La stagione dei monsoni modifica il panorama (`pioggia` [52 punti])

Un vettore $H = h_0, ..., h_{n-1}$ di interi positivi riporta le altezze di una fila di n torri consecutive. Arriva la stagione dei monsoni e piove a dirotto su tutto questo panorama. L'acqua dove può scappa via sia da destra che da sinistra, e comunque si livella in pozzanghere come in figura. Sai stabilire quale sia il profilo delle altezze acqua inclusa al termine della stagione dei monsoni?

Con quanta efficienza sai computare le seguenti informazioni?

[**Q**]: la quantità totale di acqua rimasta intrappolata espressa in numero di quadretti riempitisi d'acqua e facenti parte di pozzanghere.

[**A**]: il vettore $A = a_0, ..., a_{n-1}$ che descrive il nuovo panorama riportando le altezze acqua inclusa al termine della stagione delle grandi piogge.

Input

Si legga l'input da `stdin`. La prima riga contiene T, il numero di testcase (istanze) da risolvere. Seguono T istanze del problema, dove ogni istanza è descritta in due righe: la prima contiene il numero $n \in \mathbb{N}$, la seconda riporta gli n numeri naturali $h_0, ..., h_{n-1}$.

Output

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

[**goal 1, Q**]: la prima riga contiene il numero $Q \in \mathbb{N}$ che esprime in quadretti la quantità d'acqua rimasta intrappolata in pozzanghere..

[**goal 2, A**]: la seconda riga contiene, nell'ordine e separate da spazi, le n nuove altezze $a_0, ..., a_{n-1}$.

Oltre alle dimensioni delle istanze, ogni subtask precisa quanti punti competono a ciascuno dei due goal. Per ogni istanza di quel subtask si otterranno tutti i 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).

Esempio di Input/Output

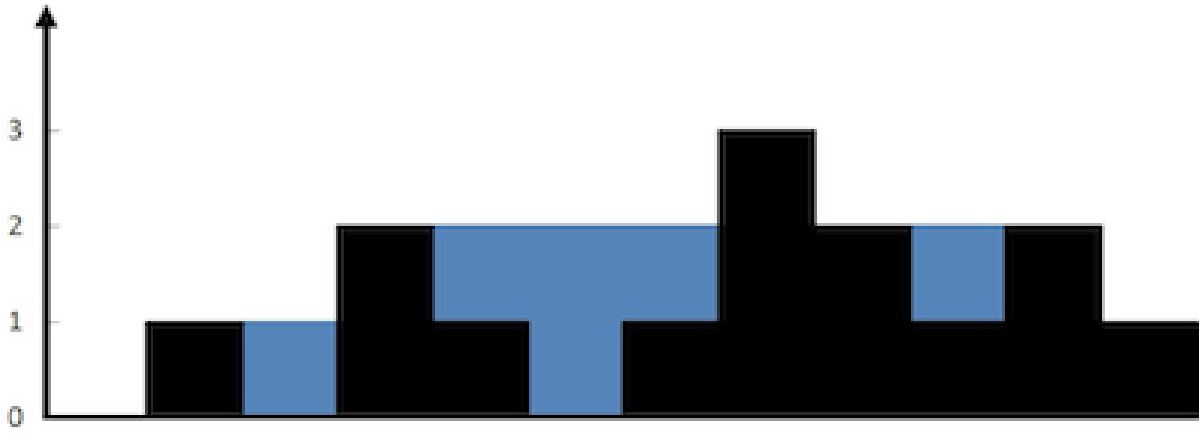
Input da `stdin`

```
2
8
3 1 2 7 1 2 1 5
12
0 1 0 2 1 0 1 3 2 1 2 1
```

Output su `stdout`

```
14
3 3 3 7 5 5 5 5
6
0 1 1 2 2 2 2 3 2 2 2 1
```

Nota: la seconda istanza è l'esempio in figura all'inizio del testo. Per comodità la riportiamo di nuovo:



Subtask

Il tempo limite per istanza (ossia per ciascun testcase) è sempre di 1 secondo.

I testcase sono raggruppati nei seguenti subtask.

1. **[4 pts← 2 istanze da 1 + 1 punti] `esempi_testo`:** i due esempi del testo
2. **[6 pts← 3 istanze da 1 + 1 punti] `small_narrow`:** $n \leq 20, \min(h) = 1, \max(h) = 2$
3. **[6 pts← 3 istanze da 1 + 1 punti] `medium_narrow`:** $n \leq 500, \min(h) = 3, \max(h) = 4$
4. **[6 pts← 3 istanze da 1 + 1 punti] `big_narrow`:** $n \leq 20, \min(h) = 1003, \max(h) = 1004$
5. **[10 pts← 5 istanze da 1 + 1 punti] `small`:** $n \leq 20, \min(h) \geq 1, \max(h) \leq 10,000$
6. **[10 pts← 5 istanze da 1 + 1 punti] `medium`:** $n \leq 500, \min(h) \geq 1, \max(h) \leq 10,000$
7. **[10 pts← 5 istanze da 1 + 1 punti] `big`:** $n \leq 200,000, \min(h) \geq 1, \max(h) \leq 10,000$

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 -x <token> -a size=medium_narrow
           pioggia  -- python my_solution.py
```

vengono valutati, nell'ordine, i subtask:

`esempi_testo`, `small_narrow`, `medium_narrow`.

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

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

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