

Laboratorio di Algoritmi e Strutture Dati

Esercitazioni del 27 Novembre 2012

Esercizio 3: otto regine

Si tratta di un esercizio sulla ricorsione e sulle tecniche di backtracking.

Si deve implementare un algoritmo che risolva il problema delle otto regine.

Bisogna collocare otto regine su una scacchiera in modo tale che nessuna di esse possa mangiarne un'altra con una mossa.

Ricordiamo che negli scacchi una regina può mangiare un altro pezzo con una mossa se esso si trova sulla stessa riga, sulla stessa colonna o su una delle diagonali che contengono la regina¹.

La scacchiera è modellata da un'array bidimensionale globale `matrix`:

```
#define N 8
```

```
static short matrix[N][N];
```

Si richiede in primo luogo di implementare una funzione

```
void stampa(void)
```

che stampi sullo schermo lo stato attuale della scacchiera (sia che contenga una soluzione, sia che la si stia ancora cercando). Esempio:

```
[ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ]
```

scacchiera vuota

```
[ ][ ][0][ ][ ][ ][ ][ ][ ]
[0][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][0][ ]
[ ][ ][ ][ ][0][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][0]
[ ][0][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][0][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][0][ ][ ][ ]
```

una soluzione

Si noti che dato che le regine sono `N == 8`, in tutte le soluzioni ogni riga e ogni colonna sarà occupata da una e una sola regina.

Per la ricerca della soluzione, si implementi un algoritmo di backtracking (cfr. con il *giro del cavallo*, sulle dispense del corso), seguendo queste osservazioni.

1. Si proceda riga per riga (o colonna per colonna).

¹e non devono essere presenti altri pezzi sulla riga/colonna/diagonale a sbarrare il passo alla regina, ma questo aspetto è irrilevante per il nostro problema

2. Per ogni $i = 0, \dots, 6$ si assuma di aver piazzato nelle prime i righe (colonne), i regine che non si disturbino fra loro.
3. Provare tutte le posizioni possibili per il collocamento di una regina nella $(i + 1)$ -esima riga, escludendo le posizioni immediatamente vietate: quelle cioè che vedono già un'altra regina nella stessa colonna (riga) o in una delle diagonali a cui la posizione considerata appartiene.
(Può essere utile implementare funzioni ausiliarie per controllare che le colonne (righe) e diagonali interessate siano libere da regine.)
4. Se la collocazione riga per riga (colonna per colonna) giunge a piazzare una regina nell'ultima riga ($i == N-1 = 7$), allora abbiamo trovato una soluzione.
5. Se invece al passo i -esimo non si è trovata alcuna posizione per la $(i + 1)$ -esima regina che conduca a una soluzione, allora si deve annullare (fare *backtrack*) anche la posizione scelta correntemente per la i -esima regina, e provarne una alternativa. Se non vi sono più alternative neanche per la i -esima regina allora si dovrà annullare la $(i - 1)$ -esima scelta e così via.

Si consiglia di implementare una funzione

```
char backtrack(short x, short y)
```

che restituisca 1 se si è trovata una soluzione a partire dallo stato attuale della scacchiera, collocando la y -esima regina in posizione (x,y) . `backtrack` restituisca 0 altrimenti.

Ovviamente questa funzione restituirà immediatamente 1 se $y == 7$.

Altrimenti proverà tutti i modi immediati di collocare una regina nella $y+1$ -esima riga, e per ognuna di queste innescherà ricorsivamente un'opportuna chiamata a se stessa.

Si noti che variando il valore della macro `N` si ottengono programmi che risolvono gli analoghi problemi di collocazione di N regine su una scacchiera $N \times N$.

A questo riguardo, al posto di usare una macro per definire il lato `N`, affrontiamo il seguente esercizio.

Il problema delle k regine

Si deve implementare un algoritmo che risolva il problema delle k regine, dove k è un intero positivo, immesso dall'utente.

Bisogna collocare k regine su una scacchiera di $k * k$ caselle, in modo tale che nessuna di esse possa mangiarne un'altra con una mossa.

Per implementare la scacchiera si ricorra a un array di `short` *allocato dinamicamente*.

Facoltativo

Modificare il programma in modo che generi tutte le soluzioni possibili del problema delle otto regine. (N.B.: ce ne sono 92 distinte.)