Autor: Velea Alexandru

Structura solutiei
1) Observatii generale
2) Solutie in O(N * X) - 15 puncte
3) Solutie restrictii de tipul 3 - 15 puncte
4) Solutie restrictii de tipul 4 - 20 puncte
5) Solutia oficiala - 100 puncte

1) Observatii generale
Pentru un drum care are lungime D si viteza V, timpul de deplasare este de D / V
Daca marim viteza drumului de la V la (V + 1), timpul de deplasare devine D / (V + 1)
Astfel, timpul salvat in urma maririi vitezei este de (D / V) - (D / (V + 1)) = D / (V * (V + 1))

Se observa ca prima imbunatatire salveaza D / (V * (V + 1)) secunde, a doua imbunatatire D / ((V + 1) * (V + 2)), etc
Astfel, fiecare imbunatatire salveaza din ce in ce mai putin timp.

2) Solutie in O(N * X)
Folosiind observatiile de la punctul 1, putem alege o strategie greedy de imbunatatire a drumurilor.
Daca am avea de facut o singura imbunatatire, am alege sa o facem pe drumulul unde aceasta ar salva cat mai mult timp.
Acesta fiind drumul cu D / (V * (V + 1)) maxim.

Astfel, putem demonstra ca tot timpul este bine sa alegem sa imbunatatim drumul care ne salveaza cat mai mult timp.
Daca am avea de facut doar o imbunatatire, e bine sa alegem acel drum. Dupa ce l-am ales, urmatoarea imbunatatire
va fii mai mica decat cea precedenta, iar acest lucru ne garanteaza faptul ca daca am putea sa facem acest lucru acum,
nu am avea nici un motiv sa facem imbunatatirea pe drumul respectiv mai tarziu.

Iteram de X ori peste toate drumurile, la fiecare pas alegem drumul care salveaza cel mai mult timp si il imbunatatim.

3) Solutia pentru tipul 3
Observam ca daca am vrea sa imbunatatim un drum care are distanta D, este cel mai bine sa imbunatatim drumul care are
viteza minima, pentru a maximiza timpul castigat.

Astfel, faptul ca distantele sunt mai mici decat 200, permite o optimizare a algoritmului de la pasul 2.
In loc sa iteram la fiecare pas prin toate drumurile si sa il alegem pe cel care salveaza cat mai mult timp, putem
itera prin distantele drumurilor, aflam care este viteza minima a unui drum cu distanta respectiva, iar mai apoi
imbunatatim unul dintre drumurile care aveau viteza minima pentru distanta fixata (este posibil sa fie mai multe astfel de drumuri)

Astfel, fiecare imbunatatire se poate realiza in 200 de iteratii, nu in N(50000).

Pentru a afla drumul de viteza minima pentru o distanta fixata, putem sa retinem drumurile intr-o matrice de frecvente
drumuri[d][v] reprezentand cate drumuri au distanta d si viteza v
mai tinem un sir viteza_minima[d] reprezentand cea mica viteza a unui drum cu distanta d
- echivalent cu prima valoare v pentru care drumuri[d][v] are o valoare nenula.

Daca la un pas am ales sa imbunatatim un drum care are distanta d si viteza v,
trebuie sa microram numarul de drumuri cu viteza v - drumuri[d][v] -= 1
marim numarul de drumuri cu viteza (v + 1) - drumuri[d][v + 1] += 1
* daca nu mai avem drumuri cu viteza v, marim viteza_minima[d] - if (drumuri[d][v] == 0) { viteza_minima[d] += 1; }

4) Solutia pentru tipul 4
Asemanator cu tipul 3, doar ca aici distantele nu sunt pana in 200, ci toate distantele sunt egale.
Astfel, ne intereseaza doar vitezele, distanta fiind necesara doar la calcularea rezultatului.

O solutie ar fii sa tinem un sir de frecvente pentru fiecare viteza, si sa aplicam urmatorul algritm

```
int i = 1;
while (x) {
        if (frecventa[i] > x) {
            frecventa[i + 1] += frecventa[i];
        x -= frecventa[i];
        frecventa[i] = 0;
    } else {
        frecventa[i] -= x;
        frecventa[i + 1] += x;
        x = 0;
```

```
    }

    i += 1;
}
```

O astfel de solutie se comporta foarte bine cand valoarea lui N este mare, dar daca N = 1, de exemplu, ar face X pasi.

Ca sa scapam de cazul acesta, am putea cauta binar cea mai mica viteza.
Astfel putem afla cate imbunatatori trebuie sa facem - suma_din(max(0, viteza_cautata_binar - viteza[i]))
Daca numarul de imbunatatiri depaseste X, reducem viteza, altfel o marim.
La final, imbunatatim toate drumurile <= viteza cautata pana la ea, si mai imbunatatim unele drumuri cu acea
viteza(nu le putem imbunatatii pe toate) daca mai avem imbunatatiri ramase.

5) Solutia oficiala

Priviind ideile de la punctul 1, am putea incerca sa cautam binar valoarea ultimului timp salvat.
Notam ultimul timp salvat cu T
Astfel, am putea calcula pentru un drum de cate ori trebuie sa il imbunatatim, ajungand la viteza V, astfel incat
imbunatatirea de la V la V + 1 sa fie < T

Acest lucru se poate rezolva tot cu o cautare binara a numarului a vitezei, sau folosind o ecuatie de gradul 2
Notam cu D distanta drumului, trebuie sa ii gasim viteza V
D / (V * (V + 1)) < T
D / T < V * (V + 1))
0 < V * V + V - D / T

Daca vrem V minim, ajungem la ecuatia
0 = V * V + V - D / T

delta = 1 + 4 * D / T

V1,2 = (-1 +- sqrt(1 + 4 * D / T)) / 2
cum V e pozitiv, valoarea lui va fii

V = -0.5 + sqrt(1 + 4 * D / T) / 2

V trebuie rotunjit in sus, deoarece daca V = 3.3 inseamna ca pentru V = 3 inca imbunatatirea de la 3 la 4 este mai buna decat T

Astfel, puteam calcula in O(N) de cate imbunatatiri am avea nevoie pentru un timp de imbunatatire fixat.
Cautarea binara se va face pe numere reale, la fel ca cea pe intregi, doar ca pe reale.
In loc de stanga < dreapte se vor face 40 de iteratii de exemplu.

La final aducem toate drumurile sa respecte valoarea cautata binar.
Daca stim ca pentru 1.5000000 avem nevoie de 10 imbunatiri, dar pentru 1.5000001 avem nevoie de 15 imbunatatiri,
avand 13 imbunatatiri posibile, putem spune ca o sa mai imbunatatim 3 drumuri, fiecare drum salvand 1.5000001 secunde.
Nu conteaza exact ce drumuri imbunatatim. Stim ca ele exista, si ca sunt fix 5 care au acea proprietate.
Acest lucru se intampla cand avem multe drumuri cu aceeasi distanta, sau cu distante proportionale.

Timp final: O(N log X)
Memorie: O(N)

Daca nu facem abstractie de faptul ca functia sqrt este inceata, si ca in practica face tot O(log), timpul este O(N log X log_de_la_sqrt)