Олимпиада СПбГУ по информатике 2021/22 учебного года

A	B	C	D	E	F	Sum
100	100	100	100	58	0	458

Task A ()

```cpp
#include <iostream>
#include <map>
using namespace std;

int main()
{
    ios_base::sync_with_stdio(0);
    cin.tie(0);

    int n, k, b;
    long long a;
    map <int, long long> m;

    cin >> n;
    for (int i = 0; i < n; i++) {
        cin >> a >> b;
        while (!(a % 10)) {
            a /= 10;
            b++;
        }
        m[b] += a;
    }
    map <int, long long> :: iterator it;
    int t = -1;
    for (it = m.begin(); it != m.end(); it++) {
//        cout << it->first << ' ' << it->second << '\n';
        b = it->first;
        a = it->second;
        if (a % 10) {
            cout << b << '\n';
            return 0;
        }
        while (!(a % 10)) {
            a /= 10;
            b++;
        }
        m[b] += a;
    }

    return 0;
}
```

Task B ()

```cpp
#include <iostream>
using namespace std;

int main()
{
    int n, m, k[110];
    string s;

    cin >> n;
    for (int i = 0; i < n; i++)
        cin >> k[i];
    for (int i = n - 1; i >= 0; i--) {
        for (int j = 0; j < k[i]; j++) {
            m = 0;
            cout << "Flip and wait" << endl;
            while (m < i + 1) {
                cin >> s;
                m += (int(s.size()) - 3) / 2;
                if (m < i + 1)
                    cout << "Wait" << endl;
            }
        }
    }
    cout << "Stop" << endl;

    return 0;
}
```

Task C ()

```cpp
#include <iostream>
using namespace std;

int main()
{
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);

    int n, k, t;
    string s;

    cin >> t;
    while (t--) {
        cin >> s;
        bool r = 0;
        for (int i = 0; i < s.size(); i++)
            if (s[i] == '?')
            r = 1;
        int cnt_0 = 0, cnt_1 = 0;
        for (int i = 0; i < s.size(); i++) {
            if (s[i] == '0')
                cnt_0++;
            else if (s[i] == '1')
                cnt_1++;
        }
        if (!r) {
            if (cnt_1 > cnt_0) {
                for (int i = 0; i < s.size(); i++)
                    if (s[i] == '1')
                        s[i] = '?';
            }
            else if (cnt_0 == (int)s.size()) {
                for (int i = 0; i < (int)s.size() / 2; i++)
                    s[i] = '?';
            }
            else {
                for (int i = 0; i < s.size(); i++)
                    if (s[i] == '0')
                        s[i] = '?';
            }
            cout << s << endl;
        }
        else {
            if (!cnt_0 && !cnt_1) {
                for (int i = 0; i < s.size(); i++)
                    s[i] = '1';
            }
            else if (cnt_0 == ((int)s.size() + 1) / 2) {
                for (int i = 0; i < s.size(); i++)
                    s[i] = '0';
            }
            else if (cnt_0) {
                for (int i = 0; i < s.size(); i++)
                    if (s[i] == '?')
                        s[i] = '1';
            }
            else {
                for (int i = 0; i < s.size(); i++)
                    if (s[i] == '?')
                        s[i] = '0';
            }
            cout << s << endl;
        }
    }

    return 0;
}
```

Task D ()

```cpp
#include <iostream>
using namespace std;

long long mod = 998244353;

int poww (int a, int m) {
    int k = 1;
    while (m) {
        if (m % 2)
            k = 1ll * k * a % mod;
        a = 1ll * a * a % mod;
        m /= 2;
    }
    return k;
}

int main()
{
    long long n, m = 1, k, f = 998244352, a[1010];

    a[0] = 1;
    a[1] = 373341033;
    a[2] = 45596018;
    a[3] = 834980587;
    a[4] = 623627864;
    a[5] = 428937595;
    a[6] = 442819817;
    a[7] = 499710224;
    a[8] = 833655840;
    a[9] = 83857087;
