14.11 Gestione di una coda mediante array

In prima approssimazione la gestione di una struttura astratta di tipo coda per mezzo di un vettore, con le relative operazioni di inserimento, eliminazione e visualizzazione degli elementi, ricorda la gestione di una pila; anche in questo caso dobbiamo dare una stima del massimo numero di elementi che la coda può contenere.

Una prima soluzione del problema si può in effetti ottenere con semplici modifiche a partire dal programma del Listato 14.4 con cui abbiamo implementato una pila. Di questo restano infatti invariate sia la struttura sia le variabili, i tipi di dato e tutte le funzioni che vi compaiono, con la sola eccezione delle funzioni di visualizzazione e di eliminazione. Per quanto non indispensabile, è inoltre consigliabile cambiare i nomi dei vari identificatori, sostituendo tutte le occorrenze della

parola *pila* con *coda*. In questo modo garantiremo al programma l'auspicabile, immediata leggibilità del ruolo giocato da variabili, costanti e funzioni.

Al lettore il compito di realizzare tale gestione della coda, noi passiamo invece a un'osservazione. Poiché l'inserimento e l'eliminazione (Figura 14.12) vengono effettuate agli estremi opposti della sequenza, la zona occupata dall'array tende a muoversi all'interno della struttura, allontanandosi dalla base.

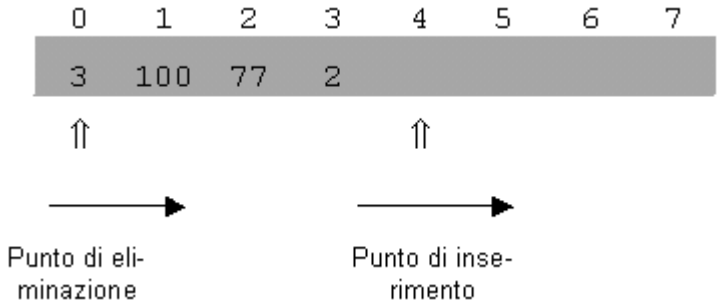

Figura 14.12 Successivi inserimenti ed eliminazioni nella coda tendono a muovere la zona occupata verso l'alto

Diventa quindi molto difficile stabilire la grandezza dell'array ed è probabile avere uno spreco di memoria. Per esemplificare quanto affermato, chiamiamo `puntCoda` il puntatore all'ultimo elemento inserito e `puntTesta` il puntatore all'elemento che deve essere estratto per primo. Dopo otto inserzioni successive avremo la coda effettivamente piena (Figura 14.13); se avvengono due eliminazioni la nuova situazione sarà quella di Figura 14.14.

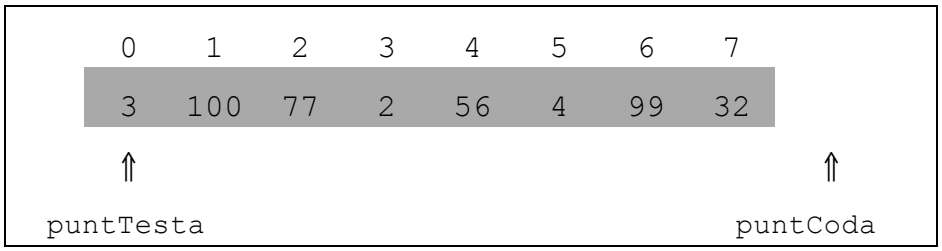

Figura 14.13 Caso di coda piena

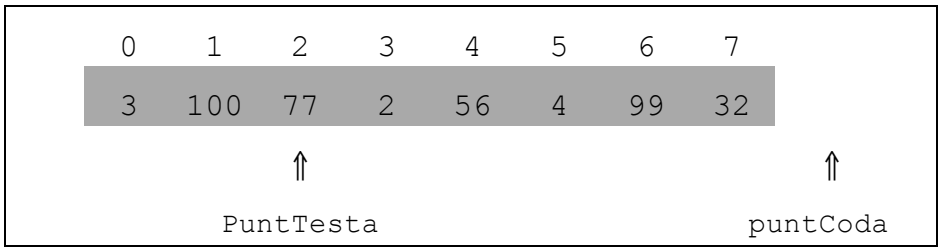

Figura 14.14 La coda ha due posizioni libere

Esistono due elementi liberi nell'array ma non sembra possibile effettuare nessun inserimento. Il fatto che in figura appaiano ancora 3 e 100 non deve indurre in errore: la sequenza significativa va dal puntatore alla testa al puntatore alla coda. Una soluzione del problema è quella di gestire le operazioni che si effettuano sull'array in modo circolare (Figura 14.15).

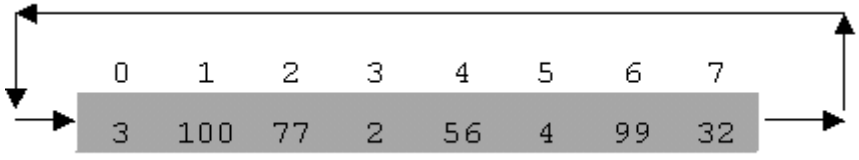

Figura 14.15 Array gestito in modo circolare

Se ritorniamo alla situazione vista in precedenza (Figura 14.14), l'inserimento di un nuovo elemento (`63`) verrebbe effettuato nella prima posizione dell'array. Per determinare l'indice dell'elemento dell'array dove effettuare l'inserimento si usa

```
puntCoda = (puntCoda % n)+1;
```

Nel caso dell'esempio `n=8` e il vecchio valore di `puntCoda` è uguale a `8`, per cui il nuovo valore di `puntCoda` è `0` (Figura 14.16).

Figura 14.16 Inserimento dell'informazione (`63`) in testa alla coda

La condizione di coda piena equivale a `puntTesta` uguale a `puntCoda` più uno, considerando l'array in modo circolare, dove la posizione *n*-esima precede la prima posizione dell'array:

if(puntTesta==((puntCoda % n)+1) coda piena...