

# IOT 2021-22 Runda 2 - Soluții

Comisia IOT

Decembrie 2021

## 1 Problema Christmas Balls

AUTOR: EDOARDO MORASSUTTO

### 1.1 Descrierea soluției

Dificultate: Grea

Pentru a rezolva problema sunt necesare cunoașterea tehnicii small-to-large, despre care se poate citi aici: Small-to-large.

Pentru rezolvarea diferitelor subtask-uri ce le are această problemă se pot folosi soluții ce fie calculează soluția în  $O(n^2)$ , fie în  $O(n)$  pentru subtask-urile 3 și 5.

Pentru rezolvarea problemei cu punctaj maxim, vom rula un DFS pentru a afla numărul de apariții al fiecărui număr. Pentru a putea optimiza complexitatea actualizărilor, vom uni întotdeauna nodurile care au dimensiunea subarborelui lor mai mică la nodurile care au dimensiunea subarborelui mai mare, astfel evitând prea multe update-uri inutile. Pentru optimizarea acestui proces, se pot folosi unordered map.

## 2 Problema Goal Statistics

AUTOR: STUD. ȘTEFAN-COSMIN DĂSCĂLESCU, UNIVERSITATEA DIN BUCUREȘTI

### 2.1 Descrierea soluției

Dificultate: Medie

Pentru a rezolva primul subtask, se poate sorta vectorul înainte de fiecare query, mai apoi aflându-se suma celor mai mici  $p$  numere, complexitatea fiind  $O(n^2 \log n)$

Pentru a rezolva cel de-al doilea subtask, ne vom folosi de faptul că valorile sunt mici, astfel încât vom memora valorile folosind un vector de frecvență, iar de aceea vom putea folosi Counting Sort, complexitatea algoritmului devenind astfel  $O(n * k)$

Pentru soluția completă, vom avea o abordare similară ca la subtaskul 2, doar că vom folosi fie arbori de intervale, fie arbori indexați binar pentru a memora frecvențele și pentru a procesa query-urile. Pentru a afla valorile ce vor fi păstrate, putem căuta binar cel mai mic index  $i$  astfel încât sunt cel puțin  $p$  valori între 1 și  $i$ , acest lucru putând fi făcut fie direct în arborele de intervale/arborele indexat binar, fie folosind o căutare binară și un query clasic în aceste structuri de date, ambele abordări obținând punctaj maxim.

În funcție de abordarea folosită, complexitatea soluției va fi  $O(n \log k)$  sau  $O(n \log^2 k)$

### 3 Problema Kalindrome

AUTOR: WILLIAM DE LUIGI

#### 3.1 Descrierea soluției

Dificultate: Ușoară

Pentru a rezolva această problemă, putem observa faptul că răspunsul poate fi doar unul dintre divizorii lungimii șirului. Pentru fiecare divizor al șirului, putem verifica proprietatea de kalindrome în  $O(n)$ , astfel complexitatea soluției fiind  $O(n * nrdivizori)$ , număr care este cel mult egal cu 240 pentru toate valorile de la 1 la  $10^6$ , Highly Composite Numbers - numere extrem compuse.

### 4 Problema Maximum Occurrences

AUTOR: STUD. VLAD ANDREI ANDRIEȘ, UNIVERSITATEA DIN BUCUREȘTI

#### 4.1 Descrierea soluției

Dificultate: Ușoară

Se poate observa că algoritmul descris în enunțul problemei este algoritmul descris în conjectura Collatz. De asemenea, se poate demonstra folosind calculatorul că pentru toate valorile posibile din input, vom ajunge la numărul 1. Să analizăm ce se întâmplă după ce ajungem la 1. Următoarele numere ar fi 4, 2, 1, 4, 2, 1, ... Astfel, pentru toate valorile cu excepția lui 1 și 2, valoarea maximă care apare cel mai des va fi 4. Pentru 1 și 2, vom afișa 1, respectiv 2. Complexitatea algoritmului descris este  $O(1)$ .

