学指到。

听到这句话，**老 Win** 就不自觉羞愧起来，作为一个专业的 **树枝修剪员**，他不允许有人在他面前说出这样的话。

他一拍桌面，跑出操场，一看，原来是最简单的直径树木修剪啊。

这个问题是这样的：

这棵树可以被简化为一棵 n 个点的树，也就是 n 个点 $n-1$ 条边的无向连通图。

一棵树在今天认为是杂乱无章，当且仅当直径长度 > **老 Win** 今天的心情指数。

老 Win 今天做了 10^9 道数竞题，很累，所以他问你，最少切割多少条边才可以使得这棵树切割后所变成的森林中的每一棵树都不杂乱无章。

直径是啥？**老 Win** 说用尺子量一量就知道。

输入格式

两个正整数 n, k 表示今天 **老 Win** 要处理的树的点数和 **老 Win** 今天的心情指数。

接下来 $n-1$ 行，每行三个正整数 x, y, c 表示 x 到 y 有一条长度为 c 的边。

输出格式

输出最少切割边数。

数据范围

对于 20% 的数据，满足 $1 \leq n \leq 20$

对于另外 10% 的数据，满足 $y = x + 1$

对于另外 10% 的数据，满足 $x = 1$

对于另外 20% 的数据，满足 $1 \leq n \leq 2000$

对于 100% 的数据，满足 $1 \leq n \leq 1000000, 1 \leq x, y \leq n, x \neq y, 1 \leq c, k \leq 10^9$

样例输入

```
5 5
1 2 2
1 3 3
2 4 1
2 5 4
```

样例输出

```
1
```

样例解释

切割 $(1, 2)$ 这条边最优。

样例输入2

点击下载（点击“阅读原文”进入信奥题库 NOIP 2020 在线模拟赛下载。）

样例输出2

点击下载（点击“阅读原文”进入信奥题库 NOIP 2020 在线模拟赛下载。）

题解

考虑贪心。

随便定一个根，往下dfs。

返回的时候返回该子树往下走的最长路径长度。

每次一个点将所有儿子的dis排个序，如果最大长度+次大长度$>k$，那么就把最大长度所对应子树的边删掉。

然后就可以得到答案。

考虑贪心的正确性，采用数学归纳法，当只有一个点的时候，啥边都不用删，肯定正确。当前在x节点上，任意一个子树的答案都是正确的。

如果最大长度+次大长度$>k$，那么说明最大长度和次大长度所在的这条链上肯定要删一条边，这条边肯定不会删在儿子的子树内，因为儿子的子树已经满足条件了，外面的任意一条链想要进入子树必定要先经过x到儿子这两条边，所以删x到儿子这两条边严格优于删儿子的子树内的边，而这两条边当然是选择最大长度的链上的那条边，因因为两条边删除的代价都是1，当然使最长链消失更好。这个自己细加思考相信一定能理解。

参考标程

```
#include <bits/stdc++.h>

using namespace std;

const int N = 1000010;

struct edge {
    int y, nex, c;
} s[N << 1];

int  first[N], len, n, m;
int  ans = 0;
void read(int& x)
{
    char ch = getchar();
    x       = 0;
    while (ch < '0' || ch > '9') ch = getchar();
    while (ch >= '0' && ch <= '9') x = x * 10 + ch - '0', ch = getchar();
}
void ins(int x, int y, int c)
{
    s[++len] = (edge){y, first[x], c};
    first[x] = len;
}
```

```cpp
int dfs(int x, int fa)
{
    vector<int> V;
    for (int i = first[x]; i != 0; i = s[i].nex)
        if (s[i].y != fa) V.push_back(dfs(s[i].y, x) + s[i].c);
    sort(V.begin(), V.end());

    for (int i = V.size() - 1; i >= 0; i--) {
        if (V[i] > m || i && V[i] > m - V[i - 1])
            ans++;
        else
            return V[i];
    }
    return 0;
}
int main()
{
    freopen("C.in", "r", stdin);
    freopen("C.out", "w", stdout);
    read(n);
    read(m);
    int x, y, c;
    for (int i = 1; i < n; i++) read(x), read(y), read(c), ins(x, y, c), ins(y, x, c);
    dfs(1, 0);
    printf("%d", ans);
}
```

