DP. Exponențiere rapidă. Ciclu hamiltonian

BUZATU GIULIAN

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/rucsac

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/rucsac

Hint: dp[i][j] = profitul maxim pe care îl putem obține dacă luăm o submulțime din primele i elemente, care au suma greutăților egală cu j.

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/rucsac

Soluție: Luăm pe rând obiecte în mulțimea noastră și actualizăm matricea dp. În momentul în care adăugăm un nou obiect, sunt două posibilități: obiectul nu îmbunătățește răspunsul și obiectul îmbunătățește răspunsul. Formal:
dp[i][j]=max{dp[i-1][j], dp[i-1][j-weight[i]]+profit[i]}

Complexitate timp și spațiu: $O(N*G)$

Implementare 1: https://infoarena.ro/job_detail/3192606?action=view-source
Implementare 2: https://infoarena.ro/job_detail/3192607?action=view-source

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/rucsac

Observație: Nu încape toată matricea de dp în memorie, deci luăm doar 50 de puncte.

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/rucsac

Observație: Nu încape toată matricea de dp în memorie, deci luăm doar 50 de puncte.

Cum rezolvăm problema?

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/rucsac

Soluție 1: Observăm că recurența noastră depinde doar de linia anterioară, deci putem păstra doar câte 2 linii ale matricei.

Implementare: https://infoarena.ro/job_detail/3192609?action=view-source

Soluție 2: Parcurgem invers în cel de-al doilea for și astfel putem reține în loc de o matrice cu două linii, un simplu vector.

Implementare: https://infoarena.ro/job_detail/3192612?action=view-source

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/jocul

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/jocul

Hint: Modelăm problema ca fiind problema rucsacului. Cât este capacitatea rucsacului?

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/jocul

Soluție: Putem modela problema ca fiind problema rucsacului, cu capacitatea G/2. De data aceasta vrem doar să știm dacă putem atinge o anumită lungime mai mică sau egală cu G/2 (nu avem un profit). Deci doar marcăm dacă se poate sau nu. Luăm suma primei submulțimi ca fiind cea mai mare valoare pe care o atingem după for-urile care calculează dinamica de rucsac, iar cealaltă va fi G-suma. Nu are sens să facem algoritmul de rucsac pentru o greutate mai mare de G/2, deoarece cealaltă submulțime ar avea greutatea mai mică de G/2, caz tratat deja, numai că sumele sunt inversate.

Implementare: https://infoarena.ro/job_detail/3192619?action=view-source

Exponențiere rapidă

Să rezolvăm problema: https://infoarena.ro/problema/lgput

Exponențiere rapidă

Să rezolvăm problema: https://infoarena.ro/problema/lgput

Hint: $a^{2k} = a^k * a^k$ și $a^{2k+1} = a^k * a^k * a$

Exponențiere rapidă

Să rezolvăm problema: https://infoarena.ro/problema/lgput

Soluție 1: Recursiv, calculăm a^k și îl folosim pentru a calcula a^{2k}.
Soluție 2: O observație importantă este că orice număr natural se scrie în mod unic ca sumă de puteri ale lui 2. Așadar, putem să ne folosim de descompunerea în baza 2 a puterii b pentru a calcula a^b .

Complexitate: $O(\log b)$
Implementare 1: https://infoarena.ro/job_detail/3192615?action=view-source
Implementare 2: https://infoarena.ro/job_detail/3192616?action=view-source

Exponențiere rapidă pe matrice

Să rezolvăm problema: https://infoarena.ro/problema/kfib

Exponențiere rapidă pe matrice

Să rezolvăm problema: https://infoarena.ro/problema/kfib

Hint: Putem transforma recurența liniară într-o înmulțire de matrice.

