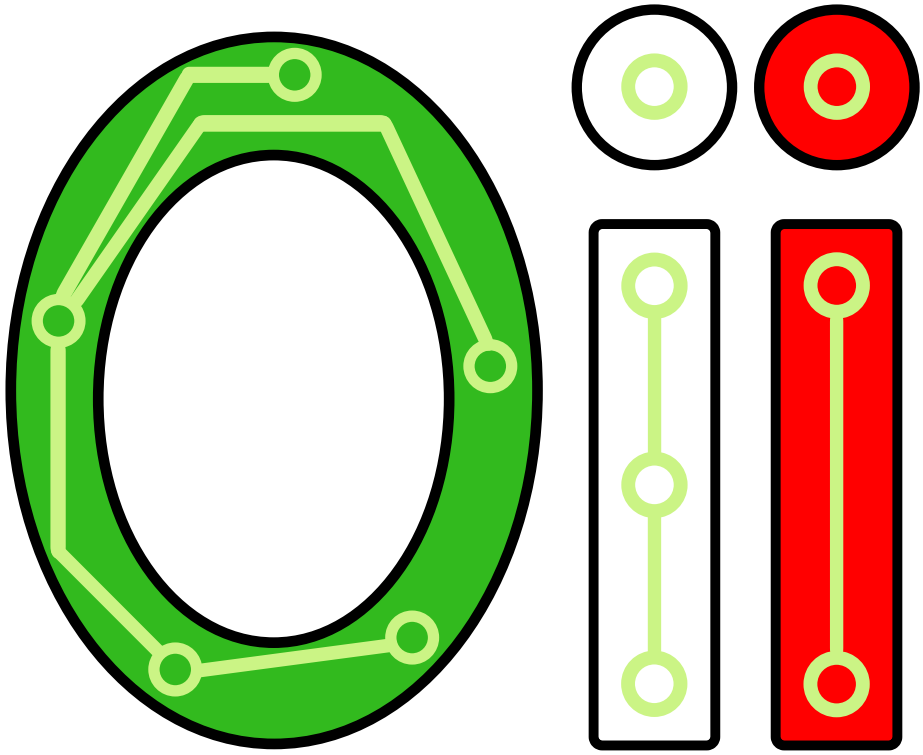
AICA
Associazione Italiana per l'Informatica
ed il Calcolo Automatico

Ministero dell'Istruzione
dell'Università e Ricerca



Olimpiadi Italiane di Informatica

Selezione Territoriale 2018

18 Aprile 2018

Testi dei problemi

Testi dei problemi

Giorgio Audrito, Massimo Cairo, William Di Luigi, Luigi Laura, Gemma Martini, Edoardo Morassutto, Dario Ostuni, Romeo Rizzi, Luca Versari

Coordinamento

Monica Gati

Supervisione a cura del Comitato per le Olimpiadi di Informatica

Festa canina

Mojito, il cane di Monica, vuole organizzare una festa con i suoi amici. Aiutalo a scegliere quali invitare e quali escludere in modo da rendere la festa più bella possibile.

Dettagli

Mojito ha N amici, ognuno dei quali ha un grado di amicizia A_i che indica quanto Mojito apprezzi la sua presenza. Ovviamente nella lista c'è anche qualche antipatico che quindi ha un grado di amicizia negativo.

La bellezza della festa è definita come la somma del grado di amicizia degli invitati. Quanto può valere al massimo questa somma?

Assunzioni

- $T = 6$, sono presenti 6 casi di input.
- $1 \leq N \leq 10.000$, il numero di amici di Mojito.
- $-100 \leq A_i \leq 100$, il grado di amicizia dell'i-esimo amico.
- È anche possibile che Mojito festeggi senza amici, in tal caso la festa ha valore 0.

Nota bene: se utilizzi il linguaggio **Pascal**, fai attenzione al fatto che il valore massimo contenuto in una variabile `integer` è 32767, troppo piccolo per risolvere questo task completamente. Superando quel numero, infatti, il programma comincerà a salvare numeri imprevedibili (senza mostrarti alcun errore!) per via dell'*overflow*. Per evitare questo fenomeno ti consigliamo di usare **sempre** il tipo `longint` al posto di `integer`.

Dati di input

La prima riga del file di input contiene un intero T, il numero di casi di test. Seguono T casi di test, numerati da 1 a T. Ogni caso di test è preceduto da una riga vuota.

In ciascun caso di test, la prima riga contiene l'unico intero N.

La seconda riga contiene gli N interi separati da spazi, A_i.

Dati di output

Il file di output deve contenere la risposta ai casi di test che sei riuscito a risolvere. Per ogni caso di test che hai risolto, il file di output deve contenere una riga con la dicitura:

```
Case #t: k
```

dove `t` è il numero del caso di test (a partire da 1) e `k` è il massimo valore di bellezza ottenibile.