D 题：三元组

三元组

输入文件名：triple.in
输出文件名：triple.out
共 20 个测试点，每个测试点 5 分
每个测试点限时 1 秒，运行内存上限 512MB

“阿巴巴巴，你以为你很牛啊？”**老 Win** 看着被你切飞的第一题，暗暗自喃道。

老 Win 随便灵机一动，想到了另一道神仙题。

对于三元函数 $f(x,y,z)$，当 $z=0$ 时，有 $f(x,y,z)=1$，当 $z\geq 1$ 时，有 $f(x,y,z)=\sum_{x_1=1}^{x}\sum_{y_1=1}^{y}(x-x_1+1)(y-y_1+1)f(x_1,y_1,z-1)$

现在给出 N,M,K，**老 Win** 想让你告诉他 $\sum_{n=1}^{N}\sum_{m=1}^{M}f(n,m,K)$

由于 **老 Win** 最近很讨厌高精度算法，所以他打算直接叫你对 998244353 取模，以减轻**蒟蒻**出题人的烦恼。

输入格式

多组数据。

第一行一个正整数 T 表示数据组数。

接下来 T 行每行三个正整数表示 N,M,K。

输出格式

每组数据输出一行一个整数表示答案。

数据范围

对于 20% 的数据，满足 $N,M,K\leq 30$

对于 40% 的数据，满足 $N,M,K\leq 1000$

对于 60% 的数据，满足 $N,M,K\leq 100000$

对于 80% 的数据，满足 $N,M,K\leq 10000000$

对于 100% 的数据，满足 $N,M,K\leq 10^{18},T\leq 20$

样例输入

```
2
2 3 5
342 234 532
```

样例输出

```
936
373362092
```

题解

两道组合数学希望不要被骂。

如果第一步转移错了后面会做的很麻烦，但还是可以用生成函数爆算得到的。

这个转移式子相当于就是在边长为n, m的矩形内，选k次子矩形的方案数。

那么我们枚举总共选了x行和y列，那么就相当于将$2k$个没有标号的球放在x个盒子里，每个盒子必有球的方案数。

用隔板法简单解决。

$$\sum_{x=1}^{n}\sum_{y=1}^{m}C(n,x)C(m,y)C(2k-1,x-1)C(2k-1,y-1)$$

发现这两个是没有什么关系的，可以解决其中一个。

$$\sum_{x=1}^{n}C(n,x)C(2k-1,x-1)$$

$$=\sum_{x=1}^{n}C(n,n-x)C(2k-1,x-1)$$

$$=C(n+2k-1,n-1)$$

答案就是：

$$\sum_{n=1}^{N}\sum_{m=1}^{M}C(n+2K-1,2K)C(m+2K-1,2K)$$

$$=C(N+2K,2K+1)C(M+2K,2K+1)$$

首先套用Lucas定理，将每一项都化简成$< mod$

发现要处理一个大数阶乘，可以使用快速阶乘算法来踩爆标程。（也许不能

也可以预处理$(A*1000000)!$，然后每次$O(1000000)$算组合数。

参考标程

```
#include <bits/stdc++.h>
using namespace std;
```

```
int pre[1000010] = {
    1,         373341033, 45596018,  834980587, 623627864, 428937595, 442819817, 499710224, 833655840, 83857087,
    295201906, 788488293, 671639287, 849315549, 597398273, 813259672, 732727656, 244038325, 122642896, 310517972,
    160030060, 483239722, 683879839, 712910418, 384710263, 433880730, 844360005, 513089677, 101492974, 959253371,
    957629942, 678615452, 34035221,  56734233,  524027922, 31729117,  102311167, 330331487, 8332991,   832392662,
    545208507, 594075875, 318497156, 859275605, 300738984, 767818091, 864118508, 878131539, 316588744, 812496962,
    213689172, 584871249, 980836133, 54096741,  417876813, 363266670, 335481797, 730839588, 393495668, 435793297,
    760025067, 811438469, 720976283, 650770098, 586537547, 117371703, 566486504, 749562308, 708205284, 932912293,
    939830261, 983699513, 206579820, 301188781, 593164676, 770845925, 247687458, 41047791,  266419267, 937835947,
    506268060, 6177705,   936268003, 166873118, 443834893, 328979964, 470135404, 954410105, 117565665, 832761782,
    39806322,  478922755, 394880724, 821825588, 468705875, 512554988, 232240472, 876497899, 356048018, 895187265,
    808258749, 575505950, 68190615,  939065335, 552199946, 694814243, 385460530, 529769387, 640377761, 916128300,
    440133909, 362216114, 826373774, 502324157, 457648395, 385510728, 904737188, 78988746,  454565719, 623828097,
    686156489, 713476044, 63602402,  570334625, 681055904, 222059821, 477211096, 343363294, 833792655, 461853093,
    741797144, 74731896,  930484262, 268372735, 941222802, 677432735, 474842829, 700451655, 400176109, 697644778,
    390377694, 790010794, 360642718, 505712943, 946647976, 339045014, 715797300, 251680896, 70091750,  40517433,
    12629586,  850635539, 110877109, 571935891, 695965747, 634938288, 69072133,  155093216, 749696762, 963086402,
    544711799, 724471925, 334646013, 574791029, 722417626, 377929821, 743946412, 988034679, 405207112, 18063742,
    104121967, 638607426, 607304611, 751377777, 35834555,  313632531, 18058363,  656121134, 40763559,  562910912,
