

Double Agents (trees)


L'agenzia di spionaggio britannica MI6 sta progettando di infiltrare degli agenti doppiogiochisti nell'organizzazione criminale SPECTRE. SPECTRE ha N dipendenti, numerati da 0 a $N - 1$ e il loro organigramma forma un albero.

L'MI6 selezionerà un insieme non vuoto S di dipendenti SPECTRE da trasformare in infiltrati. A causa del rischio di ulteriori tradimenti (ovvero possibili agenti triplogiochisti), è essenziale che la catena di comunicazione tra due qualsiasi agenti infiltrati passi solo attraverso dipendenti 'innocenti'. Ovvero, per ogni coppia di dipendenti distinti a e b in S , il percorso semplice tra a e b non deve contenere altri dipendenti di S .

Il tuo compito è contare il numero di possibili set S di agenti infiltrati. Dato che questo valore può essere molto grande, restituisci il valore modulo $10^9 + 7$.

Implementazione

Dovrai inviare un singolo file sorgente `.cpp`.

 Tra gli allegati di questo problema troverai un template `trees.cpp` con un esempio di implementazione.

Dovrai implementare la seguente funzione:

```
C++ | int count_sets(int N, vector<int> U, vector<int> V);
```

- L'intero N rappresenta il numero di dipendenti di SPECTRE.
- Gli array U e V , indicizzati da 0 a $N - 2$, contengono i valori U_0, U_1, \dots, U_{N-2} e V_0, V_1, \dots, V_{N-2} , dove U_i e V_i sono gli estremi dell' i -esimo arco nell'albero dell'organizzazione.
- La funzione deve restituire il numero di possibili insiemi modulo $10^9 + 7$.

Il grader chiamerà la funzione `count` e stamperà il valore restituito sul file di output.

Grader di prova

La cartella del problema contiene una versione semplificata del grader, che puoi usare per testare la tua soluzione in locale. Il grader semplificato legge i dati di input da `stdin`, chiama le funzioni che devi implementare, e scrive l'output su `stdout`.

- Riga 1: l'intero N .
- Riga $2 + i$ ($0 \leq i \leq N - 2$): gli interi U_i e V_i .

Assunzioni

- $2 \leq N \leq 500\,000$.
- $0 \leq U_i, V_i \leq N - 1$.
- I nodi formano un albero valido.

Assegnazione del punteggio

Il tuo programma verrà testato su più test case raggruppati in subtask. Per ottenere il punteggio associato a un subtask, devi risolvere correttamente tutti i casi di test che contiene.

- **Subtask 1** [0 punti]: Casi di esempio.
- **Subtask 2** [20 punti]: $N \leq 16$.
- **Subtask 3** [15 punti]: L'albero è una linea. Una linea è un albero in cui i nodi possono essere disposti in un certo ordine P_0, P_1, \dots, P_{N-1} in cui esiste un arco tra i nodi P_i e P_{i+1} per ogni $0 \leq i \leq N - 2$.
- **Subtask 4** [15 punti]: $N \leq 500$, e l'albero è un bruco. Un bruco è una linea con alcuni nodi foglia aggiunti. Ovvero, i vertici dell'albero possono essere scritti come $P_0, \dots, P_k, Q_0, \dots, Q_l$, dove P_0, \dots, P_k è una linea e ogni Q_i è un nodo foglia (ovvero ha grado 1).
- **Subtask 5** [15 punti]: $N \leq 5000$, e l'albero è un bruco.
- **Subtask 6** [10 punti]: $N \leq 500\,000$, e l'albero è un bruco.
- **Subtask 7** [25 punti]: Nessuna limitazione aggiuntiva.

Esempi di input/output

stdin	stdout
3 0 1 1 2	6
5 0 1 0 2 1 3 1 4	17

Spiegazione

Nel **primo caso esempio**, l'albero è una linea $(0 - 1 - 2)$. Ci sono 6 insiemi possibili in questo albero: $\{0\}, \{1\}, \{0, 1\}, \{2\}, \{0, 2\}, \{1, 2\}$.

L'insieme $\{0, 1, 2\}$ non è consentito poiché 1 si trova sul percorso semplice tra 0 e 2.

Nel **secondo caso esempio**, l'albero è una linea $(3 - 1 - 0 - 2)$ con una singola foglia aggiuntiva (4). Ci sono 17 possibili insiemi in questo albero.