    a[10] = 295201906;
    a[11] = 788488293;
    a[12] = 671639287;
    a[13] = 849315549;
    a[14] = 597398273;
    a[15] = 813259672;
    a[16] = 732727656;
    a[17] = 244038325;
    a[18] = 122642896;
    a[19] = 310517972;
    a[20] = 160030060;
    a[21] = 483239722;
    a[22] = 683879839;
    a[23] = 712910418;
    a[24] = 384710263;
    a[25] = 433880730;
    a[26] = 844360005;
    a[27] = 513089677;
    a[28] = 101492974;
    a[29] = 959253371;
    a[30] = 957629942;
    a[31] = 678615452;
    a[32] = 34035221;
    a[33] = 56734233;
    a[34] = 524027922;
    a[35] = 31729117;
    a[36] = 102311167;
    a[37] = 330331487;
    a[38] = 8332991;
    a[39] = 832392662;
    a[40] = 545208507;
    a[41] = 594075875;
    a[42] = 318497156;
    a[43] = 859275605;
    a[44] = 300738984;
    a[45] = 767818091;
    a[46] = 864118508;
    a[47] = 878131539;
    a[48] = 316588744;
    a[49] = 812496962;
    a[50] = 213689172;
    a[51] = 584871249;
```

```
a[52] = 980836133;
a[53] = 54096741;
a[54] = 417876813;
a[55] = 363266670;
a[56] = 335481797;
a[57] = 730839588;
a[58] = 393495668;
a[59] = 435793297;
a[60] = 760025067;
a[61] = 811438469;
a[62] = 720976283;
a[63] = 650770098;
a[64] = 586537547;
a[65] = 117371703;
a[66] = 566486504;
a[67] = 749562308;
a[68] = 708205284;
a[69] = 932912293;
a[70] = 939830261;
a[71] = 983699513;
a[72] = 206579820;
a[73] = 301188781;
a[74] = 593164676;
a[75] = 770845925;
a[76] = 247687458;
a[77] = 41047791;
a[78] = 266419267;
a[79] = 937835947;
a[80] = 506268060;
a[81] = 6177705;
a[82] = 936268003;
a[83] = 166873118;
a[84] = 443834893;
a[85] = 328979964;
a[86] = 470135404;
a[87] = 954410105;
a[88] = 117565665;
a[89] = 832761782;
a[90] = 39806322;
a[91] = 478922755;
a[92] = 394880724;
a[93] = 821825588;
a[94] = 468705875;
a[95] = 512554988;
a[96] = 232240472;
a[97] = 876497899;
a[98] = 356048018;
a[99] = 895187265;
a[100] = 808258749;
a[101] = 575505950;
a[102] = 68190615;
a[103] = 939065335;
a[104] = 552199946;
a[105] = 694814243;
a[106] = 385460530;
a[107] = 529769387;
a[108] = 640377761;
a[109] = 916128300;
a[110] = 440133909;
a[111] = 362216114;
a[112] = 826373774;
a[113] = 502324157;
a[114] = 457648395;
a[115] = 385510728;
a[116] = 904737188;
a[117] = 78988746;
a[118] = 454565719;
a[119] = 623828097;
a[120] = 686156489;
a[121] = 713476044;
a[122] = 63602402;
a[123] = 570334625;
a[124] = 681055904;
a[125] = 222059821;
```

```
a[126] = 477211096;
a[127] = 343363294;
a[128] = 833792655;
a[129] = 461853093;
a[130] = 741797144;
a[131] = 74731896;
a[132] = 930484262;
a[133] = 268372735;
a[134] = 941222802;
a[135] = 677432735;
a[136] = 474842829;
a[137] = 700451655;
a[138] = 400176109;
a[139] = 697644778;
a[140] = 390377694;
a[141] = 790010794;
a[142] = 360642718;
a[143] = 505712943;
a[144] = 946647976;
a[145] = 339045014;
a[146] = 715797300;
a[147] = 251680896;
a[148] = 70091750;
a[149] = 40517433;
a[150] = 12629586;
a[151] = 850635539;
a[152] = 110877109;
a[153] = 571935891;
a[154] = 695965747;
a[155] = 634938288;
a[156] = 69072133;
a[157] = 155093216;
a[158] = 749696762;
a[159] = 963086402;
a[160] = 544711799;
a[161] = 724471925;
a[162] = 334646013;
a[163] = 574791029;
a[164] = 722417626;
a[165] = 377929821;
a[166] = 743946412;
a[167] = 988034679;
a[168] = 405207112;
a[169] = 18063742;
a[170] = 104121967;
a[171] = 638607426;
a[172] = 607304611;
a[173] = 751377777;
a[174] = 35834555;
a[175] = 313632531;
a[176] = 18058363;
a[177] = 656121134;
a[178] = 40763559;
a[179] = 562910912;
a[180] = 495867250;
a[181] = 48767038;
a[182] = 210864657;
a[183] = 659137294;
a[184] = 715390025;
a[185] = 865854329;
a[186] = 324322857;
a[187] = 388911184;
a[188] = 286059202;
a[189] = 636456178;
a[190] = 421290700;
a[191] = 832276048;
a[192] = 726437551;
a[193] = 526417714;
a[194] = 252522639;
