Problema 1 datorii – Descriere a unei soluții posibile

prof. Emanuela Cerchez, Colegiul Naţional "Emil Racoviţă" Iaşi

Vom citi datoriile firmelor succesiv, linie cu linie, într-un şir de caractere **`s`**.

Suntem interesaţi doar de firmele distincte, ca urmare vom reţine evidenţa firmelor distincte într-un vector **`F`**. O firmă va fi o structură cu 3 câmpuri (un şir de caractere reprezentând numele acesteia şi două numere naturale **`SD`** şi **`SP`**, reprezentând totalul datoriilor firmei către alte firme, respectiv suma totală pe care firma trebuie să o primească de la alte firme).

Vom prelucra şirul **`s`**, care reprezintă o datorie, în scopul de a extrage numele celor două firme (**`X`** şi **`Y`**) şi suma datorată **`S`**. Deoarece numele firmelor pot conţine şi ele cifre, vom parcurge şirul **`s`** de la sfârşit către început, construind suma **`S`**, din cifrele întâlnite până la primul spaţiu. Eliminăm din şirul **`s`** suma **`S`**, prin trunchierea acestuia. Şirul **`s`** va avea acum forma **`X > Y`**. Separăm şirul **`s`** în două subşiruri (unul care să reprezinte numele firmei **`X`**, celălalt numele firmei **`Y`**), ştiind că cele două nume sunt separate prin **`>`**. Vom căuta numele firmei **`X`** în vectorul **`F`**. Dacă acesta există deja în vectorul **`F`**, doar adăugăm suma **`S`** la câmpul **`SD`** al firmei cu numele **`X`**. Dacă nu există o firmă cu numele **`X`**, o adăugăm în vectorul **`F`**, iniţializând **`SD`** cu **`S`** şi **`SP`** cu **`0`**.

În mod similar procedăm cu firma **`Y`**: o căutăm în vectorul **`F`**; dacă există deja adăugăm suma **`S`** la câmpul **`SP`** al firmei cu numele **`Y`**; dacă nu există o firmă cu numele **`Y`**, o adăugăm la vectorul **`F`** şi iniţializăm câmpul **`SD`** cu **`0`**, iar **`SP`** cu **`S`**.

Căutarea numelui unei firme se poate face secvenţial în vectorul **`F`**, dar în acest caz se obţine punctaj parţial, din cauza depăşirii timpului de execuţie.

Observaţie: pentru teste valorând 45 de puncte, numele firmelor nu conţin spaţii şi deci etapa de prelucrare a şirului **`s`** este eliminată, citind separat **`X`**, **`Y`**, **`S`**.

Pentru un algoritm eficient, va fi necesar să facem o căutare binară în vectorul **`F`**. Pentru ca acest lucru să fie posibil, ar trebui ca, la fiecare pas, vectorul **`F`** să fie deja sortat lexiicografic după numele firmelor. Pentru aceasta, vom face o sortare prin inserare. Mai exact, de fiecare dată când adăugăm o nouă firmă în vectorul **`F`**, căutăm locul corect şi o inserăm direct în poziţia corectă, astfel încât **`F`** să rămână sortat.

Problema 1 triunghi – Descriere a unei soluții posibile

prof. Alin Burţa, Colegiul Naţional "B. P. Haşdeu" Buzău

Soluția 1. (30 de puncte)

Este o abordare brute-force în care, pentru fiecare întrebare Q, vom calcula suma elementelor din triunghiul determinat de tripletul `lin col k`, parcurgând liniile acestuia în funcție de tipul triunghiului (`k > 0` sau `k < 0`). La fiecare zonă vom reactualiza, dacă e cazul, suma maximă.
Complexitatea algoritmului este $O(\,Q \times k \times k + n \times n)$ deoarece, pentru fiecare zonă, parcurgem toate cele $k \times (k+1)/2$ elemente ale triunghiului respectiv, la care se adaugă parcurgerea tabloului la citirea elementelor.
Această soluție obține 30 de puncte.

Soluția 2. (70 de puncte)

Soluția aceasta reprezintă o îmbunătățire a soluției precedente prin reducerea numărului operațiilor necesare calculării sumei elementelor dintr-un triunghi.
Vom precalcula un tablou bidimensional `Slin`, cu următoarea semnificație:
`Slin[i][j]` = suma primelor `j` elemente de pe linia `i`
Pe baza tabloului `Slin` putem afla în timp constant suma elementelor unei linii din triunghiul determinat de tripletul `lin col k`. De exemplu, presupunând `k > 0`, suma elementelor din triunghi se reduce la:

```
SumTriunghi = (Slin[lin][col + k - 1] – Slin[lin][col-1]) +
(Slin[lin+1][col + k -2] – Slin[lin][col-1])          + ... +
(Slin[lin + k - 1][col] – Slin[lin + k – 1][col-1])
```

