Uso della fast-matrix exponentiation nel computo di ricorrenze (`prima_PD_su_linea_fast_exp` [95 punti])

In questo esercizio vogliamo spingere sul computo di $f(n)$, dove $f(n)$ è il numero di sottoinsiemi S di $\{0, 1, ..., n-1\}$ che non contengano due elementi i, j con $0 \leq i < j < n$ e $j < i + 3$. Avevamo lavorato sulla struttura ricorsiva di tale famiglia di sottoinsiemi dell'insieme $\{0, 1, ..., n-1\}$ degli indice di un vettore di interi A nel problema `prima_PD_su_linea`, dove questi sottoinsiemi corrispondevano alle soluzioni ammissibili e si chiedeva di trovare una soluzione ammissibile che massimizzasse il funzionale $\sum_{i \in S} A[i]$. Nello stesso esercizio chiedevamo anche di computare $f(n)$, per evidenziale il legame tra la struttura dello spazio delle soluzioni ammissibili e la struttura degli algoritmi per il problema di massimizzazione.

In questo esercizio torniamo al problema più elementare del computo di $f(n)$, chiedendo però di affrontarlo per valori di n significativamente maggiori. Per farlo, dopo aver isolato una ricorrenza opportuna per $f(n)$, occorre esprimerla come una ricorrenza matriciale in modo da avvalersi della tecnica della fast-matrix multiplication. Se una ricorrenza per $f(n)$ può essere ottenuta che faccia riferimento solo a valori $f(n')$ con $n' \in [n-k, n)$ con k costante, allora nel progredire nel calcolo di $f(n)$ per valori di n via via più grandi possiamo pensare che tutti gli n' che ci interessano siano contenuti in una sliding window $[n-k, n-k+1, ..., n-1]$ di larghezza costante e possiamo guardare al vettore di valori $[f(n-k), f(n-k+1), ..., f(n-1)]$ come al nostro "stato" da aggiornare man mano che la finestra scorre verso destra. La ricorrenza ci consente di condurre tale aggiornamento. Se la ricorrenza è a coefficienti costanti possiamo quindi utilizzarla per comporre una matrice di transizione di stato. Ed a questo punto è possibile applicare la tecnica della fast-matrix multiplication.

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 consta di un singolo valore per n. Ogni istanza è descritta su una nuova riga, successiva a quella dell'istanza precedente. La riga contiene il solo numero n.

Output

Per ciascuna istanza n, prima di leggere da `stdin` l'istanza successiva, scrivi su `stdout`, in una nuova riga, il valore $f(n) \% 1,000,000,007$.

Esempio di Input/Output

Input da `stdin`

```
3
4
6
120
```

Output su `stdout`

```
6
13
228104745
```

Spiegazione: Su `stdout`, nei primi due casi ($n = 4$ e $n = 6$) è riportato proprio il valore $f(n)$ (ossia $f(4) = 6$ e $f(6) = 13$). Nel terzo caso, poichè $f(120) > 1,000,000,007$, su `stdout` viene stampato $f(120)\%1,000,000,007$, ossia il resto della divisione del dividendo $f(120)$ sul divisore $1,000,000,007$.

Subtask

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

I testcase sono raggruppati nei seguenti subtask.

1. **[0 pts← 3 istanze da 0 punti] `esempi_testo`:** i tre casi di esempio del testo
2. **[5 pts← 5 istanze da 1 punti] `practice`:** piccoli casi che puoi risolvere a mano per impratichirti col problema comunque facendo punti (non puoi richiedere delle risposte corrette dal server)
3. **[10 pts← 5 istanze da 2 punti] `tiny`:** $n \leq 20$
4. **[10 pts← 5 istanze da 2 punti] `small`:** $n \leq 200$
5. **[6 pts← 3 istanze da 2 punti] `medium`:** $n \leq 2000$
6. **[6 pts← 3 istanze da 2 punti] `big`:** $n \leq 10000$
7. **[6 pts← 2 istanze da 3 punti] `large`:** $n \leq 100000$
8. **[16 pts← 2 istanze da 8 punti] `extra_large`:** $n \leq 10000000$
9. **[16 pts← 2 istanze da 8 punti] `huge`:** $n \leq 10^{10}$
10. **[20 pts← 2 istanze da 10 punti] `colossal`:** $n \leq 10^{100}$

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
prima_PD_su_linea_fast_exp  -- python my_solution.py
```

vengono valutati, nell'ordine, i subtask:

`esempi_testo`, `practice`, `tiny`, `small`, `medium`.

Il valore di default per l'argomento `size` è `colossal` 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`