a[195] = 386147469;
a[196] = 674313019;
a[197] = 274769381;
a[198] = 226519400;
a[199] = 272047186;
```

```
a[200] = 117153405;
a[201] = 712896591;
a[202] = 486826649;
a[203] = 119444874;
a[204] = 338909703;
a[205] = 18536028;
a[206] = 41814114;
a[207] = 245606459;
a[208] = 140617938;
a[209] = 250512392;
a[210] = 57084755;
a[211] = 157807456;
a[212] = 261113192;
a[213] = 40258068;
a[214] = 194807105;
a[215] = 325341339;
a[216] = 884328111;
a[217] = 896332013;
a[218] = 880836012;
a[219] = 737358206;
a[220] = 202713771;
a[221] = 785454372;
a[222] = 399586250;
a[223] = 485457499;
a[224] = 640827004;
a[225] = 546969497;
a[226] = 749602473;
a[227] = 159788463;
a[228] = 159111724;
a[229] = 218592929;
a[230] = 675932866;
a[231] = 314795475;
a[232] = 811539323;
a[233] = 246883213;
a[234] = 696818315;
a[235] = 759880589;
a[236] = 4302336;
a[237] = 353070689;
a[238] = 477909706;
a[239] = 559289160;
a[240] = 79781699;
a[241] = 878094972;
a[242] = 840903973;
a[243] = 367416824;
a[244] = 973366814;
a[245] = 848259019;
a[246] = 462421750;
a[247] = 667227759;
a[248] = 897917455;
a[249] = 81800722;
a[250] = 956276337;
a[251] = 942686845;
a[252] = 420541799;
a[253] = 417005912;
a[254] = 272641764;
a[255] = 941778993;
a[256] = 217214373;
a[257] = 192220616;
a[258] = 267901132;
a[259] = 50530621;
a[260] = 652678397;
a[261] = 354880856;
a[262] = 164289049;
a[263] = 781023184;
a[264] = 105376215;
a[265] = 315094878;
a[266] = 607856504;
a[267] = 733905911;
a[268] = 457743498;
a[269] = 992735713;
a[270] = 35212756;
a[271] = 231822660;
a[272] = 276036750;
a[273] = 734558079;
```

```
a[274] = 424180850;
a[275] = 433186147;
a[276] = 308380947;
a[277] = 18333316;
a[278] = 12935086;
a[279] = 351491725;
a[280] = 655645460;
a[281] = 535812389;
a[282] = 521902115;
a[283] = 67016984;
a[284] = 48682076;
a[285] = 64748124;
a[286] = 489360447;
a[287] = 361275315;
a[288] = 786336279;
a[289] = 805161272;
a[290] = 468129309;
a[291] = 645091350;
a[292] = 887284732;
a[293] = 913004502;
a[294] = 358814684;
a[295] = 281295633;
a[296] = 328970139;
a[297] = 395955130;
a[298] = 164840186;
a[299] = 820902807;
a[300] = 761699708;
a[301] = 246274415;
a[302] = 592331769;
a[303] = 913846362;
a[304] = 866682684;
a[305] = 600130702;
a[306] = 903837674;
a[307] = 529462989;
a[308] = 90612675;
a[309] = 526540127;
a[310] = 533047427;
a[311] = 110008879;
a[312] = 674279751;
a[313] = 801920753;
a[314] = 645226926;
a[315] = 676886948;
a[316] = 752481486;
a[317] = 474034007;
a[318] = 457790341;
a[319] = 166813684;
a[320] = 287671032;
a[321] = 188118664;
a[322] = 244731384;
a[323] = 404032157;
a[324] = 269766986;
a[325] = 423996017;
a[326] = 182948540;
a[327] = 356801634;
a[328] = 737863144;
a[329] = 652014069;
a[330] = 206068022;
a[331] = 504569410;
a[332] = 919894484;
a[333] = 593398649;
a[334] = 963768176;
a[335] = 882517476;
a[336] = 702523597;
a[337] = 949028249;
a[338] = 128957299;
a[339] = 171997372;
a[340] = 50865043;
a[341] = 20937461;
a[342] = 690959202;
a[343] = 581356488;
a[344] = 369182214;
a[345] = 993580422;
a[346] = 193500140;
a[347] = 540665426;
```

```
a[348] = 365786018;
a[349] = 743731625;
a[350] = 144980423;
a[351] = 979536721;
a[352] = 773259009;
a[353] = 617053935;
a[354] = 247670131;
a[355] = 843705280;
a[356] = 30419459;
a[357] = 985463402;
a[358] = 261585206;
a[359] = 237885042;
a[360] = 111276893;
a[361] = 488166208;
a[362] = 137660292;
a[363] = 720784236;
a[364] = 244467770;
a[365] = 26368504;
a[366] = 792857103;
a[367] = 666885724;
a[368] = 670313309;
a[369] = 905683034;
a[370] = 259415897;
a[371] = 512017253;
a[372] = 826265493;
a[373] = 111960112;
a[374] = 633652060;
a[375] = 918048438;
a[376] = 516432938;
a[377] = 386972415;
a[378] = 996212724;
a[379] = 610073831;
a[380] = 444094191;
a[381] = 72480267;
a[382] = 665038087;