    495867250, 48767038,  210864657, 659137294, 715390025, 865854329, 324322857, 388911184, 286059202, 636456178,
    421290700, 832276048, 726437551, 526417714, 252522639, 386147469, 674313019, 274769381, 226519400, 272047186,
    117153405, 712896591, 486826649, 119444874, 338909703, 18536028,  41814114,  245606459, 140617938, 250512392,
    57084755,  157807456, 261113192, 40258068,  194807105, 325341339, 884328111, 896332013, 880836012, 737358206,
    202713771, 785454372, 399586250, 485457499, 640827004, 546969497, 749602473, 159788463, 159111724, 218592929,
    675932866, 314795475, 811539323, 246883213, 696818315, 759880589, 4302336,   353070689, 477909706, 559289160,
    79781699,  878094972, 840903973, 367416824, 973366814, 848259019, 462421750, 667227759, 897917455, 81800722,
    956276337, 942686845, 420541799, 417005912, 272641764, 941778993, 217214373, 192220616, 267901132, 50530621,
    652678397, 354880856, 164289049, 781023184, 105376215, 315094878, 607856504, 733905911, 457743498, 992735713,
    35212756,  231822660, 276036750, 734558079, 424180850, 433186147, 308380947, 18333316,  12935086,  351491725,
    655645460, 535812389, 521902115, 67016984,  48682076,  64748124,  489360447, 361275315, 786336279, 805161272,
    468129309, 645091350, 887284732, 913004502, 358814684, 281295633, 328970139, 395955130, 164840186, 820902807,
```

```
761699708, 246274415, 592331769, 913846362, 866682684, 600130702, 903837674, 529462989, 90612675,  526540127,
533047427, 110008879, 674279751, 801920753, 645226926, 676886948, 752481486, 474034007, 457790341, 166813684,
287671032, 188118664, 244731384, 404032157, 269766986, 423996017, 182948540, 356801634, 737863144, 652014069,
206068022, 504569410, 919894484, 593398649, 963768176, 882517476, 702523597, 949028249, 128957299, 171997372,
50865043,  20937461,  690959202, 581356488, 369182214, 993580422, 193500140, 540665426, 365786018, 743731625,
144980423, 979536721, 773259009, 617053935, 247670131, 843705280, 30419459,  985463402, 261585206, 237885042,
111276893, 488166208, 137660292, 720784236, 244467770, 26368504,  792857103, 666885724, 670313309, 905683034,
259415897, 512017253, 826265493, 111960112, 633652060, 918048438, 516432938, 386972415, 996212724, 610073831,
444094191, 72480267,  665038087, 11584804,  301029012, 723617861, 113763819, 778259899, 937766095, 535448641,
593907889, 783573565, 673298635, 599533244, 655712590, 173350007, 868198597, 169013813, 585161712, 697502214,
573994984, 285943986, 675831407, 3134056,   965907646, 401920943, 665949756, 236277883, 612745912, 813282113,
892454686, 901222267, 624900982, 927122298, 686321335, 84924870,  927606072, 506664166, 353631992, 165913238,
566073550, 816674343, 864877926, 171259407, 908752311, 874007723, 803597299, 613676466, 880336545, 282280109,
128761001, 58852065,  474075900, 434816091, 364856903, 149123648, 388854780, 314693916, 423183826, 419733481,
888483202, 238933227, 336564048, 757103493, 100189123, 855479832, 51370348,  403061033, 496971759, 831753030,
251718753, 272779384, 683379259, 488844621, 881783783, 659478190, 445719559, 740782647, 546525906, 985524427,
548033568, 333772553, 331916427, 752533273, 730387628, 93829695,  655989476, 930661318, 334885743, 466041862,
428105027, 888238707, 232218076, 769865249, 730641039, 616996159, 231721356, 326973501, 426068899, 722403656,