Pentru cazul `k < 0` se procedează similar.
Complexitatea algoritmului scade la $O(\,Q \times k + n \times n)$

Soluția 3. (90 de puncte)

Pornim de la ideea de a calcula suma elementelor din triunghiul determinat de tripletul `lin col k ( k > 0)` în timp constant. Vom defini trei tablouri bidimensionale de ordin `n`: `sum`, `slin` și `sdiag`, având semnificația:

`sum[i][j]` = suma elementelor din zona dreptunghiulară având colțul din stânga-sus de coordonate `(1, 1)` și colțul din dreapta-jos de coordonate `(i, j)`

`slin[i][j]` = suma primelor `j` elemente de pe linia `i`;

`Sdiag[i][j]` va fi suma elementelor dintr-o zonă determinată astfel:

1. Dacă ipotenuza triunghiului se găsește de-a lungul unei diagonale aflate deasupra diagonalei principale, zona va fi de forma unui trapez dreptunghic determinat de punctele `(1, 1), (i, 1), (i, j), (1, i+j-1)` – *vezi figura 1*.
2. Dacă ipotenuza triunghiului se găsește de-a lungul unei diagonale aflate sub diagonala principală, zona va fi de forma unui pentagon determinat de punctele `(1, 1), (i, 1), (i, j), (x, n) și (1, n)`, unde `(x, n)` este punctul în care prelungirea ipotenuzei triunghiului intersectează ultima coloană (*vezi figura 2*).

	1	2	3	4	5	6
1	5	8	10	4	9	4
2	2	10	10	2	4	8
3	8	10	3	4	6	6
4	4	6	9	7	1	9
5	6	7	2	2	10	6
6	10	4	6	1	10	4

Figura 1

	1	2	3	4	5	6
1	5	8	10	4	9	4
2	2	10	10	2	4	8
3	8	10	3	4	6	6
4	4	6	9	7	1	9
5	6	7	2	2	10	6
6	10	4	6	1	10	4

Figura 2

Valorile tabloului `Sdiag` se calculează folosind tablourile `sum` și `slin`:

```
for(i = 1; i <= N; ++i)
    diag[1][i] = slin[1][i];
for(i = 2; i <= N; ++i)
{
    for(j = 1; j < N; ++j)
        diag[i][j] = diag[i-1][j+1] + slin[i][j];
    diag[i][N] = sum[i][N];
}
```

Pe baza tablourilor precalculate, aflarea sumei valorilor din triunghiul determinat de tripletul `i j k`, cu `k > 0`, se realizează, în general, cu formula:

`Suma=sdiag[i+k-1][j]-sdiag[i-1][j+k]-sum[i+k-1][j-1]+sum[i-1][j-1]`

Mai exact, scădem din suma întregii zone (zona1 + zona2 + zona 3 + triunghiul curent) zona compusă din alipirea zonei 1 cu zona 3 și a zonei compuse din alipirea zonei 1 și zonei 2, la care adăugăm zona 1, care a fost scăzută de două ori (*vezi figura 3*).

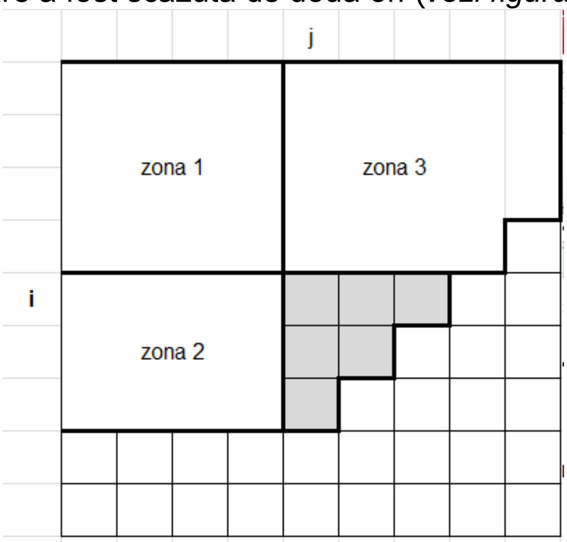

Figura 3

Formula de calcul suferă unele modificări în funcție de poziția triunghiului și de forma zonei. Dacă triunghiul are `k < 0` (triunghi cu unghiul drept *"în jos"*), suma elementelor sale se calculează ca fiind diferența dintre suma elementelor din pătratul de latură `k`, având colțul din dreapta-jos de coordonate `(i, j)` și suma elementelor triunghiului determinat de tripletul `i-|k|+1 j-|k|+1, |k|` (calculat prin procedeul descris anterior).

Complexitatea algoritmului este $O(Q + n \times n)$.