# Mozzarelle di Bufala (`bufale` [ 65 punti ])

*Questo problema è preso ed adattato dalla Finale Nazionale 2013 (Salerno) delle Oii (Olimpiadi Italiane di Informatica).*

Salerno è la patria delle mozzarelle di bufala. Come benvenuto, a Monica e Paola sono stati regalati $N$ assaggi di questo rinomato prodotto caseario, tutti diversi tra loro. Siccome $N$ è un numero pari Monica e Paola decidono di mangiarsi subito $\frac{N}{2}$ mozzarelle a testa, finchè sono ancora fresche. Avendo preferenze diverse su ciascuno dei campioni, decidono di stilare innanzitutto una tabella con le loro valutazioni:

| Numero mozzarella | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Voto di Monica | 10 | 2 | 4 | 6 | 1 | 7 | 3 | 4 |
| Voto di Paola | 6 | 6 | 1 | 0 | 3 | 8 | 5 | 7 |

Come ripartirsi ora le mozzarelle?

Una ripartizione è *equa* se Monica e Paola ricevono lo stesso numero di mozzarelle. La soddisfazione complessiva comportata da una ripartizione è la somma delle soddisfazioni totali di Monica e di Paola, dove la soddisfazione totale di ciascuna di loro è la somma dei voti da lei stessa attribuiti alle mozzarelle che degusta.

**Goals:** con quanta efficienza sai rispondere alle seguenti domande?

[**LB**]: quale è il minimo valore di soddisfazione complessiva che si sarebbe potuto ottenere se gli ospiti avessero deciso loro come ripartire le $N$ mozzarelle tra Monica e Paola, eventualmente anche in numero diverso?

[**worst_val**]: quale è il minimo valore di soddisfazione complessiva ottenibile dando metà delle mozzarelle a Monica e l'altra metà a Paola?

[**opt_val**]: quale è il massimo valore di soddisfazione complessiva di una ripartizione equa?

[**opt_sol**]: sai proporre una ripartizione ottima, ossia di massima soddisfazione totale?

[**count_opts**]: quante sono le diverse ripartizioni ottime?

## Input

Si legga l'input da `stdin`. La prima riga contiene $T$, il numero di testcase (istanze) da risolvere. Seguono $T$ istanze del problema, ciascuna descritta nel seguente formato: la prima riga contenente il numero $N$ di mozzarelle seguono $N$ righe di due numeri ciascuna: l'$i$-esima di queste righe contiene, separati da spazio e in questo ordine, il voto di Monica e quello di Paola per la mozzarella $i$-esima.

## Output

Per ciascuna istanza, prima di leggere l'istanza successiva, scrivi su `stdout` il tuo output così strutturato:

[**goals 1,2,3 (LB, worst_val, opt_val)**]: la prima riga contiene i tre numeri `LB`, `worst_val` e `opt_val`.

[**goal 4 (opt_sol)**]: la seconda riga contiene $N$ cifre separate da spazio. Se l'$i$-esima di queste cifre è un 1 ciò significa che l'$i$-esima mozzarella va a Monica, se è un 2 l'$i$-esima mozzarella va a Paola.

[**goal 5 (num_sols)**]: il numero di soluzioni ottime potrebbe essere molto grande. Calcola il resto della sua divisione per $1,000,000,007$.

## Assunzioni

1. $1 \leq N \leq 500,000$
2. i voti sono tutti compresi nell'intervallo $[0, 1000]$

## Esempio di Input/Output

```
<start in>
2
8
10 6
2 6
4 1
6 0
1 3
7 8
3 5
4 7
<more>
```

```
8
5 0
5 1
5 8
5 0
5 0
5 1
5 1
5 9

 <end>
```

```
<start out>
24 25 48
1 2 1 1 2 1 2 2
1
13 21 39
1 1 2 1 1 2 2 2
3

 <end>
```