a[383] = 11584804;
a[384] = 301029012;
a[385] = 723617861;
a[386] = 113763819;
a[387] = 778259899;
a[388] = 937766095;
a[389] = 535448641;
a[390] = 593907889;
a[391] = 783573565;
a[392] = 673298635;
a[393] = 599533244;
a[394] = 655712590;
a[395] = 173350007;
a[396] = 868198597;
a[397] = 169013813;
a[398] = 585161712;
a[399] = 697502214;
a[400] = 573994984;
a[401] = 285943986;
a[402] = 675831407;
a[403] = 3134056;
a[404] = 965907646;
a[405] = 401920943;
a[406] = 665949756;
a[407] = 236277883;
a[408] = 612745912;
a[409] = 813282113;
a[410] = 892454686;
a[411] = 901222267;
a[412] = 624900982;
a[413] = 927122298;
a[414] = 686321335;
a[415] = 84924870;
a[416] = 927606072;
a[417] = 506664166;
a[418] = 353631992;
a[419] = 165913238;
a[420] = 566073550;
a[421] = 816674343;
```

```
a[422] = 864877926;
a[423] = 171259407;
a[424] = 908752311;
a[425] = 874007723;
a[426] = 803597299;
a[427] = 613676466;
a[428] = 880336545;
a[429] = 282280109;
a[430] = 128761001;
a[431] = 58852065;
a[432] = 474075900;
a[433] = 434816091;
a[434] = 364856903;
a[435] = 149123648;
a[436] = 388854780;
a[437] = 314693916;
a[438] = 423183826;
a[439] = 419733481;
a[440] = 888483202;
a[441] = 238933227;
a[442] = 336564048;
a[443] = 757103493;
a[444] = 100189123;
a[445] = 855479832;
a[446] = 51370348;
a[447] = 403061033;
a[448] = 496971759;
a[449] = 831753030;
a[450] = 251718753;
a[451] = 272779384;
a[452] = 683379259;
a[453] = 488844621;
a[454] = 881783783;
a[455] = 659478190;
a[456] = 445719559;
a[457] = 740782647;
a[458] = 546525906;
a[459] = 985524427;
a[460] = 548033568;
a[461] = 333772553;
a[462] = 331916427;
a[463] = 752533273;
a[464] = 730387628;
a[465] = 93829695;
a[466] = 655989476;
a[467] = 930661318;
a[468] = 334885743;
a[469] = 466041862;
a[470] = 428105027;
a[471] = 888238707;
a[472] = 232218076;
a[473] = 769865249;
a[474] = 730641039;
a[475] = 616996159;
a[476] = 231721356;
a[477] = 326973501;
a[478] = 426068899;
a[479] = 722403656;
a[480] = 742756734;
a[481] = 663270261;
a[482] = 364187931;
a[483] = 350431704;
a[484] = 671823672;
a[485] = 633125919;
a[486] = 226166717;
a[487] = 386814657;
a[488] = 237594135;
a[489] = 451479365;
a[490] = 546182474;
a[491] = 119366536;
a[492] = 465211069;
a[493] = 605313606;
a[494] = 728508871;
a[495] = 249619035;
```

```
a[496] = 663053607;
a[497] = 900453742;
a[498] = 48293872;
a[499] = 229958401;
a[500] = 62402409;
a[501] = 69570431;
a[502] = 71921532;
a[503] = 960467929;
a[504] = 537087913;
a[505] = 514588945;
a[506] = 513856225;
a[507] = 415497414;
a[508] = 286592050;
a[509] = 645469437;
a[510] = 102052166;
a[511] = 163298189;
a[512] = 873938719;
a[513] = 617583886;
a[514] = 986843080;
a[515] = 962390239;
a[516] = 580971332;
a[517] = 665147020;
a[518] = 88900164;
a[519] = 89866970;
a[520] = 826426395;
a[521] = 616059995;
a[522] = 443012312;
a[523] = 659160562;
a[524] = 229855967;
a[525] = 687413213;
a[526] = 59809521;
a[527] = 398599610;
a[528] = 325666688;
a[529] = 154765991;
a[530] = 159186619;
a[531] = 210830877;
a[532] = 386454418;
a[533] = 84493735;
a[534] = 974220646;
a[535] = 820097297;
a[536] = 2191828;
a[537] = 481459931;
a[538] = 729073424;
a[539] = 551556379;
a[540] = 926316039;
a[541] = 151357011;
a[542] = 808637654;
a[543] = 218058015;
a[544] = 786112034;
a[545] = 850407126;
a[546] = 84202800;
a[547] = 94214098;
a[548] = 30019651;
a[549] = 121701603;
a[550] = 176055335;
a[551] = 865461951;
a[552] = 553631971;
a[553] = 286620803;
a[554] = 984061713;
a[555] = 888573766;
a[556] = 302767023;
a[557] = 977070668;
a[558] = 110954576;
a[559] = 83922475;
a[560] = 51568171;
a[561] = 60949367;
a[562] = 19533020;
a[563] = 510592752;
a[564] = 615419476;
a[565] = 341370469;
a[566] = 912573425;
a[567] = 286207526;
a[568] = 206707897;
a[569] = 384156962;
```

```
a[570] = 414163604;