## 5 Problema Secret Santa

AUTOR: STUD. IOAN POPESCU, UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI

### 5.1 Descrierea soluției

#### 5.1.1 Solutie 60 de puncte

Dificultate: Ușoară

Pentru a obține acest punctaj parțial, putem folosi o abordare similară cu principiul includerii și excluderii. Astfel, vom scădea din numărul permutărilor de lungime  $n$  numărul permutărilor ce nu respectă condiția cerută de enunț. Pentru a face asta, vom fixa pe rând numărul punctelor fixe, ce poate fi între 1 și  $n$ .

Pentru o valoare  $k$ , numărul permutărilor ce au  $k$  puncte fixe este  $C(n, k) * (n - k)!$ , unde  $C(n, k)$  reprezintă combinări de  $n$  luate câte  $k$ . Așa cum se obișnuiește în cazul principiului includerii și excluderii, vom aduna la răspuns valorile corespunzătoare pozițiilor cu  $k$  impar și vom scădea valorile corespunzătoare pozițiilor cu  $k$  par. În cele din urmă, vom scădea din  $n!$  răspunsul aflat la acest pas. Complexitatea algoritmului este  $O(sum_n)$ , unde  $sum_n$  e suma valorilor lui  $n$  din input.

#### 5.1.2 Solutie 100 de puncte

Dificultate: Medie

Pentru a obține punctaj maxim se poate folosi programarea dinamică. Astfel, vom defini  $dp[i]$  ca fiind numărul de permutări care respectă condiția dată. Se poate observa că  $dp[i]$  e egal cu  $(i - 1) * (dp[i - 1] + dp[i - 2])$ , unde  $dp[0] = 1$  și  $dp[1] = 0$ . Pentru mai multe detalii, se poate consulta acest articol de pe wikipedia: Derangements

## 6 Problema Tiny Types

AUTOR: STUD. ȘTEFAN-COSMIN DĂSCĂLESCU, UNIVERSITATEA DIN BUCUREȘTI

### 6.1 Descrierea soluției

Dificultate: Medie

Pentru a rezolva problema vom folosi metoda programării dinamice. Astfel, vom defini  $dp[i][j]$  ca fiind costul minim de a memora primele  $i$  numere în ordine, știind că tipul de date al numărului de pe poziția  $i$  este  $j$ . Pentru a afla valorile corespunzătoare stărilor din dinamică, vom procesa toate tranzițiile de la un tip de date  $a$  la un tip de date  $b$ , pentru toate stările eligibile de tipul  $dp[i-1][a]$  și  $dp[i][b]$ .

Astfel, complexitatea algoritmului este  $O(n * k^2)$ .

## 7 Problema Cntnk

AUTORI: STUD. BOGDAN IOAN POPA, UNIVERSITATEA DIN BUCUREȘTI,  
PROF. DANIEL POPA, LICEUL TEORETIC AUREL VLAICU” ORĂȘTIE

### 7.1 Descrierea soluției

Dificultate: Grea

O primă soluție ar fi generarea folosind metoda Backtracking a tuturor soluțiilor posibile, abordare ce poate lua 57 de puncte.

Soluția constă în rezolvarea problemei folosind programare dinamică. Putem calcula  $dp[i][conf] =$  în câte moduri putem colora primele  $i$  linii folosind  $K$  culori, astfel încât a  $i$ -a linie are configurația  $conf$ .  $conf$  reprezintă un număr în baza  $K$  de  $N$  cifre, fiecare cifră de la 0 la  $K-1$  reprezintă o culoare, care nu are două cifre egale pe poziții consecutive.

Recurența este  $dp[i][conf] =$  suma din  $dp[i-1][confant]$  unde  $confant$  reprezintă o configurație de linie care se poate afla pe linia  $i-1$  considerând că pe linia  $i$  este configurația  $conf$  ( $confant$  și  $conf$  nu au cifre egale pe aceeași poziție).

Pentru obținerea punctajului maxim, se pot precalculea toate răspunsurile posibile folosind soluția anterioară (sunt doar 32 de perechi posibile de forma  $(N, K)$ ), lucru ce ar trebui să dureze cel mult câteva minute.