Esempi di input/output

Input: Download

```
2

8
1 -4 5 -2 -1 8 0 1

3
-1 -2 -4
```

Output: Download

```
Case #1: 15
Case #2: 0
```

Spiegazione

Nel **primo caso d'esempio** la soluzione si ottiene invitando il primo, il terzo, il sesto, il settimo e l'ottavo amico, totalizzando una somma di 15.

Nel **secondo caso d'esempio** la soluzione si ottiene non invitando alcun amico, totalizzando quindi 0.

Esempi di implementazione

Per aiutarti con questo task, abbiamo preparato degli esempi di implementazione di una soluzione. Includono solo le parti di lettura dell'input e scrittura dell'output:

- Versione C: Download
- Versione C++: Download
- Versione Pascal: Download

Antivirus

Il nuovo sistema di gara delle Selezioni Territoriali funziona alla grande, ma Mojito non è così convinto... sembra infatti che la nota mascotte delle Olimpiadi abbia fiutato un **virus** nascosto fra i file inviati da un partecipante!

Conosciamo la lunghezza del virus e sappiamo che si ripete uguale nei quattro file che abbiamo ricevuto, ma non sappiamo dove. Aiutaci ad individuare il virus!

Dettagli

I quattro file F_1, F_2, F_3, F_4 sono dati in input, rappresentati come quattro stringhe di caratteri di lunghezza rispettivamente N_1, N_2, N_3, N_4.

Il virus è una stringa di caratteri V di lunghezza M. La lunghezza M è data in input, ma non si conosce il contenuto della stringa V del virus.

Sappiamo con certezza che il virus V appare all'interno di tutti e quattro i file, come sottostringa di caratteri *consecutivi*. Sappiamo inoltre che *NON* ci sono altre sottostringhe consecutive di lunghezza M che si ripetono uguali in tutti e quattro i file.

Le posizioni dei caratteri nelle stringhe sono numerati a partire da 0. Per ciascuno dei quattro file F_i, trova la posizione in cui è inserito il virus, ovvero la posizione dove appare il primo carattere del virus V all'interno della stringa F_i.

Assunzioni

- $T = 12$, sono presenti 12 casi di input.
- $2 \leq N_1, N_2, N_3, N_4 \leq 100$, i file non sono più lunghi di 100 caratteri.
- $2 \leq M \leq 20$, il virus non è più lungo di 20 caratteri.
- $M \leq \min(N_1, N_2, N_3, N_4)$, il virus non è più lungo del file più corto.
- Tutti i caratteri dei file sono lettere minuscole dell'alfabeto inglese (dalla a alla z), *NON* sono presenti spazi.
- È garantito che il virus esiste ed è unico.

Dati di input

La prima riga del file di input contiene un intero T, il numero di casi di test. Seguono T casi di test, numerati da 1 a T. Ogni caso di test è preceduto da una riga vuota.

In ciascun caso di test:

- La prima riga contiene quattro interi, N_1, N_2, N_3 e N_4, separati da uno spazio, che corrispondono alla lunghezza di ciascuno dei quattro file.
- La seconda riga contiene un solo intero M, che corrisponde alla lunghezza del virus.
- Le successive 4 righe contengono rispettivamente le quattro stringhe F_1, F_2, F_3 e F_4.

Dati di output

Il file di output deve contenere la risposta ai casi di test che sei riuscito a risolvere. Per ogni caso di test che hai risolto, il file di output deve contenere una riga con la dicitura

```
Case #t: p1 p2 p3 p4
```

dove `t` è il numero del caso di test (a partire da 1) e i valori `p1`, `p2`, `p3`, `p4` sono le posizioni in cui si trova il virus in ciascuno dei quattro file. Con posizione si intende l'indice del primo carattere del virus, il primo carattere del file ha indice zero.

Esempi di input/output

Input: Download

```
2

8 12 10 7
4
ananasso
associazione
tassonomia
massone

6 9 11 10
3
simone
ponessimo
milionesimo
cassonetto
```

Output: Download

```
Case #1: 4 0 1 1
Case #2: 3 1 4 4
```

Spiegazione

Nel **primo caso d'esempio** il virus è *asso*: anan**asso**, **asso**ciazione, t**asso**nomia, m**asso**ne.

Nel **secondo caso d'esempio** il virus è *one*: sim**one**, p**one**ssimo, mili**one**simo, cass**one**tto. Nota che *sim* è presente nei primi tre file ma non nel quarto, quindi non è il virus cercato.