a[571] = 193301813;
a[572] = 749570167;
a[573] = 366933789;
a[574] = 11470970;
a[575] = 600191572;
a[576] = 391667731;
a[577] = 328736286;
a[578] = 30645366;
a[579] = 215162519;
a[580] = 604947226;
a[581] = 236199953;
a[582] = 718439098;
a[583] = 411423177;
a[584] = 803407599;
a[585] = 632441623;
a[586] = 766760224;
a[587] = 263006576;
a[588] = 757681534;
a[589] = 61082578;
a[590] = 681666415;
a[591] = 947466395;
a[592] = 12206799;
a[593] = 659767098;
a[594] = 933746852;
a[595] = 978860867;
a[596] = 59215985;
a[597] = 161179205;
a[598] = 439197472;
a[599] = 259779111;
a[600] = 511621808;
a[601] = 145770512;
a[602] = 882749888;
a[603] = 943124465;
a[604] = 872053396;
a[605] = 631078482;
a[606] = 166861622;
a[607] = 743415395;
a[608] = 772287179;
a[609] = 602427948;
a[610] = 924112080;
a[611] = 385643091;
a[612] = 794973480;
a[613] = 883782693;
a[614] = 869723371;
a[615] = 805963889;
a[616] = 313106351;
a[617] = 262132854;
a[618] = 400034567;
a[619] = 488248149;
a[620] = 265769800;
a[621] = 791715397;
a[622] = 408753255;
a[623] = 468381897;
a[624] = 415812467;
a[625] = 172922144;
a[626] = 64404368;
a[627] = 281500398;
a[628] = 512318142;
a[629] = 288791777;
a[630] = 955559118;
a[631] = 242484726;
a[632] = 536413695;
a[633] = 205340854;
a[634] = 707803527;
a[635] = 576699812;
a[636] = 218525078;
a[637] = 875554190;
a[638] = 46283078;
a[639] = 833841915;
a[640] = 763148293;
a[641] = 807722138;
a[642] = 788080170;
a[643] = 556901372;
```

```
a[644] = 150896699;
a[645] = 253151120;
a[646] = 97856807;
a[647] = 918256774;
a[648] = 771557187;
a[649] = 582547026;
a[650] = 472709375;
a[651] = 911615063;
a[652] = 743371401;
a[653] = 641382840;
a[654] = 446540967;
a[655] = 184639537;
a[656] = 157247760;
a[657] = 775930891;
a[658] = 939702814;
a[659] = 499082462;
a[660] = 19536133;
a[661] = 548753627;
a[662] = 593243221;
a[663] = 563850263;
a[664] = 185475971;
a[665] = 687419227;
a[666] = 396799323;
a[667] = 657976136;
a[668] = 864535682;
a[669] = 433009242;
a[670] = 860830935;
a[671] = 33107339;
a[672] = 517661450;
a[673] = 467651311;
a[674] = 812398757;
a[675] = 202133852;
a[676] = 431839017;
a[677] = 709549400;
a[678] = 99643620;
a[679] = 773282878;
a[680] = 290471030;
a[681] = 61134552;
a[682] = 129206504;
a[683] = 929147251;
a[684] = 837008968;
a[685] = 422332597;
a[686] = 353775281;
a[687] = 469563025;
a[688] = 62265336;
a[689] = 835064501;
a[690] = 851685235;
a[691] = 21197005;
a[692] = 264793769;
a[693] = 326416680;
a[694] = 118842991;
a[695] = 84257200;
a[696] = 763248924;
a[697] = 687559609;
a[698] = 150907932;
a[699] = 401832452;
a[700] = 242726978;
a[701] = 766752066;
a[702] = 959173604;
a[703] = 390269102;
a[704] = 992293822;
a[705] = 744816299;
a[706] = 476631694;
a[707] = 177284763;
a[708] = 702429415;
a[709] = 374065901;
a[710] = 169855231;
a[711] = 629007616;
a[712] = 719169602;
a[713] = 564737074;
a[714] = 475119050;
a[715] = 714502830;
a[716] = 40993711;
a[717] = 820235888;
```

```
a[718] = 749063595;
a[719] = 239329111;
a[720] = 612759169;
a[721] = 18591377;
a[722] = 419142436;
a[723] = 442202439;
a[724] = 941600951;
a[725] = 158013406;
a[726] = 637073231;
a[727] = 471564060;
a[728] = 447222237;
a[729] = 701248503;
a[730] = 599797734;
a[731] = 577221870;
a[732] = 69656699;
a[733] = 51052704;
a[734] = 6544303;
a[735] = 10958310;
a[736] = 554955500;
a[737] = 943192237;
a[738] = 192526269;
a[739] = 897983911;
a[740] = 961628039;
a[741] = 240232720;
a[742] = 627280533;
a[743] = 710239542;
a[744] = 70255649;
a[745] = 261743865;
a[746] = 228474833;
a[747] = 776408079;
a[748] = 304180483;
a[749] = 63607040;
a[750] = 953297493;
a[751] = 758058902;
a[752] = 395529997;
a[753] = 156010331;
a[754] = 825833840;
a[755] = 539880795;
a[756] = 234683685;
a[757] = 52626619;
a[758] = 751843490;
a[759] = 116909119;