Exponențiere rapidă pe matrice

Să rezolvăm problema: https://infoarena.ro/problema/kfib

Soluție: $\begin{pmatrix} F_n & F_{n+1} \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} F_{n-1} & F_n \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$. Notăm $M_i = \begin{pmatrix} F_n & F_{n+1} \\ 0 & 0 \end{pmatrix}$ și $A = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$.
Avem $M_i = M_{i-1}A$ și $M_{i-1} = M_{i-2}A^2$. Din asociativitatea înmulțirii matricelor rezultă că $M_i = M_1 A^{i-1}$. Putem folosi algoritmul de exponențiere rapidă pentru a calcula A^{i-1} și să aflăm al n-lea termen Fibonacci în complexitate $O(\log n)$.

Implementare: https://infoarena.ro/job_detail/3192617?action=view-source

Exponențiere rapidă pe matrice

Să rezolvăm problema: https://infoarena.ro/problema/iepuri

Exponențiere rapidă pe matrice

Să rezolvăm problema: https://infoarena.ro/problema/iepuri

Hint: Putem transforma recurența liniară într-o înmulțire de matrice, aplicând aceiași pași ca la problema anterioară.

Exponențiere rapidă pe matrice

Să rezolvăm problema: https://infoarena.ro/problema/iepuri

Soluție: $\begin{pmatrix} a_{n-2} & a_{n-1} & a_n \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} a_{n-3} & a_{n-2} & a_{n-1} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & C \\ 1 & 0 & B \\ 0 & 1 & A \end{pmatrix}$. Acum aplicăm același raționament ca la problema anterioară, numai că ridicăm matricea $\begin{pmatrix} 0 & 0 & C \\ 1 & 0 & B \\ 0 & 1 & A \end{pmatrix}$ la puterea n-2 pentru a afla a_n.

Implementare: https://infoarena.ro/job_detail/3192618?action=view-source

Ciclu hamiltonian

Să rezolvăm problema: https://infoarena.ro/problema/hamilton

Ciclu hamiltonian

Să rezolvăm problema: https://infoarena.ro/problema/hamilton

Hint 1: Ne-ar ajuta să știm în ce noduri am ajuns până acum.

Ciclu hamiltonian

Să rezolvăm problema: https://infoarena.ro/problema/hamilton

Hint 1: Ne-ar ajuta să știm în ce noduri am ajuns până acum.
Hint 2: Ne-ar ajuta să știm care a fost ultimul nod.

Ciclu hamiltonian

Să rezolvăm problema: https://infoarena.ro/problema/hamilton

Hint 1: Ne-ar ajuta să știm în ce noduri am ajuns până acum.
Hint 2: Ne-ar ajuta să știm care a fost ultimul nod.
Hint 3: Pentru a ține minte în ce noduri am fost până acum folosim o mască de biți.

Ciclu hamiltonian

Să rezolvăm problema: https://infoarena.ro/problema/hamilton

Soluție: dp[mask][i] = costul minim prin care putem parcurge mulțimea de noduri reprezentată de biții de 1 din mask, cu ultimul nod fiind i.
Pentru o mască fixată, cu i și j fiind doi biți setați din ea. Avem recurența:
dp[mask][i] = min{dp[mask][i], dp[mask^(1<<i)][j]+graph[j][i]}.
Inițializăm matricea de dp cu INF, mai puțin dp[1][0], care e 0, pentru că pornim din nodul 0. La final, răspunsul va fi minimul din costul lanțului găsit + muchia de la capătul lanțului la nodul 0.
Complexitate: $O(N^2 2^N)$.
Implementare: https://infoarena.ro/job_detail/3189431?action=view-source

Temă

- https://infoarena.ro/problema/energii
- https://csacademy.com/contest/archive/task/partial_ladder_graph/statement
- https://infoarena.ro/problema/seg
- https://infoarena.ro/problema/ubuntzei

Probleme suplimentare

- https://infoarena.ro/problema/triunghi
- https://qoj.ac/contest/1223/problem/6410
- https://codeforces.com/problemset/problem/430/E

Lectură suplimentară

- https://usaco.guide/gold/knapsack?lang=cpp
- https://usaco.guide/gold/dp-bitmasks?lang=cpp