IIOT 2022-23 Runda 4 - Solutii

Comisia IIOT

Februarie 2023

1 Problem Books

Autor: Filippo Casarin

Mai intai vom misca cate o carte de $ceil(n/2)$ ori de la teancul 0 la teancul 1 cu costul 0, acum cartile din teancul 1 fiind ordonate corect, deci trebuie acum inversate cartile din cea de-a doua jumatate.

Pentru a face asta vom muta mai intai cele $floor(n/2)$ carti ramase din teancul 0 in teancul 1 cu costul 1, apoi mutandu-le una cate una inapoi pe teancul 0.

In cele din urma, vom muta toate cartile din teancul 0 in teancul 1 cu costul 1 pentru a uni cele doua jumatati.

Costul final va fi 2, ceea ce aduce punctajul maxim conform formulei de punctare.

2 Problem GCD

Autor: László Nikházy

Raspunsul nostru va fi mereu de forma $k * D$, $(k + 1) * D$, unde k este un numar intreg pozitiv. Deoarece k si $k + 1$ sunt mereu prime intre ele, cmmdc-ul celor doua valori va fi mereu D. Nu avem solutie daca $2 * D$ are mai mult de N cifre, deoarece D si $2 * D$ sunt cele mai mici numere pozitive care sunt multiplii ai lui D. Altfel, vom putea adauga zerouri la D pana cand are N cifre, gasind astfel un multiplu al lui D de N cifre, numar pe care il vom nota A. Se poate demonstra cu usurinta ca fie $A - D$, fie $A + D$ for avea N cifre, si vom denumi acest numar ca fiind B.

3 Problem Spring Carnival

Autor: Stefan Dascalescu

Putem pastra un vector de frecventa pentru fiecare frecventa pe care o avem in vector, pe baza valorilor pe care le obtinem de la intrare. La fiecare pas, daca avem x valori cu frecventa y si citim un $y + 1$ de la intrare, vom inmulti raspunsul cu x deoarece avem x moduri de a colora acele palarii cu culorile respective. De asemenea, va trebui sa avem grija la actualizarea frecventelor, precum si la calcularea produsului modulo $10^9 + 7$, pentru a evita eventualele overflow-uri.

4 Problem Lottery

Autor: Luca Chiodini

Pentru primul subtask, putem continua să creștem S cu 1 până când găsim o soluție sau atingem un prag și afisam -1.

Pentru celelalte subsarcini, avem nevoie de o metoda rapidă de a verifica dacă putem obține un număr valid schimbând doar ultimele K cifre ale lui S.

Putem face câteva observații care ne permit să putem verifica acest lucru in $O(1)$:

- suma minimă posibilă de cifre este realizată folosind modelul 00100100100100100

- suma maximă posibilă de cifre este realizată folosind modelul 99899899899899899

- toate sumele dintre minim și maxim sunt posibile (singura excepție este atunci când avem doar o cifră, dar suma necesară este egală cu cele 2 cifre anterioare)

Pentru a calcula sumele minime și maxime, trebuie să acordăm atenție celor 2 cifre anterioare pentru a evita 3 cifre egale consecutive.

Acum putem rezolva problema inițială după cum urmează:

Pentru fiecare K, creștem K-a cifră cu 1 și verificăm dacă putem obține un răspuns valid schimbând doar ultimele K cifre. Găsim cel mai mic K valid și reconstruim răspunsul.

Câteva observatii: - numărul S din intrare poate conține 3 cifre egale consecutive, deci K trebuie să fie și suficient de mare pentru a include toate acele triple. - dacă N este mare, poate fi necesar să adăugați cifre in plus la numărul S pentru a-l face suficient de lung.

5 Problem Ancient Text

Autor: László Nikházy

Algoritmul brute force are complexitatea $O(K * N^2)$, deoarece calculul distantei a doua stringuri este $O(K)$, iar pentru fiecare string ar trebui sa calculam distanta de la el la toate celelalte, ceea ce este indeajuns pentru 36 de puncte.

O prima observatie pentru a imbunatati complexitatea ar fi ca in loc sa calculam media distantelor, sa calculam suma tuturor distantelor la toate celelalte stringuri. Pentru a putea face asta, vom putea folosi un algoritm ce ruleaza in $O(K)$.