a[760] = 62806842;
a[761] = 574857555;
a[762] = 353417551;
a[763] = 40061330;
a[764] = 822203768;
a[765] = 681051568;
a[766] = 490913702;
a[767] = 9322961;
a[768] = 766631257;
a[769] = 124794668;
a[770] = 37844313;
a[771] = 163524507;
a[772] = 729108319;
a[773] = 490867505;
a[774] = 47035168;
a[775] = 682765157;
a[776] = 53842115;
a[777] = 817965276;
a[778] = 757179922;
a[779] = 339238384;
a[780] = 909741023;
a[781] = 150530547;
a[782] = 158444563;
a[783] = 140949492;
a[784] = 993302799;
a[785] = 551621442;
a[786] = 137578883;
a[787] = 475122706;
a[788] = 443869843;
a[789] = 605400098;
a[790] = 689361523;
a[791] = 769596520;
```

```
a[792] = 801661499;
a[793] = 474900284;
a[794] = 586624857;
a[795] = 349960501;
a[796] = 134084537;
a[797] = 650564083;
a[798] = 877097974;
a[799] = 379857427;
a[800] = 887890124;
a[801] = 159436401;
a[802] = 133274277;
a[803] = 986182139;
a[804] = 729720334;
a[805] = 568925901;
a[806] = 459461496;
a[807] = 499309445;
a[808] = 493171177;
a[809] = 460958750;
a[810] = 380694152;
a[811] = 168836226;
a[812] = 840160881;
a[813] = 141116880;
a[814] = 225064950;
a[815] = 109618190;
a[816] = 842341383;
a[817] = 85305729;
a[818] = 759273275;
a[819] = 97369807;
a[820] = 669317759;
a[821] = 766247510;
a[822] = 829017039;
a[823] = 550323884;
a[824] = 261274540;
a[825] = 918239352;
a[826] = 29606025;
a[827] = 870793828;
a[828] = 293683814;
a[829] = 378510746;
a[830] = 367270918;
a[831] = 481292028;
a[832] = 813097823;
a[833] = 798448487;
a[834] = 230791733;
a[835] = 899305835;
a[836] = 504040630;
a[837] = 162510533;
a[838] = 479367951;
a[839] = 275282274;
a[840] = 806951470;
a[841] = 462774647;
a[842] = 56473153;
a[843] = 184659008;
a[844] = 905122161;
a[845] = 664034750;
a[846] = 109726629;
a[847] = 59372704;
a[848] = 325795100;
a[849] = 486860143;
a[850] = 843736533;
a[851] = 924723613;
a[852] = 880348000;
a[853] = 801252478;
a[854] = 616515290;
a[855] = 776142608;
a[856] = 284803450;
a[857] = 583439582;
a[858] = 274826676;
a[859] = 6018349;
a[860] = 377403437;
a[861] = 244041569;
a[862] = 527081707;
a[863] = 544763288;
a[864] = 708818585;
a[865] = 354033051;
```

```
a[866] = 904309832;
a[867] = 589922898;
a[868] = 673933870;
a[869] = 682858433;
a[870] = 945260111;
a[871] = 899893421;
a[872] = 515264973;
a[873] = 911685911;
a[874] = 9527148;
a[875] = 239480646;
a[876] = 524126897;
a[877] = 48259065;
a[878] = 578214879;
a[879] = 118677219;
a[880] = 786127243;
a[881] = 869205770;
a[882] = 923276513;
a[883] = 937928886;
a[884] = 802186160;
a[885] = 12198440;
a[886] = 638784295;
a[887] = 34200904;
a[888] = 758925811;
a[889] = 185027790;
a[890] = 80918046;
a[891] = 120604699;
a[892] = 610456697;
a[893] = 573601211;
a[894] = 208296321;
a[895] = 49743354;
a[896] = 653691911;
a[897] = 490750754;
a[898] = 674335312;
a[899] = 887877110;
a[900] = 875880304;
a[901] = 308360096;
a[902] = 414636410;
a[903] = 886100267;
a[904] = 8525751;
a[905] = 636257427;
a[906] = 558338775;
a[907] = 500159951;
a[908] = 696213291;
a[909] = 97268896;
a[910] = 364983542;
a[911] = 937928436;
a[912] = 641582714;
a[913] = 586211304;
a[914] = 345265657;
a[915] = 994704486;
a[916] = 443549763;
a[917] = 207259440;
a[918] = 302122082;
a[919] = 166055224;
a[920] = 623250998;
a[921] = 239642551;
a[922] = 476337075;
a[923] = 283167364;
a[924] = 211328914;
a[925] = 68064804;
a[926] = 950202136;
a[927] = 187552679;
a[928] = 18938709;
a[929] = 646784245;
a[930] = 598764068;
a[931] = 538505481;
a[932] = 610424991;
a[933] = 864445053;
a[934] = 390248689;
a[935] = 278395191;
a[936] = 686098470;
a[937] = 935957187;
a[938] = 868529577;
a[939] = 329970687;
```

```
a[940] = 804930040;
a[941] = 84992079;
a[942] = 474569269;
a[943] = 810762228;
a[944] = 573258936;
a[945] = 756464212;
a[946] = 155080225;
a[947] = 286966169;
a[948] = 283614605;