742756734, 663270261, 364187931, 350431704, 671823672, 633125919, 226166717, 386814657, 237594135, 451479365,
546182474, 119366536, 465211069, 605313606, 728508871, 249619035, 663053607, 900453742, 48293872,  229958401,
62402409,  69570431,  71921532,  960467929, 537087913, 514588945, 513856225, 415497414, 286592050, 645469437,
102052166, 163298189, 873938719, 617583886, 986843080, 962390239, 580971332, 665147020, 88900164,  89866970,
826426395, 616059995, 443012312, 659160562, 229855967, 687413213, 59809521,  398599610, 325666688, 154765991,
159186619, 210830877, 386454418, 84493735,  974220646, 820097297, 2191828,   481459931, 729073424, 551556379,
926316039, 151357011, 808637654, 218058015, 786112034, 850407126, 84202800,  94214098,  30019651,  121701603,
176055335, 865461951, 553631971, 286620803, 984061713, 888573766, 302767023, 977070668, 110954576, 83922475,
51568171,  60949367,  19533020,  510592752, 615419476, 341370469, 912573425, 286207526, 206707897, 384156962,
414163604, 193301813, 749570167, 366933789, 11470970,  600191572, 391667731, 328736286, 30645366,  215162519,
604947226, 236199953, 718439098, 411423177, 803407599, 632441623, 766760224, 263006576, 757681534, 61082578,
681666415, 947466395, 12206799,  659767098, 933746852, 978860867, 59215985,  161179205, 439197472, 259779111,
511621808, 145770512, 882749888, 943124465, 872053396, 631078482, 166861622, 743415395, 772287179, 602427948,
924112080, 385643091, 794973480, 883782693, 869723371, 805963889, 313106351, 262132854, 400034567, 488248149,
```

```
265769800, 791715397, 408753255, 468381897, 415812467, 172922144, 64404368,  281500398, 512318142, 288791777,
955559118, 242484726, 536413695, 205340854, 707803527, 576699812, 218525078, 875554190, 46283078,  833841915,
763148293, 807722138, 788080170, 556901372, 150896699, 253151120, 97856807,  918256774, 771557187, 582547026,
472709375, 911615063, 743371401, 641382840, 446540967, 184639537, 157247760, 775930891, 939702814, 499082462,
19536133,  548753627, 593243221, 563850263, 185475971, 687419227, 396799323, 657976136, 864535682, 433009242,
860830935, 33107339,  517661450, 467651311, 812398757, 202133852, 431839017, 709549400, 99643620,  773282878,
290471030, 61134552,  129206504, 929147251, 837008968, 422332597, 353775281, 469563025, 62265336,  835064501,
851685235, 21197005,  264793769, 326416680, 118842991, 84257200,  763248924, 687559609, 150907932, 401832452,
242726978, 766752066, 959173604, 390269102, 992293822, 744816299, 476631694, 177284763, 702429415, 374065901,
169855231, 629007616, 719169602, 564737074, 475119050, 714502830, 40993711,  820235888, 749063595, 239329111,
612759169, 18591377,  419142436, 442202439, 941600951, 158013406, 637073231, 471564060, 447222237, 701248503,
599797734, 577221870, 69656699,  51052704,  6544303,   10958310,  554955500, 943192237, 192526269, 897983911,
961628039, 240232720, 627280533, 710239542, 70255649,  261743865, 228474833, 776408079, 304180483, 63607040,
953297493, 758058902, 395529997, 156010331, 825833840, 539880795, 234683685, 52626619,  751843490, 116909119,
62806842,  574857555, 353417551, 40061330,  822203768, 681051568, 490913702, 9322961,   766631257, 124794668,
37844313,  163524507, 729108319, 490867505, 47035168,  682765157, 53842115,  817965276, 757179922, 339238384,
909741023, 150530547, 158444563, 140949492, 993302799, 551621442, 137578883, 475122706, 443869843, 605400098,
