Problema Parola

Fişier de intrare	`input.txt`
Fişier de ieşire	`output.txt`

Compania Einheit oferă un serviciu de stocare de parole. Recent au anunțat o schimbare a politicii de monetizare: fiecare utilizator va fi taxat proporțional cu lungimea totală a parolelor stocate.
Vestea asta ți-ar distruge bugetul, pentru că în calitate de fanatic în securitate, parolele tale sunt distincte și absurd de lungi. Pentru că alte servicii sunt mai greu de folosit, migrarea tuturor parolelor nu e o opțiune încă.
În încercarea ta de a-ți reduce costurile, ai eliminat parolele nenecesare și ți-ai propus să găsești cele mai scurte parole pentru fiecare site rămas.
Pentru fiecare site știi alfabetul minim, lungimea minimă a parolei și ai modelat un automat finit determinist cu regulile lor arbitrare de complexitate pentru parole.
Scrie un program care citește un automat și o lungime minimă și generează o parolă cât mai scurtă.

Cerință

Se dă un automat finit determinist D cu n stări, numerotate de la 1 la n $(q_1, q_2, .., q_n)$, alfabetul Σ, starea inițială q_x, m stări finale $\{q_{s_1}, q_{s_2}, .., q_{s_m}\}$ și funcția de tranziție δ.

Să se găsească un cuvânt w acceptat de D care are lungimea cel puțin k, dar care are lungimea cât mai mică posibilă. În alte cuvinte, w este o soluție validă pentru problemă dacă $|w| \geq k$, $w \in \mathcal{L}(D)$ și $\forall x \in \mathcal{L}(D)$ cu $|x| \geq k$, $|w| \leq |x|$.

Se consideră că Σ este format din primele $|\Sigma|$ litere mici din alfabetul latin.

Date Intrare

Pe prima linie se găsește un număr $t \in \{1, 2, 3\}$, care reprezintă tipul de subtask de care aparține testul.
Pe a doua linie se găsesc 4 numere, n, $|\Sigma|$, m și k.
Pe a treia linie se găsește un singur număr, q_x, starea inițială.
Pe a patra linie se găsesc m numere, $\{q_{s_1}, q_{s_2}, .., q_{s_m}\}$, stările finale.
Pe următoarele n linii se găsesc câte $|\Sigma|$ numere, al j-lea număr de pe linia $4 + i$ $(1 \leq i \leq n, 1 \leq j \leq |\Sigma|)$ reprezintă starea $\delta(q_i, 'a' + j - 1)$ în care se ajunge pornind din starea q_i dacă următorul caracter din input este a j-a literă din alfabetul latin.

Date Ieșire

Dacă nu există niciun cuvânt acceptat de D cu lungimea minim k, afișați -1.

Altfel, pe prima linie trebuie afișat un număr $|w|$, lungimea cuvântului găsit w cu proprietățile cerute în descriere. Pe a doua linie trebuie afișat cuvântul w.

Restricții

- $2 \leq |\Sigma| \leq 26$, $2 \leq n$, $1 \leq k$.
- $1 \leq q_x \leq n$. $\forall i, j, 1 \leq i < j \leq m, q_{s_i} \neq q_{s_j}$.
- Se acceptă soluții doar pentru $|w| \leq 2 \cdot 10^8$. Se garantează că dacă există o soluție validă, aceasta nu depășește limita impusă.

#	Punctaj	Restricţii
1	60	$n \leq 5000$, $k \leq 2 \cdot 10^4$.
2	40	$n \leq 200$, $k \leq 10^8$.
3	25	$n \leq 10^6$, $k \leq 10^8$. Se punctează parțial orice cuvânt w acceptat de D cu $2 \cdot 10^8 \geq \|w\| \geq k$.

- Pentru primul subtask, se acordă 50% din punctaj dacă cuvântul w afișat este acceptat de D cu $2 \cdot 10^8 \geq |w| \geq k$, chiar dacă există un alt cuvânt x acceptat de D cu $k \leq |x| < |w|$.
- Pentru ultimul subtask, formula pentru calculul coeficientului punctajului parțial este $2 \cdot \frac{|w_{ref}|-k+1}{(|w_{ref}|-k+1)+(|w|-k+1)}$, unde w_{ref} este un cuvânt obținut de o soluție a echipei.

