Descriere soluţie - **problema palindrom**:

Se precalculează elementele matricei ajunge, după relaţia

`ajunge[i][j]` = numărul minim de transformări ce pot fi efectuate pornind de la cifra `i` pentru a obtine cifra `j`.

Având aceasta matrice calculată, primul subpunct este trivial, rămâne de văzut pentru fiecare pereche de cifre din şirul iniţial (`i, N-i+1`), cu `i` de la `1` la `N/2` (unde N = lungimea şirului) care este cea mai mare cifră care se poate obţine prin transformări repetate, atât din A[i], cât şi din `A[N-i+1]` (am notat cu `A` şirul iniţial de lungime `N`).

Pentru al doilea subpunct vom folosi o recurenţă începând de la jumătatea şirului spre început, `D[i][j]` = numărul minim de aplicări pentru care şirul cuprins intre poziţiile `i` şi `N-i+1` este palindrom şi dă restul `j` la împărţirea prin `K`. Dacă pe poziţiile (`i, N-i+1`) punem cifra `q`, atunci restul obţinut la pasul anterior `S` devine `(S + q * (10^(N-i) + 10^(i-1)) % K`. Astfel se deduce relaţia de recurenţă:

```
if(ajunge[ A[i] ][p] >= 0 && ajunge[ A[N-i+1] ][p] >= 0) {
      //se poate ajunge la cifra p
      //vechiul rest este j
       int nou_rest = (j + p * (put_zece[N-i] + put_zece[i-1])) % K;
      D[i][nou_rest] = min(D[i][nou_rest], D[i+1][j] + ajunge[cif1][p]
                        + ajunge[cif2][p]);
}
```

unde i ia valori de la `N/2` la `1`, `j` ia valori de la `1` la `K` şi `p` ia valori de la `1` la `9`. Astfel vom avea în `D[1][0]` numărul minim de aplicări.

Pentru a reconstitui palindromul maxim divizibil cu `K`, repetăm procedeul plecând de la început spre mijloc (pentru a ne asigura că se obţine cel mai mare palindrom), reţinând poziţia pe care ne aflăm şi restul corespunzător secvenţei de palindrom rămase.

Complexitatea algoritmului: `O(N * K)`.

Autori: Student Robert Hasna
Student Vlad Ionescu
Student Cosmin Tutunaru
Student Vlad Duţă
Student Dragoş Oprică
Student Alexandru Cazacu