Vom parcurge fiecare pozitie de la stanga la dreapta. Sa presupunem ca $S[i][0] = a$. La pozitia 0, fiecare alt string $S[j]$ va adauga 1 la distanta daca litera de pe acea pozitie nu este 'a', acest numar de stringuri se poate calcula folosind un vector de frecventa pentru fiecare litera si pozitie in parte.

Sa recapitulam solutia completa. Mai intai, calculam pentru fiecare litera de la 'a' la 'z' de cate ori apare la fiecare pozitie, numar pe care il vom nota cu $cnt[i][c]$. Apoi, pentru fiecare string S vom calcula distanta sa totala fata de toate celelalte stringuri prin insumarea valorilor N - $cnt[k][S[k]]$ pentru fiecare k de la 0 la $K-1$.

6 Problem LRTree

Autor: Mihnea-Vicentiu Buca

Toate afirmatiile de mai jos sunt adevarate pentru n par.

Pentru $n \leq 20$, generam toti arborii binari cu proprietatea ceruta.

Pentru $n \leq 1000$, putem folosi urmatoarea abordare:

Observatie 1: Rezultatul este dat de relatia $dp[n-1] * dp[0] + dp[n-2] * dp[1] + ... + dp[n/2] * dp[n/2-1]$, unde $dp[i]$ = numarul arborilor binari cu n noduri

Observatie 2: Numarul arborilor binari cu n noduri este dat de al n lea numar catalan

Demonstratie: $dp[i] = dp[0] * dp[i-1] + dp[1] * dp[i-2] + ... + dp[i-1] * dp[0]$, aceasta recurenta este identica cu cea a lui $catalan(i)$ Corolar: Rezultat = $catalan(n-1) * catalan(0) + catalan(n-2) * catalan(1) + ... + catalan(n/2) * catalan(n/2-1)$

$catalan(i) = (2n!)/((n+1)!n!)$, sau construim tabloul unidimensional dp[i] folosind formula de recurenta din demonstratie

Rezultatul trebuie afisat modulo 1000000007

Pentru a obtine punctajul maxim, vom calcula $catalan(i)$ folosind precalcularea factorialelor si a inverselor modulare, folosind formula de mai sus.

Pentru a ajunge la raspunsul final, vom imparti raspunsul obtinut la 2.

7 Problem Tree Median

Autor: Bence Deak

Putem reformula problema in urmatorul fel: care este cel mai mic numar M astfel incat sunt cel putin K drumuri care au o mediana cel mult egala cu M. Putem rezolva aceasta problema cu o cautare binara pe raspuns.

Pentru un M fixat, putem face urmatorul lucru: Daca $w[i] \leq M$, vom nota a i-a muchie cu 0, altfel cu 1. Putem observa faptul ca mediana unui drum e cel mult M daca si numai daca nu avem mai multi de 1 pe drum decat avem 0. Vom nota aceste drumuri ca fiind bune.

In cele ce urmeaza, putem folosi centroid decomposition. Sa consideram un arbore cu L noduri si centroidul C. Vrem sa gasim numarul de drumuri bune cu lungime nenula care trec prin C. Sa notam C[u, 0] si C[u, 1] ca fiind numarul de 0 si 1 pe drumul de la C la u.

Sa consideram un drum oarecare de la u la v care trece prin C. Acest drum este bun daca si numai daca C[u, 0] + C[v, 0] ¿= C[u, 1] + C[v, 1] ¡=¿ C[u, 0] - C[u, 1] ¿= C[v, 1] - C[v, 0].

Putem calcula cu usurinta numarul de perechi (u, v) != (c, c) care satisfac inegalitatea de mai sus folosind sume partiale. Vom nota raspunsul final cu D.

Acum trebuie doar sa excludem drumurile care nu trec prin C. Toate aceste perechi sunt in totalitate continute de un subarbore corespunzator unui copil de-al lui C. Astfel, va trebui sa rulam calculul urmator pentru fiecare subarbore, astfel scazand raspunsul din D. Numarul rezultat este exact numarul de drumuri bune cu lungime pozitiva care trec prin centroid.

Complexitatea finala este $O(N * log^2(N))$, iar memoria necesara este $O(N)$