Secţiunea 11-12

PROBLEMA 1 **100 puncte**

Un număr X se numeşte oltenesc dacă nu conţine nici o putere de 2 de cel puţin 2 cifre ca subsecvenţă în scrierea sa zecimală. Nea Mărin are un număr N format din cel mult 100 de cifre şi se întreabă câte numere naturale cel mult egale cu N sunt olteneşti. Deoarece răspunsul poate fi destul de mare, se cere doar restul împărțirii sale la $10^9 + 7$.

Cerinţă
Se dau T întrebări, fiecare constând dintr-un singur număr N. Pentru fiecare întrebare să se calculeze câte numere $0 \le X \le N$ sunt olteneşti, modulo $10^9 + 7$.

Date de intrare
Pe prima linie a fişierului ***oltenesc.in*** se află numărul T. Urmează T linii, fiecare conţinând câte un număr N format din cel mult 100 de cifre zecimale, reprezentând o întrebare.

Date de ieșire
Fișierul de ieșire ***oltenesc.out*** va conţine T linii, constând în răspunsurile la cele T întrebări din fișierul de intrare.
Restricţii şi precizări

- $1 \le T \le 10$;
- $1 \le N < 10^{100}$;
- Pentru 10% din teste, $N < 10^6$;
- Pentru 50% din teste, $N < 10^{18}$.
- Prin subsecvenţă a unui număr zecimal înţelegem numărul obţinut prin păstrarea unei secvenţe continue formate din cifrele numărului iniţial. De exemplu, 234, 12 şi 12345 sunt subsecvenţe ale numărului 12345, dar 135 şi 432 nu sunt.

Exemplu

oltenesc.in	oltenesc.out	Explicaţie
4 33 999 232323 992391662939123897	32 938 194003 515744048	Singurele numere care nu sunt olteneşti cuprinse între 0 şi 33 sunt 16 şi 32. Singurele numere care nu sunt olteneşti cuprinse între 0 şi 999 sunt 128, 256, 512 şi cele de formele 16?, 32?, 64?, ?16, ?32, ?64 (observaţi cum numărul 164 se regăseşte de 2 ori în această listă).

Timp maxim de execuţie: 2 secunde/test
Memorie totală disponibilă: 512 MB din care 512 MB pentru stivă
Dimensiunea maximă a sursei: 10 Kb

Secţiunea 11-12

PROBLEMA 2 CARACATIȚĂ 100 puncte

Observăm că șirurile a, abb, cdd, opp, xzzyyy, qppaaammmm au ceva în comun. Mai exact, primul caracter se repetă o dată, al doilea de 2 ori, al treilea de 3 ori, etc. Vom numi aceste șiruri vrednice. Se dau N șiruri vrednice de caractere.

Cerință

Se dau M query-uri. Fiecare query este un șir de caractere. Vrem să aflam de câte ori fiecare query se "potrivește" în cele N șiruri date. Considerăm că un query q se potrivește într-un sir s dacă:

- q este subsir al lui s (abc este subșir al lui axbxcx)
- daca asupra lui q și s se aplică o operație care restrânge toate caracterele adiacente egale într-unul singur, q este subsecvență a lui s (abc este subsecvență a lui xabcx) (q se potrivește în s de câte ori q restrâns apare ca subsecvența în s restrâns)

De exemplu, abc se potrivește în abbccc, xaabbbccc o dată, în abbcccaaaabbbbbccccc de 2 ori, și în abbdddcccc, axxbbbccc, abb niciodată.

Date de intrare

Pe prima linie a fișierului ***caracatita.in*** se află numărul N. Urmează N linii care conțin fiecare câte un șir vrednic de caractere,conținând doar caracterele a-z .
Pe următoarea linie se afla numărul M. Urmează M linii care conțin fiecare câte un șir de caractere conținând doar caracterele a-z.

Date de ieșire

Pe linia i din fișierul de ieșire ***caracatita.out*** se va afla numărul de potriviri al query-ului i.

Restricții și precizări

- 1 <= N <= 500
- 1 <= Lungimea unui sir vrednic <= 5050
- 1 <= M <= 100 000
- 2 <= Lungimea unui şir query <= 1000
- Pentru 20 % din teste, N <= 50
- Pentru alte 20 % din teste, M <= 1000

Exemple

caracatita.in	caracatita.out
1 abbcccdddd 4 abc abcc abccc abcccc	1 1 1 0

caracatita.in	caracatita.out
2 axxyyyzzzzppppp xyy 5 axp axz xxyyyzzz xyy z	0 0 1 2 1

Timp maxim de execuţie: 0.5 secunde/test
Memorie totală disponibilă: 300 MB, din care 300 MB pentru stivă
Dimensiunea maximă a sursei: 10 Kb

PROBLEMA 3 DOUBLEDROP 100 puncte

3XIMO este un DJ faimos. El vrea să pună o secvență de melodii 2 câte 2 (double drop) la o anumită petrecere.
O n-melodie este o secvență de n vibrații(V) și n pauze (P) (n >= 0).
O n-melodie k-rupe dacă numărul de prefixe care se termină în V, cu diferență dintre V și P strict pozitivă, este egală cu k.
De exemplu: (ultima literă dintr-un prefix care contribuie la k-rupere este îngroșată):