Esempi di implementazione

Per aiutarti con questo task, abbiamo preparato degli esempi di implementazione di una soluzione. Includono solo le parti di lettura dell'input e scrittura dell'output:

- Versione C: Download
- Versione C++: Download
- Versione Pascal: Download

Radioanalisi fossile

È stato appena ritrovato un fossile della rarissima specie *Canis mojitus albus*, ritenuta antenata della più comune *Canis mojitus familiaris*. Per analizzarlo, gli scienziati devono trattarlo con delle radiazioni: ogni centimetro dell'osso deve riceverne una precisa quantità. La macchina che fa il trattamento può applicare radiazioni in modo uniforme su un qualsiasi segmento contiguo: calcola quante volte deve essere azionata la macchina per ottenere la giusta quantità di radiazioni su ogni punto dell'osso.

Dettagli

L'osso da trattare è lungo N centimetri, numerati da 1 a N. Il centimetro i deve ricevere una quantità di radiazioni specificata da un numero naturale R_i. Il numero N ed i numeri $R_1, ..., R_N$ sono dati in input.

La macchina viene azionata specificando due numeri interi positivi a e b, che indicano gli estremi del segmento di osso su cui la macchina opera ($a \leq b$). Dopo tale azionamento, tutti i centimetri da a a b dell'osso accumulano 1 unità di radiazioni.

Dopo aver azionato la macchina un certo numero di volte, la quantità di radiazioni ricevute sul centimetro i si può conoscere contando quante volte una radiazione ha operato su quella zona (ovvero, quante volte la macchina è stata azionata con valori tali per cui $a \leq i \leq b$).

Calcola il numero minimo di volte in cui è necessario azionare la macchina affinché ciascuna zona i riceva *esattamente* la quantità di radiazioni richiesta R_i.

Assunzioni

- $T = 19$, ci sono 19 casi di prova.
- $1 \leq N \leq 1000$, ovvero, l'osso è lungo al massimo 1000 centimetri.
- $0 \leq R_i \leq 1000$, ogni centimetro può dover ricevere una quantità di radiazione fino a 1000 .

Dati di input

La prima riga del file di input contiene un intero T, il numero di casi di test. Seguono T casi di test, numerati da 1 a T. Ogni caso di test è preceduto da una riga vuota.

In ciascun caso di test, la prima riga contiene l'intero N. La seconda riga contiene gli N valori $R_1, ..., R_N$, separati da spazio.

Dati di output

Il file di output deve contenere la risposta ai casi di test che sei riuscito a risolvere. Per ogni caso di test che hai risolto, il file di output deve contenere una riga con la dicitura

```
Case #t: p
```

dove `t` è il numero del caso di test (a partire da 1) e `p` è è il minimo numero di volte in cui la macchina deve essere azionata.

Esempi di input/output

Input: Download

```
2

4
1 2 3 1

4
100 0 1 1
```

Output: Download

```
Case #1: 3
Case #2: 101
```

Spiegazione

Nel **primo caso d'esempio**, è possibile azionare la macchina ad esempio nel seguente modo:

1. segmento da $a = 2$ a $b = 3$
2. segmento da $a = 1$ a $b = 4$
3. segmento da $a = 3$ a $b = 3$

Graficamente:

```
. x x .  <-- azionamento 1
x x x x  <-- azionamento 2
. . x .  <-- azionamento 3
-------
1 2 3 1  <-- totale radiazione accumulata
```

Non ci sono soluzioni con solo 2 azionamenti o meno, quindi la risposta corretta è 3.

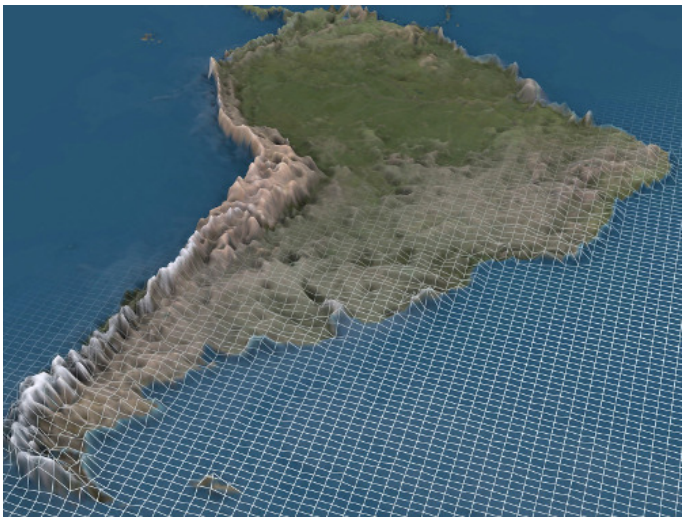
Nel **secondo caso d'esempio**, è possibile azionare la macchina ad esempio nel seguente modo:

1. segmento da $a = 1$ a $b = 1$ (ripeti 100 volte)
2. segmento da $a = 3$ a $b = 4$

Non ci sono soluzioni con solo 100 azionamenti o meno, quindi la risposta corretta è 101.

