Problema 2018.1.1 – Blockchain hash

Pentru a deveni competitivă în domeniul Blockchain și a monedelor virtuale, firma la care lucrați a dezvoltat un algoritm de hashing inovativ pentru valori numerice. Algoritmul funcționează în modul următor: asupra unui număr natural se face următoarea transformare: dacă numărul are cel puțin două cifre, se iau pe rând din număr câte două cifre vecine și se scade cea mai mică din cea mai mare. Cu cifrele astfel obținute se formează un nou număr. De exemplu, pentru numărul `5734`, din cifrele 5 și 7 se obține 2, din 7 și 3 se obține 4 iar din 3 și 4 se obține 1. Formăm deci un nou număr `241`, căruia i se poate aplica aceeași transformare, obținându-se `23`. Din `23`, prin același procedeu, obținem `1`. Dacă numărul este format dintr-o singură cifră, transformarea îl lasă nemodificat. Hash-ul se calculează realizând succesiv transformarea asupra numărului original și apoi asupra rezultatului obținut, de `k` ori, și sumând toate rezultatele parțiale și rezultatul final (excluzând valoarea de start).

Cerință

Se dau două numere naturale `N` şi `k` și apoi încă `N` numere naturale. Se cere să se determine valoarea maximă a hash-ului care se va obține aplicând algoritmul de hash celor `N` numere folosind `k` pentru numărul de iterații de transformare, conform descrierii de mai sus.

Date de intrare

La intrarea programului se prezintă pe prima linie două numere naturale `N` și `k`, separate prin spațiu. Pe a doua linie se află cele `N` numere, separate tot prin spații. Liniile de intrare se încheie cu caracterul *newline* (`\n`), obținut prin apăsarea tastei *Enter*.

Date de ieşire

Programul va afișa la consolă (pe stream-ul *stdout*) un singur număr, reprezentând valoarea maximă dintre toate hash-urile obținute conform procedeului descris anterior.

ATENŢIE la respectarea cerinței problemei: afișarea rezultatelor trebuie făcută EXACT în modul în care a fost indicat! Cu alte cuvinte, pe stream-ul standard de ieșire nu se va afișa nimic în plus față de cerința problemei; ca urmare a evaluării automate, orice caracter suplimentar afișat, sau o afișare diferită de cea indicată, duc la un rezultat eronat şi prin urmare la obținerea calificativului „Respins”.

Restricţii

1. $0 < N < 10000$
2. $0 \leq n_i < 100000000$ (valorile asupra cărora se va aplica hash-ul).
3. $0 < k < 100$
4. **Atenție**: În funcție de limbajul de programare ales, fișierul ce conține codul trebuie să aibă una din extensiile .c, .cpp, .java, sau .m. Editorul web **nu va adăuga automat** aceste extensii şi lipsa lor duce la imposibilitatea compilării programului!
5. **Atenție**: Fișierul sursă trebuie numit de candidat sub forma: <nume>.<ext> unde *nume* este numele de familie al candidatului și extensia (*ext*) este cea aleasă conform punctului anterior. Atenție la restricțiile impuse de limbajul Java legate de numele clasei şi numele fișierului!

Exemple

Intrare	Ieșire	Explicații

`2 2`
`5734`
`1234`
	`264`	Se aplică de 2 ori succesiunea de transformări pentru fiecare din cele două valori.
		Astfel, din `5734` se obține `241`, iar din `241` se obține `23`. Adunând numerele `241` și `23` se obține rezultatul: **`264`**.

		Din `1234` se obține `111` iar din `111` se obține `0`. Sumând cele două valori rezultatul este **`111.`** Valoarea maximă este **`264`**.
`2 3` `2228 6`	`18`	Se aplică transformările de 3 ori asupra celor două numere. Pentru `2228`, în urma primei transformări se obțin cifrele `0, 0, 6`, din care formăm secvența `006`, adică numărul `6`. La următoarele 2 transformări rezultă aceeași valoare, `6`. Prin urmare suma obținută este: `6+6+6 =` **`18`**. Pentru valoarea `6`, cele 3 transformări vor produce tot `6` iar suma va fi tot **`18`**. Rezultatul este deci **`18`**.

Timp efectiv de lucru: 120 de minute