Super Marco 2 (monete)

Limite di tempo:	1.0 secondi
Limite di memoria:	256 MiB

William sta di nuovo giocando a Super Marco 64, e stavolta per superare il livello deve raccogliere il maggior numero possibile di monete. Il livello è composto da una serie di piattaforme collegate tra loro. Su ciascuna piattaforma possono esserci delle monete e, appena Super Marco raggiunge una certa piattaforma, raccoglie istantaneamente tutte le monete che essa contiene.

Data la mappa del livello, e sapendo che Super Marco si trova nella piattaforma 0, determina quante monete al massimo è possibile raccogliere.

Implementazione

Dovrai sottoporre esattamente un file con estensione `.c`, `.cpp` o `.pas`.

> ☞ Tra gli allegati a questo task troverai un template (`monete.c`, `monete.cpp`, `monete.pas`) con un esempio di implementazione da completare.

Se sceglierai di utilizzare il template, dovrai implementare la seguente funzione:

C/C++	`int raccogli(int N, int M, int monete[], int A[], int B[]);`
Pascal	`function raccogli(N, M: longint; var monete, A, B: array of longint): longint;`

In cui:

- Gli interi N ed M rappresentano rispettivamente il numero di piattaforme ed il numero di collegamenti tra di esse.
- L'array `monete`, indicizzato da 0 a $N-1$, contiene il numero di monete che si trovano in ciascuna piattaforma.
- Gli array `A` e `B`, indicizzati da 0 a $M-1$, identificano una coppia di piattaforme `A[i]` e `B[i]` per le quali esiste un collegamento diretto.
- La funzione dovrà restituire il massimo numero di monete che si possono raccogliere in totale. Tale numero verrà stampato sul file di output.

Dati di input

Il file `input.txt` è composto da due righe. La prima riga contiene i due interi N e M separati da uno spazio. La seconda riga contiene gli N interi `monete[i]` separati da spazi. Ciascuna delle successive M righe contiene una coppia di interi `A[i]` e `B[i]`.

Dati di output

Il file `output.txt` è composto da un'unica riga contenente un unico intero, la risposta a questo problema.

Assunzioni

- $1 \leq N \leq 10\,000$.

- $0 \leq M \leq 100\,000$.
- I collegamenti sono bidirezionali.
- $0 \leq$ `monete[i]` ≤ 1000 per ogni $i = 0 \ldots N-1$.
- $0 \leq$ `A[i]`, `B[i]` $\leq N-1$.
- `A[i]` $\neq$ `B[i]` e non ci sono collegamenti duplicati.

Assegnazione del punteggio

Il tuo programma verrà testato su diversi test case raggruppati in subtask. Per ottenere il punteggio relativo ad un subtask, è necessario risolvere correttamente tutti i test relativi ad esso.

- **Subtask 1** [**10 punti**]: Casi d'esempio.
- **Subtask 2** [**30 punti**]: Il grafo è connesso (tutte le piattaforme sono raggiungibili).
- **Subtask 3** [**30 punti**]: Tutte le piattaforme hanno esattamente 1 moneta.
- **Subtask 4** [**30 punti**]: Nessuna limitazione specifica.

Esempi di input/output

input.txt	output.txt
5 3 1 2 1 2 1 0 2 4 1 3 0	4
1 0 1000	1000

Spiegazione

Nel **primo caso di esempio** si può raccogliere immediatamente una moneta, dopodiché è possibile raggiungere la piattaforma 2 raccogliendo un'altra moneta, per poi tornare indietro e recarsi alla piattaforma 3 che ha ben due monete. In totale possiamo raccogliere quindi 4 monete. Purtroppo non c'è alcun modo di raggiungere le piattaforme 1 e 4.

Nel **secondo caso di esempio** c'è una sola piattaforma e non è possibile quindi spostarsi.