- **VVV**P**VV**PPPP este o 5-melodie care 5-rupe
- **V**P**V**PPPVV este o 4-melodie care 2-rupe
- PPVV este o 2-melodie care 0-rupe

3XIMO își cunoaște foarte bine publicul și știe la fiecare moment i următoarele informații:
N[i] M[i] - publicul dorește ca 3XIMO fie să facă double drop (ceea ce înseamnă că alege 2 melodii și le pune într-o ordine) cu 2 melodii într-o ordine: prima o j-melodie care min(j, M[i])-rupe și a doua o (N[i]-j)-melodie care min((N[i]-j), M[i])-rupe. (0 <= j <= N[i], M[i] <= N[i], 1 <= i <= LEN)
Deci la momentul i, 3XIMO poate alege oricare 2 melodii care satisfac constrâgerile.
O secvență de double drop-uri se numește "blanaos" dacă respectă toate informațiile la fiecare moment. 3XIMO vă roagă să îi spuneți câte secvențe "blanaos" sunt modulo $10^9 + 7$.

Date de intrare
Fișierul ***doubledrop.in*** conține pe prima linie un număr LEN, egal cu lungimea secvenței de double dropuri. Pe a (i+1)-a linie vor apărea N[i] <= 1e5 și M[i] <= 1e5 cu semnificația din problema.

Date de ieșire
Fișierul ***doubledrop.out*** conține pe prima linie, numărul de secvențe "blanaos" modulo $10^9 + 7$.

Restricții și precizări
- LEN <= 10, 0 <= M[i] <= N[i] <= 10 pentru 10 puncte;
- LEN <= 4000, 0 <= M[i] <= N[i] <= 4000 pentru alte 30 de puncte;

- LEN <= 100.000, 0 <= M[i] <= N[i] <= 100.000 pentru 100 de puncte;
- Prin convenție, exista o singura 0-melodie care 0-rupe;

Exemple

doubledrop.in	doubledrop.out	Explicație
2 1 1 0 0	2	3XIMO la timp 1 poate să facă double drop în următoarele moduri: Prima o 1-melodie care 1-rupe (există doar o astfel de melodie) cu a doua o 0-melodie care 0-rupe (există doar o astfel de melodie) Prima o 0-melodie care 0-rupe (există doar o astfel de melodie) cu a doua o 1-melodie care 1-rupe (există doar o astfel de melodie) 3XIMO la timp 2 poate să facă double drop în următorul mod: Prima o 0 melodie care 0-rupe (există doar o astfel de melodie) și a doua același fel de melodie Deci 3XIMO în total poate alege o subsecvență de double drop-uri "blanaos" în (1+1)*1 = 2 moduri
4 0 0 0 0 0 0 0 0	1	Există o posibilitate pentru fiecare drop deci există o singură secvență de double drop-uri "blanaos".
3 2 1 0 0 3 2	70	

Timp maxim de execuţie: 0.5 secunde/test
Memorie totală disponibilă: 16 MB din care 16 MB pentru stivă
Dimensiunea maximă a sursei: 5 Kb

PROBLEMA 4 LANŢURI 100 puncte

Se dă un arbore oarecare (graf neorientat conex şi fără cicluri) cu n noduri. Fiecare nod are asociată o cheie (număr natural). Determinați numărul de lanțuri elementare care au suma cheilor asociate nodurilor componente un număr divizibil cu 3. Un lanț se cheamă elementar dacă nodurile sale sunt distincte. Două lanțuri se consideră distincte dacă diferă prin cel puțin un nod sau prin ordinea de așezare a nodurilor. Un lanț se consideră identic cu cel obținut citind nodurile invers.

Date de intrare
Fişierul ***lanturi.in*** conține pe prima linie un număr natural n – numărul de noduri ale arborelui. Pe linia a doua sunt valorile cheilor nodurilor, separate prin câte un spațiu. Pe fiecare din următoarele

n-1 linii se află câte două numere naturale separate prin spațiu, reprezentând capetele câte unei muchii.

Date de ieșire
Fișierul ***lanturi.out*** conține restul împărțirii valorii determinate la 10007 (zece mii șapte).

Restricții și precizări:
- 2 <= n <= 100000;
- Cheile sunt numere naturale de maxim 3 cifre;
- Nodurile sunt numerotate de la 1 la n.
- Considerăm inclusiv lanțurile de lungime 0 (cele formate dintr-un singur nod).

Exemplu

lanturi.in	lanturi.out	Explicație
4 1 2 3 4 1 2 1 3 4 1	3	Cele 3 lanţuri sunt: [3], [1,2], [3,1,2]

Timp maxim de execuţie: 1 secundă/test.
Memorie totală disponibilă: 10 MB, din care 10 MB pentru stivă
Dimensiunea maximă a sursei: 10 Kb