**Spiegazione:** Nel primo caso di esempio la cosa peggiore sarebbe che a Monica andassero le mozzarelle 2,5,6,7,8 e a Paola andassero le mozzarelle 1,3 e 4 (soddisfazione1 = 2+1+7+3+4=17, soddisfazione2 = 6+1+0=7, soddisfazione complessiva = 17+7=24). La peggiore ripartizione equa darebbe le mozzarelle 2,5,7,8 a Monica e le mozzarelle 1,3,4,6 a Paola (soddisfazione complessiva = 10+15=25). La migliore ripartizione equa assegna a Monica le mozzarelle 1,3,4,6 e a Paola le mozzarelle 2,5,7,8 (soddisfazione complessiva= 27+21=48). Questa è l'unica ripartizione equa delle mozzarelle che totalizza 48 in soddisfazione complessiva. Nel secondo caso di esempio la ripartizione equa ottima non è unica.

## Subtask

Sommiamo i punti raccolti sulle varie istanze del problema affrontate (testcase).

Il punteggio ottenuto su un'istanza sarà la somma dei punti dei goal correttamente evasi, purchè si rispetti almeno il formato in tutte le righe di output, incluse quelle che competono agli altri goal (altrimenti salta il protocollo di comunicazione tra il tuo programma risolutore e il server).

I testcase sono raggruppati nei seguenti subtask. Oltre alle dimensioni o altre caratteristiche peculiari delle istanze, ogni subtask precisa quanti punti competono ai vari goal.

1. **[ 0 pts← 2 istanze da 0 + 0 + 0 + 0 + 0 punti] `esempi_testo`:** i tre esempi del testo
2. **[12 pts← 4 istanze da 0 + 0 + 1 + 1 + 1 punti] `small`:** $n \leq 10$
3. **[12 pts← 4 istanze da 0 + 0 + 1 + 1 + 1 punti] `indifferent`:** Monica assegna lo stesso voto a tutte le mozzarelle, $n \leq 100,000$
4. **[20 pts← 4 istanze da 1 + 1 + 1 + 1 + 1 punti] `medium`:** $n \leq 300$
5. **[21 pts← 7 istanze da 0 + 0 + 1 + 1 + 1 punti] `big`:** $n \leq 100,000$

In generale, quando si richiede la valutazione di un subtask vengono valutati anche i subtask che li precedono, ma si evita di avventurarsi in subtask successivi fuori dalla portata del tuo programma che potrebbe andare in crash o comportare tempi lunghi per ottenere la valutazione completa della sottomissione. Ad esempio, chiamando[1, 2] :

```
rtal -s <URL> connect -a size=indifferent bufale -- <MY_SOLUTION>
```

vengono valutati, nell'ordine, i subtask:

[1] <URL> server esame: `wss://ta.di.univr.it/esame`

[2] <URL> server esercitazioni e simula-prove: `wss://ta.di.univr.it/algo2025`

`esempi_testo`, `small`, `indifferent`.

Il valore di default per l'argomento `size` è `big` che include tutti i testcase.

Nel template di riga di comando quì sopra `<MY_SOLUTION>` è una qualsiasi scrittura che ove immessa anche da sola al prompt della CLI comporti l'avvio del solver da tè realizzato. Solo alcuni esempi:

- `./a.out` per un compilato da `C/c++`, eventualmente seguito dagli argomenti che prevede
- `./my_solution.py arg1 arg2 ...` se il tuo file `my_solution.py` col codice python ha i permessi di esecuzione e inizia con la riga di shebang
- `python my_solution.py` o `python3 my_solution.py` per far eseguire il tuo script da un interprete python

Se vuoi che una tua sottomissione venga conteggiata ai fini di un esame o homework devi allegare il sorgente della tua soluzione con `-fsource=<FILENAME>` (ad esempio `-fsource=my_solution.py` subito a valle del comando `connect`. Inoltre devi aver precedentemente affettuato il login tramite credenziali GIA (lancia prima `rtal -s <URL> login` e segui le istruzioni per impostare il sign-on).

Il comando `rtal` prevede diversi parametri, consigliamo di esplorarne e sperimentarne le potenzialità ed opzioni d'uso. Un tutorial all'uso di `rtal` è esposto alla pagina:

`https://github.com/romeorizzi/AlgoritmiUniVR/tree/main/strumenti`