689361523, 769596520, 801661499, 474900284, 586624857, 349960501, 134084537, 650564083, 877097974, 379857427,
887890124, 159436401, 133274277, 986182139, 729720334, 568925901, 459461496, 499309445, 493171177, 460958750,
380694152, 168836226, 840160881, 141116880, 225064950, 109618190, 842341383, 85305729,  759273275, 97369807,
669317759, 766247510, 829017039, 550323884, 261274540, 918239352, 29606025,  870793828, 293683814, 378510746,
367270918, 481292028, 813097823, 798448487, 230791733, 899305835, 504040630, 162510533, 479367951, 275282274,
806951470, 462774647, 56473153,  184659008, 905122161, 664034750, 109726629, 59372704,  325795100, 486860143,
843736533, 924723613, 880348000, 801252478, 616515290, 776142608, 284803450, 583439582, 274826676, 6018349,
377403437, 244041569, 527081707, 544763288, 708818585, 354033051, 904309832, 589922898, 673933870, 682858433,
945260111, 899893421, 515264973, 911685911, 9527148,   239480646, 524126897, 48259065,  578214879, 118677219,
786127243, 869205770, 923276513, 937928886, 802186160, 12198440,  638784295, 34200904,  758925811, 185027790,
80918046,  120604699, 610456697, 573601211, 208296321, 49743354,  653691911, 490750754, 674335312, 887877110,
875880304, 308360096, 414636410, 886100267, 8525751,   636257427, 558338775, 500159951, 696213291, 97268896,
364983542, 937928436, 641582714, 586211304, 345265657, 994704486, 443549763, 207259440, 302122082, 166055224,
623250998, 239642551, 476337075, 283167364, 211328914, 68064804,  950202136, 187552679, 18938709,  646784245,
598764068, 538505481, 610424991, 864445053, 390248689, 278395191, 686098470, 935957187, 868529577, 329970687,
```

```
804930040, 84992079,  474569269, 810762228, 573258936, 756464212, 155080225, 286966169, 283614605, 19283401,
24257676,  871831819, 612689791, 846988741, 617120754, 971716517, 979541482, 297910784, 991087897, 783825907,
214821357, 689498189, 405026419, 946731704, 609346370, 707669156, 457703127, 957341187, 980735523, 649367684,
791011898, 82098966,  234729712, 105002711, 130614285, 291032164, 193188049, 363211260, 58108651,  100756444,
954947696, 346032213, 863300806, 36876722,  622610957, 289232396, 667938985, 734886266, 395881057, 417188702,
183092975, 887586469, 83334648,  797819763, 100176902, 781587414, 841864935, 371674670, 18247584};
```

```cpp
const int mod = 998244353;

int       ksm(int x, int t)
{
    int tot = 1;
    while (t) {
        if (t & 1) tot = 1ll * tot * x % mod;
        x = 1ll * x * x % mod;
        t /= 2;
    }
    return tot;
}

int C(long long x, long long y)
{
    int ans = 1;
    if (x >= mod || y >= mod) ans = C(x / mod, y / mod);
    x %= mod;
    y %= mod;
    if (x < y) return 0;
    int z = x - y, s1 = pre[x / 1000000], s2 = pre[y / 1000000], s3 = pre[z / 1000000];
    for (int j = x / 1000000 * 1000000 + 1; j <= x; j++) s1 = 1ll * s1 * j % mod;
    for (int j = y / 1000000 * 1000000 + 1; j <= y; j++) s2 = 1ll * s2 * j % mod;
    for (int j = z / 1000000 * 1000000 + 1; j <= z; j++) s3 = 1ll * s3 * j % mod;
    return 1ll * ans * s1 % mod * ksm(s2, mod - 2) % mod * ksm(s3, mod - 2) % mod;
}

int main()
{
    freopen("D.in", "r", stdin);
    freopen("D.out", "w", stdout);
    int T;
    scanf("%d", &T);
    long long n, m, k;
    while (T--) {
        scanf("%lld %lld %lld", &n, &m, &k);
        printf("%lld\n", 1ll * C(n + 2 * k, 2 * k + 1) * C(m + 2 * k, 2 * k + 1) % mod);
    }
}
```