a[949] = 19283401;
a[950] = 24257676;
a[951] = 871831819;
a[952] = 612689791;
a[953] = 846988741;
a[954] = 617120754;
a[955] = 971716517;
a[956] = 979541482;
a[957] = 297910784;
a[958] = 991087897;
a[959] = 783825907;
a[960] = 214821357;
a[961] = 689498189;
a[962] = 405026419;
a[963] = 946731704;
a[964] = 609346370;
a[965] = 707669156;
a[966] = 457703127;
a[967] = 957341187;
a[968] = 980735523;
a[969] = 649367684;
a[970] = 791011898;
a[971] = 82098966;
a[972] = 234729712;
a[973] = 105002711;
a[974] = 130614285;
a[975] = 291032164;
a[976] = 193188049;
a[977] = 363211260;
a[978] = 58108651;
a[979] = 100756444;
a[980] = 954947696;
a[981] = 346032213;
a[982] = 863300806;
a[983] = 36876722;
a[984] = 622610957;
a[985] = 289232396;
a[986] = 667938985;
a[987] = 734886266;
a[988] = 395881057;
a[989] = 417188702;
a[990] = 183092975;
a[991] = 887586469;
a[992] = 83334648;
a[993] = 797819763;
a[994] = 100176902;
a[995] = 781587414;
a[996] = 841864935;
a[997] = 371674670;
a[998] = 18247584;

cin >> n;
n++;
k = n / mod;
if (n >= mod * mod)
    cout << k + 1 << '␣';
else
    cout << k << '␣';

m = poww(f, k) % mod;
n %= mod;
int i = n / 1000000;
m = m * a[i] % mod;
for (i = i * 1000000 + 1; i <= n; i++)
    m = 1ll * m * i % mod;
```

```
    cout << m << '\n';

//     cin >> n;
//     cout << "a[0] = 1;\n";
//     for (int i = 2; i <= 1e9; i++) {
//         m = (1ll * m * i) % mod;
//         if (!(i % 1000000))
//             cout << "a[" << i / 1000000 << "] = " << m << ";\n";
//     }
//     cout << "0 " << m << '\n';

    return 0;
}
```

Task E ()

```
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;

int cnt[300100], doo[1200100], k = 500;
vector <vector <pair <pair <int , int>, pair <int , int> > > > q(1010);
vector <long long> ans(300100);

void build (int v, int tl, int tr) {
    if (tl == tr || !doo[v]) {
        doo[v] = 0;
        return;
    }

    doo[v] = 0;
    int tm = (tl + tr) / 2;
    build (2 * v + 1, tl, tm);
    build (2 * v + 2, tm + 1, tr);
}

void add (int v, int k, int tl, int tr) {
    if (tl > k || tr < k)
        return;
    if (tl == k && tr == k) {
//        cout << "add:  " << tl << '\n';
        doo[v]++;
        return;
    }

    int tm = (tl + tr) / 2;
    add (2 * v + 1, k, tl, tm);
    add (2 * v + 2, k, tm + 1, tr);
    doo[v] = doo[2 * v + 1] + doo[2 * v + 2];
}

void eras (int v, int k, int tl, int tr) {
    if (tl > k || tr < k)
        return;
    if (tl == k && tr == k) {
//        cout << "erase:  " << tl << '\n';
        doo[v]--;
        return;
    }

    int tm = (tl + tr) / 2;
    eras (2 * v + 1, k, tl, tm);
    eras (2 * v + 2, k, tm + 1, tr);
    doo[v] = doo[2 * v + 1] + doo[2 * v + 2];
}

int countt (int v, int l, int r, int tl, int tr) {
    if (tl > r || tr < l)
        return 0;
    if (tl >= l && tr <= r)
        return doo[v];

    int tm = (tl + tr) / 2;
    return countt (2 * v + 1, l, r, tl, tm) + countt (2 * v + 2, l, r, tm + 1, tr);
}

int main()
{
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);

    int n, m, qq, t[300100];

    cin >> n;
    for (int i = 0; i < n; ++i)
        cin >> t[i];
```

```
    cin >> qq;
    for (int i = 0; i < qq; ++i) {
        int a, b, d;
        cin >> a >> b >> d;
        b--;
        q[b / k].push_back({{a, b}, {d, i}});
    }
    build (0, 0, 300010);

    for (int f = 0; f <= qq / k; ++f) {
        sort(q[f].begin(), q[f].end());
        if (!q[f].size())
            continue;
        int l = q[f][0].first.first, r = q[f][0].first.second, d = q[f][0].second.first, num = q[f
            ][0].second.second;
        for (int i = l; i <= r; ++i) {
            add(0, t[i], 0, 300010);
            cnt[t[i]]++;
        }

        vector<pair<int, pair<int, int>>> gr;
        int i = 1;
