Laboratorio di algoritmi e strutture dati*

Docente: Violetta Lonati

Cristalli

Consideriamo un cristallo che si evolve nel tempo nel modo seguente. All'istante $t = 0$ il cristallo è costituito da un solo elemento quadrato di lato unitario. All'istante $t + 1$, il cristallo ha al centro un elemento quadrato di lato unitario e, a ciascuno dei quattro vertici del quadrato è adiacente un cristallo ottenuto al tempo t.

Ad esempio, indicando con '`*`' un elemento quadrato di lato unitario del cristallo e con con '`.`' un quadrato unitario vuoto, i cristalli ottenuti ai tempi $t = 0, 1, 2$ sono:

t = 0

```
*
```

t = 1

```
*.*
.*.
*.*
```

t = 2

```
*.*.*.*
.*...*.
*.*.*.*
...*...
*.*.*.*
.*...*.
*.*.*.*
```

t = 3

```
*.*.*.*.*.*.*.*
.*...*...*...*.
*.*.*.*.*.*.*.*
...*.......*...
*.*.*.*.*.*.*.*
.*...*...*...*.
*.*.*.*.*.*.*.*
.......*.......
*.*.*.*.*.*.*.*
.*...*...*...*.
*.*.*.*.*.*.*.*
...*.......*...
*.*.*.*.*.*.*.*
.*...*...*...*.
*.*.*.*.*.*.*.*
```

L'obiettivo è quello di costruire una matrice di `char` che rappresenta il cristallo ottenuto all'istante t. Ogni cella della matrice rappresenta un quadrato unitario dello spazio che può essere pieno o vuoto, come nell'esempio precedente.

1. Scrivete una funzione ricorsiva `int latoCristallo( int t )` dove $t \geq 0$, che calcola la misura del lato l del cristallo al tempo t. Ad esempio, per $t = 0$ il valore di l è 1; per $t = 1$ l vale 3.
2. Scrivete una funzione `char **creaMatrice( int n )` che crea dinamicamente una matrice quadrata di `char` di lato n (ossia, una matrice $n \times n$) in cui tutti gli elementi risultano vuoti.
3. Scrivete una funzione `void stampaMatrice( char **m, int n )` che stampa il contenuto della matrice quadrata $n \times n$ passata come primo parametro.
4. Il cristallo va costruito ricorsivamente. Non essendo possibile in C passare una sottomatrice di una matrice, occorre passare alla funzione l'*intera* matrice, specificando le righe e le colonne della sottomatrice da considerare.

   La funzione ricorsiva `void crist( char **m, int r0, int c0, int l )` costruisce il cristallo di lato l nella sottomatrice della matrice m composta dagli elementi $M[r][c]$ tali che:

$$r_0 \leq r < r_0 + l \qquad c_0 \leq c < c_0 + l$$

*Ultima modifica 14 novembre 2019

La funzione deve riempire la sottomatrice in esame in base alla definizione di cristallo. Assumete che gli indici r_0, c_0,l siano corretti (ossia, $m[r][c]$ è effettivamente un elemento della matrice m).

Suggerimento: Il caso base si ha quando $l = 1$ (cristallo al tempo 0). Nel passo induttivo, occorre riempire l'elemento centrale della sottomatrice e fare quattro chiamate ricorsive per riempire le quattro sottomatrici che descrivono i cristalli adiacenti ai vertici dell'elemento centrale (cristalli costruiti al tempo immediatamente precedente a quello attuale).

Un altro suggerimento: essendo i cristalli di lato l uguali, si può in realtà fare una sola chiamata ricorsiva per determinare un cristallo C_1 al tempo immediatamente precedente, che ha come lato $l/2$; per gli altri tre cristalli è sufficiente copiare C al posto giusto. A tal fine è utile definire una funzione `copiaSottoMatrice( Cell **m, int l, int r0, int c0, int r1, int c1 )` che copia in posizione $(r1, c1)$ la sottomatrice di m di dimensione l che si trova in posizione $(r0, c0)$.

5. Scrivete una funzione per fare la chiamata principale della funzione del punto precedente. La funzione deve avere intestazione `void cristallo( char **m, int l )` dove l è la misura del lato di un cristallo e m è una matrice quadrata $l \times l$. La funzione deve riempire la matrice in modo che rappresenti il cristallo di lato l.

Se l'esercizio è stato svolto correttamente, le seguenti linee di codice stampano il cristallo al tempo t, dove il valore di t è letto da standard input.

```
char **matrix;
int t, lato;
scanf( "%d", &t );                // legge il tempo
if( t >= 0 ){
  lato = latoCristallo( t );      // dimensione della matrice
  matrix = creaMatrice( lato );   // crea matrice per rappresentare il cristallo
  cristallo( matrix, lato );      // costruisce il cristallo avente lato assegnato
  stampaMatrice( matrix, lato );  // stampa la matrice
}
```

Se non si usa la ricorsione, la soluzione non è banale!!

Completare l'esercizio inserendo istruzioni per liberare lo spazio occupato dalla matrice.