Architetture dei sistemi di elaborazione
Esame del 31.1.2019 - parte II

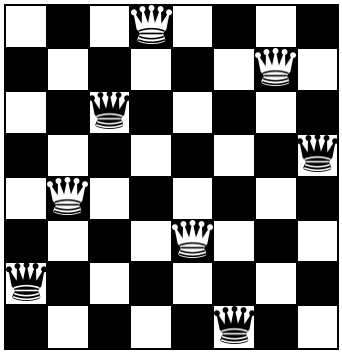
B2

Durata: 90 minuti.

E' possibile consultare:
- qualunque materiale cartaceo
- i lucidi scaricati dalla pagina del corso sul portale della didattica
- il codice dei laboratori eventualmente caricato fra gli elaborati sul portale della didattica.

Gli studenti sorpresi a comunicare fra loro saranno immediatamente allontanati dal laboratorio.

Nel gioco degli scacchi, la regina può muoversi in orizzontale, verticale o diagonale di un numero qualunque di caselle. Il rompicapo delle 8 regine consiste nel posizionare 8 regine in una scacchiera 8x8, in modo che nessuna regina possa raggiungerne un'altra effettuando un solo movimento. In altre parole, su ciascuna riga, colonna e diagonale della scacchiera ci deve essere una sola regina.
La scacchiera è memorizzata tramite una matrice `scacchiera` 8x8, in cui ogni elemento occupa 1 byte e assume il valore 1 se sulla casella corrispondente è presente una regina, 0 altrimenti.



a) Scacchiera

0	0	0	1	0	0	0	0
0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0
0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0

b) matrice `scacchiera`

Indice di riga \ Indice di colonna	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*
0	0	1	2	3	4	5	6	7
1	8	9	10	11	12	13	14	15
2	16	17	18	19	20	21	22	23
3	24	25	26	27	28	29	30	31
4	32	33	34	35	36	37	38	39
5	40	41	42	43	44	45	46	47
6	48	49	50	51	52	53	54	55
7	56	57	58	59	60	61	62	63

c) indici di riga, colonna e cella della matrice `scacchiera`

Si vuole scrivere un programma per determinare se una regina può raggiungerne un'altra compiendo un solo movimento.
Si richiede di aprire con Keil il progetto **progetto.uvproj** presente all'interno della cartella **template** e scrivere subroutine **debuggate e funzionanti** che rispondano alle 3 specifiche seguenti.
Nota 1: non si deve modificare il codice che chiama le subroutine. E' richiesto solamente di implementare le subroutine. La matrice `scacchiera` è già dichiarata.
Nota 2: le specifiche devono essere svolte in ordine. Si può passare alla specifica 2 solamente dopo aver verificato che la soluzione alla specifica 1 funzioni correttamente. Lo stesso per la specifica 3.

Specifica 1 (8 punti). Scrivere una subroutine `checkRow` che controlli se una data regina R (indicata fra i parametri in input) possa raggiungere una qualunque altra regina muovendosi orizzontalmente. La subroutine `checkRow` riceve attraverso lo stack i seguenti parametri, nell'ordine indicato:

- indirizzo della matrice `scacchiera`
- indice della riga `row` in cui è presente la regina R
- indice della colonna `column` in cui è presente la regina R
- spazio per il valore di ritorno.

La subroutine `checkRow` restituisce attraverso lo stack il valore 1 se la regina R può raggiungere un'altra regina muovendosi orizzontalmente (a destra o a sinistra), il valore 0 altrimenti.

Esempio: `row` = 6, `column` = 0.
Il valore restituito è 0 perché nella riga 6 è presente solamente la regina R.

Specifica 2 (4 punti). Scrivere una subroutine `checkColumn` che controlli se la regina R indicata come parametro in input possa raggiungerne un'altra muovendosi verticalmente. La subroutine `checkColumn` riceve attraverso lo stack i seguenti parametri, nell'ordine indicato:

- indirizzo della matrice `scacchiera`
- indice della riga `row` in cui è presente la regina R
- indice della colonna `column` in cui è presente la regina R
- spazio per il valore di ritorno.

La subroutine `checkColumn` restituisce attraverso lo stack il valore 1 se la regina R può raggiungere un'altra regina muovendosi verticalmente (in alto o in basso), il valore 0 altrimenti.

Esempio: `row` = 6, `column` = 0.
Il valore restituito è 0 perché nella colonna 0 è presente solamente la regina R.

Specifica 3 (6 punti). Scrivere una subroutine `checkDiagonal` che controlli se la regina R indicata come parametro in input possa raggiungerne un'altra muovendosi in diagonale. E' richiesto di controllare solo la diagonale che inizia in basso a sinistra e termina in alto a destra. La subroutine `checkDiagonal` riceve attraverso lo stack i seguenti parametri, nell'ordine indicato:

- indirizzo della matrice `scacchiera`
- indice della riga `row` in cui è presente la regina R
- indice della colonna `column` in cui è presente la regina R
- spazio per il valore di ritorno.

La subroutine `checkDiagonal` restituisce attraverso lo stack il valore 1 se la regina R può raggiungere un'altra regina muovendosi in diagonale (in avanti o indietro), il valore 0 altrimenti.

Esempio: `row` = 6, `column` = 0.
Il valore restituito è 0 perché nella diagonale che comprende le celle 48 – 41 – 34 – 27 – 20 – 13 - 6 è presente solamente la regina R.

Suggerimento: dati gli indici `row` e `column`, l'indice della prima cella della diagonale è:
MIN((`row` + `column`) * 8, `row` + `column` + 49)
Nell'esempio, la prima cella della diagonale ha indice MIN((6 + 0)*8, 6 + 0 + 49) = 48.

Il numero di celle presenti nella diagonale è pari a:
MIN(`row` + `column` + 1, 15 - `row` - `column`)
Nell'esempio, il numero di celle nella diagonale è MIN(6 + 0 + 1, 15 – 6 – 0) = MIN(7, 9) = 7.