        for (i = 1; i * i <= d; ++i)
            gr.push_back({(d + i - 1) / i, {i, i}});
        for (; i <= d;) {
            int g = (d + i - 1) / i;
            if ((d + i) / (i + 1) != g) {
                gr.push_back({g, {i, i}});
                ++i;
                continue;
            }
            int l = i + 1, r = d + 1;
            while (l < r) {
                int m = (l + r) / 2;
                if ((d + m - 1) / m != g)
                    r = m;
                else
                    l = m + 1;
            }
            gr.push_back({(d + i - 1) / i, {i, l - 1}});
            i = l;
        }
        gr[gr.size() - 1].second.second = 300010;

        for (int i = 0; i < gr.size(); ++i) {
            if (gr[i].second.first == gr[i].second.second) {
                ans[num] += 1ll * gr[i].first * cnt[gr[i].second.first];
            }
            else if (gr[i].second.first + 1 == gr[i].second.second) {
                ans[num] += 1ll * gr[i].first * (cnt[gr[i].second.first] + cnt[gr[i].second.first
                    + 1]);
            }
            else if (gr[i].second.first <= 300010) {
                int h = countt(0, gr[i].second.first, min(300010, gr[i].second.second), 0, 300010)
                    ;
                ans[num] += 1ll * gr[i].first * h;
            }
        }
//        cout << "\n\n";

        for (int j = 1; j < q[f].size(); ++j) {
            int d = q[f][j].second.first, num = q[f][j].second.second;
            while (l < q[f][j].first.first) {
                eras (0, t[l], 0, 300010);
                cnt[t[l]]--;
                l++;
            }
            while (r > q[f][j].first.second) {
                eras (0, t[r], 0, 300010);
                cnt[t[r]]--;
                r--;
            }
            while (r < q[f][j].first.second) {
                r++;
```

```
                add (0, t[r], 0, 300010);
                cnt[t[r]]++;
            }
//            cout << l << ' ' << r << "    " << d << ' ' << num << '\n';

            vector <pair <int, pair <int, int> > > gr;
            int i;
            for (i = 1; i * i <= d; ++i)
                gr.push_back({(d + i - 1) / i, {i, i}});
            for (; i <= d;) {
                int g = (d + i - 1) / i;
                if ((d + i) / (i + 1) != g) {
                    gr.push_back({g, {i, i}});
                    ++i;
                    continue;
                }
                int l = i + 1, r = d + 1;
                while (l < r) {
                    int m = (l + r) / 2;
                    if ((d + m - 1) / m != g)
                        r = m;
                    else
                        l = m + 1;
                }
                gr.push_back({(d + i - 1) / i, {i, l - 1}});
                i = l;
            }
            gr[gr.size() - 1].second.second = 300010;
            for (int i = 0; i < gr.size(); ++i) {
                if (gr[i].second.first == gr[i].second.second) {
                    ans[num] += 1ll * gr[i].first * cnt[gr[i].second.first];
                }
                else if (gr[i].second.first + 1 == gr[i].second.second) {
                    ans[num] += 1ll * gr[i].first * (cnt[gr[i].second.first] + cnt[gr[i].second.
                        first + 1]);
                }
                else if (gr[i].second.first <= 300010) {
                    int h = countt(0, gr[i].second.first, min(300010, gr[i].second.second), 0,
                        300010);
                    ans[num] += 1ll * gr[i].first * h;
                }
            }
        }

        for (int i = 0; i <= 300010; ++i)
            cnt[i] = 0;
        build (0, 0, 300010);
    }

    for (int i = 0; i < qq; ++i)
        cout << ans[i] << '␣';
    cout << '\n';

    return 0;
}
```

Task F ()