Exemple

input.txt	output.txt
1 11 2 1 8 1 10 2 11 3 11 4 11 5 8 6 8 7 11 5 11 9 11 3 10 11 11 11 11	10 aaabaaabab
1 9 2 2 6 1 2 4 2 5 5 3 5 4 9 9 8 6 8 4 8 8 7 7 9 9	-1

Explicații

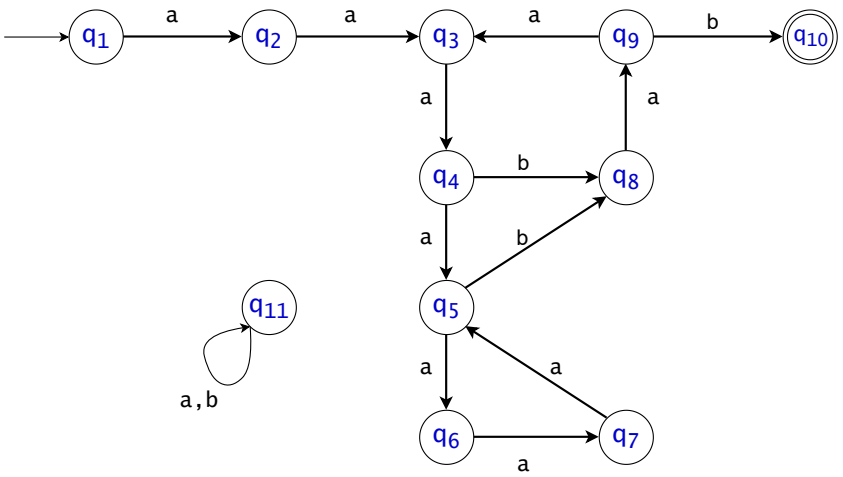

Figura 1: Automatul din exemplul 1 arată astfel. În figură nu sunt reprezentate muchiile $\delta(q_1, b) = \delta(q_2, b) = \delta(q_4, b) = \delta(q_5, b) = \delta(q_6, b) = \delta(q_7, b) = \delta(q_8, b) = \delta(q_{10}, b) = q_{11}$. Un alt cuvânt de lungime 10 valid ar fi a^7bab. Se poate demonstra că nu există niciun cuvânt acceptat de D care să aibă lungimea 8 sau 9.

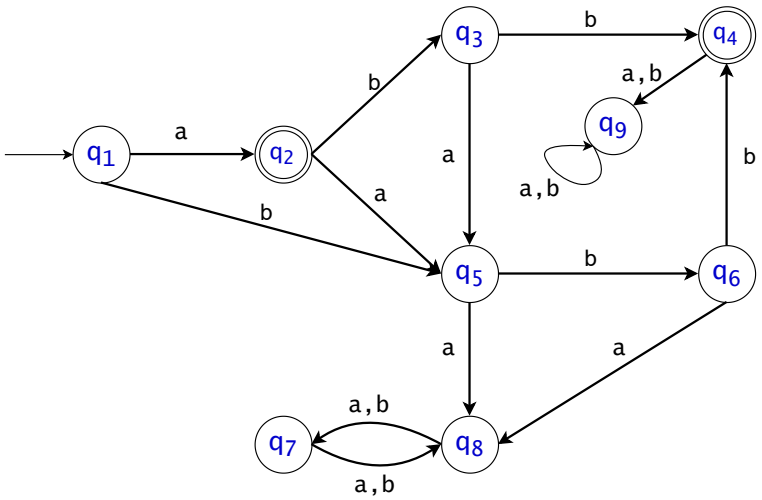

Figura 2: În al doilea exemplu, avem mai multe cicluri în D care sunt accesibile din starea inițială, însă odată ce am intrat într-unul dintre ele, ne este imposibil să mai ajungem ulterior într-o stare finală. Cel mai lung cuvânt acceptat de D are lungimea 5: *ababb*, deci răspunsul pentru $k = 6$ este -1.

Precizări

- Tema va fi testată pe VMChecker. Limita de timp per test este de 120 de secunde, iar pentru toată tema limita este de 800 de secunde. Nu există o limită de memorie pentru heap. Mașina virtuală are 4 GB de memorie alocați.
- În general, timpul de rulare pe VMChecker este în jur de 2 ori mai mare decât pe mașina locală.
- `Makefile`-ul trebuie să se afle neapărat în rădăcina arhivei incărcate pe VMChecker.
- Dimensiunea stivei este mărită de checker (doar în timpul rulării lui) la 256 MB. Dacă vă testați independent de checker programul, puteți să vă măriți dimensiunea stivei în sesiunea curentă de terminal cu `ulimit -s 262144`. Puteți să verificați dimensiunea stivei cu `ulimit -a | grep stack`.
- Checker-ul poate fi invocat astfel: `./checker checker_info.txt ../src/`, unde `checker_info.txt` este un fișier care descrie toate testele pe care va fi rulată sursa. Checker-ul va încerca să apeleze `make` din folder-ul dat ca argument (`../src/`), deci trebuie neapărat să existe un `Makefile` acolo cu regulile `build`, `run` și `clean`.
- Puteți să folosiți orice flag de optimizare doriți.