Şiruri - Soluţie

prof. Constantin Creţu, Colegiul Naţional

```
int LevenshteinDistance(char s[1..m], char t[1..n])
 {

   declare int d[0..m, 0..n]

   for i from 0 to m
     d[i, 0] := i //stergere
   for j from 0 to n
     d[0, j] := j //inserare

   for j from 1 to n
   {
     for i from 1 to m
     {
       if s[i] = t[j] then
         cost:=0
       else
         cost:=1
       d[i, j] := minimum(d[i-1, j] + 1, d[i, j-1] + 1, d[i-1, j-1]+cost)
     }
   }
   return d[m, n]
 }
```

Algoritmul poate folosi o matrice de dimensiune mxn (m dimensiunea primului sir, iar n dimensiunea celui de-al doilea sir) sau prin operatia de copiere a coloanei a doua peste prima se poate folosi doar o matrice de dimensiune mx2.

In cazul algoritmului cu matricea de dimensiune mxn se initializeaza prima linie si prima coloana cu valoarea indicelui randului respectiv coloanei, deoarece s [1 .. i] poate fi transformat în şirul gol t [1 .. 0], prin eliminarea tuturor caracterelor. În mod similar, se poate transforma s [1 .. 0] in t [1 .. j] prin simpla adăugare a tuturor caracterelor.

Minimul este de luat din trei distanţe:

1. Dacă se poate transforma s [1 .. i] in t [1 .. j-1] în k operaţii, atunci putem adăuga pur şi simplu t [j] după aceea pentru a obţine t [1 .. j], în k+1 operaţii.
2. Dacă se poate transforma s [1 .. i-1] in t [1 .. j] în k operaţii, atunci putem face aceleaşi operaţii pe s [1 .. i] şi apoi sa scoatem originalul s [i] la sfârşit în k+1 operaţii.
3. Dacă se poate transforma s [1 .. i-1] in t [1 .. j-1] în k operaţii, atunci putem face acelaşi lucru la s [1 .. i] şi apoi sa facem o substitutie a lui t [j] cu originalul s [i] la sfârşit, dacă este necesar, care necesită k + cost operaţii. Costul poate fi 0 sau 1, in functie de egalitatea intre s[i] si t[j].

Operaţiile necesare pentru a transforma s [1 .. n] în t [1 .. m] reprezinta numărul necesar de a transforma toate caracterele din s în cele din t, astfel incat d [n, m] deţine rezultatul nostru.

	OLIMPIADA DE INFORMATICĂ Etapa municipală – IAŞI, 20 februarie 2010	Clasele XI-XII

Descrierea soluţiei

prof. Emanuela Cerchez

Să notăm cu A şi B cele două şiruri de caractere, de lungime n, respectiv m.
Vom nota cu Ai sufixul şirului A care începe la poziţia i ($0 \leqslant i < n$), iar cu Bj sufixul şirului B care începe la poziţia j ($0 \leqslant j < m$).
Vom rezolva problema prin programare dinamica

Subproblemă:
Să se determine numărul minim de operaţii necesare pentru transformare sufixului Ai în sufixul Bj.

Relaţia de recurenţă:
Nrmin(i,j)=min {
1+Nrmin(i+1,j)//sterg caracterul de pe poziţia i din A
1+Nrmin(i,j+1)//inserez caracterul de pe poziţia j din B
1+NrMin(i+1,j+1)//inlocuiesc caracterul de pe poziţia i cu caracterul de pe poziţia j
}, dacă A[i]≠B[j]
Dacă A[i]=B[j], atunci NrMin[i][j]=NrMIn[i+1][j+1].

Evident,
Nrmin(i,m)=n-i //stergem ultimele n-i caractere din A, $0 \leqslant i < n$
Nrmin(n,j)=m-j //inseram ultimele m-j caractere din B, $0 \leqslant j < m$
Nrmin (n,m)=0;
Soluţia problemei se obţine în Nrmin(0,0).