Escursione

Mojito vuole pianificare un'escursione sulle colline di Volterra. Ha a disposizione una mappa rettangolare, in cui è indicata l'altitudine della zona. Mojito vuole fare un percorso che parte dall'angolo in alto a sinistra della mappa e raggiunge l'angolo in basso a destra, in modo tale che il dislivello massimo che è costretto a fare ad ogni spostamento sia il mimimo possibile. Aiuta Mojito a calcolare questo dislivello!



Dettagli

La mappa è una tabella di numeri interi: ciascuno esprime l'altitudine in metri nel corrispondente punto della mappa. La tabella è composta di H righe e W colonne, numerate rispettivamente da 1 a H e da 1 a W. Nella *cella* di coordinate (i, j), ovvero in corrispondenza della riga i e della colonna j, è indicato il valore dell'altitudine $A_{i,j}$.

Mojito inizia l'escursione dalla cella di coordinate $(1, 1)$, in alto a sinistra, ed arriva alla cella di coordinate (H, W), in basso a destra. Ogni minuto si sposta di esattamente una cella, in una della quattro possibili direzioni (in alto, in basso, a destra o a sinistra). Non può però uscire dalla mappa.

Stabilito un percorso lungo la mappa, il **pericolo** associato a quel percorso è il *massimo dislivello tra due celle consecutive lungo il percorso*, ovvero la differenza di altitudine fra due celle consecutive: non cambia nulla se lo spostamento è in salita o in discesa.

Calcola il **pericolo** minimo, fra tutti i percorsi possibili che partono dalla cella $(1, 1)$ e arrivano alla cella (H, W).

Assunzioni

- $T = 27$, ci sono 27 casi di prova.
- $1 \leq H, W \leq 100$, la mappa ha dimensione massima 100×100.
- $(1, 1) \neq (H, W)$, ovvero la mappa è abbastanza grande da avere partenza e arrivo in punti diversi.
- $1 \leq A_{i,j} \leq 1.000.000$, l'altitudine in ogni cella è compresa fra 1 e $1.000.000$.

Dati di input

La prima riga del file di input contiene un intero T, il numero di casi di test. Seguono T casi di test, numerati da 1 a T. Ogni caso di test è preceduto da una riga vuota.

In ciascun caso di test, la prima riga contiene due interi H e W separati da uno spazio che corrispondono all'altezza, H, e alla larghezza, W, della mappa. Le successive H righe contengono ciascuna W interi separati da spazi, corrispondenti all'altitudine in metri lungo una riga della mappa. Ovvero, in ciascun caso di test, l'altitudine $A_{i,j}$ alle coordinate i e j appare sulla riga $(i + 1)$-esima, in posizione j.

Dati di output

Il file di output deve contenere la risposta ai casi di test che sei riuscito a risolvere. Per ogni caso di test che hai risolto, il file di output deve contenere una riga con la dicitura

```
Case #t: p
```

dove `t` è il numero del caso di test (a partire da 1) e `p` è il minimo valore di pericolo trovato per quel test case.

Esempi di input/output

Input: Download

```
3

2 2
100 150
110 130

4 4
1 5 6 7
2 4 3 8
2 9 2 8
3 3 2 9

1 10
2 4 6 8 10 12 14 16 18 20
```

Output: Download

```
Case #1: 20
Case #2: 1
Case #3: 2
```

Spiegazione

Nel **primo caso d'esempio**, Mojito sceglie il percorso:

```
100   150
 ▼
110 ▶ 130
```

ovvero, con i seguenti spostamenti:

- *in basso*, da $(1, 1)$ a $(2, 1)$, con un dislivello pari a $110 - 100 = 10$
- *a destra*, da $(2, 1)$ a $(2, 2) = (H, W)$, con un dislivello pari a $130 - 110 = 20$.

Il pericolo del percorso è 20 (il massimo fra i dislivelli, 10 e 20).

Non ci sono percorsi migliori, quindi la risposta corretta è 20. L'altro percorso possibile è:

```
100 ▶ 150
        ▼
110   130
```

che ha dislivelli 50 e 20, e quindi ha pericolo 50.

Nel **secondo caso d'esempio**, Mojito sceglie il percorso:

```
1   5 ▶ 6 ▶ 7
▼   ▲       ▼
2   4 ◀ 3   8
▼       ▲   ▼
2   9   2   8
▼       ▲   ▼
3 ▶ 3 ▶ 2   9
```

Gli spostamenti hanno tutti dislivello 0 o 1, quindi il pericolo del percorso è 1. Non ci sono percorsi di pericolo pari a 0, quindi la risposta corretta è 1.

Nel **terzo caso d'esempio** c'è un solo percorso possibile.