Scritto di Programmazione del 10/7/2013

Note importanti: nella vostra home trovate 2 file, esercizio1.cpp ed esercizio2.cpp relativi all'esercizio iterativo e a quello ricorsivo, rispettivamente. Ciascuno di questi file contiene un main in cui si aprono i file di i/o (input1, output1 e input2, output2) e si eseguono le letture dei dati che i file di input devono contenere. Le letture spiegano i dati che i file di input devono contenere. I main eseguono anche le scritture finali sui file in output, quindi l'i/o è già fatto. Il file esercizio2.cpp contiene anche le funzioni ricorsive R0 ed R1, come spiegato nel seguito. **Dovete consegnare i 2 file, esercizio1.cpp ed esercizio2.cpp, ampliati con le funzioni richieste nel seguito**. Il comando di consegna (eseguito in una directory che contiene solo i file da consegnare) è: consegna esercitazione

Introduzione: affrontiamo di nuovo un problema di pattern matching (simile a quello del 28/6) in cui abbiamo un array T ad una dimensione di C*R interi che "va visto" come un array a 2 dimensioni con R righe e C colonne. Come nell'esame del 28/6, anche in questo esame vanno considerati **solamente match contigui ed eventualmente non completi**. Quindi un match deve sempre coinvolgere il primo elemento del pattern, poi il secondo e il terzo e così via e può interrompersi anche prima della fine del pattern..

In questo esercizio non abbiamo un solo pattern, ma R pattern, infatti vogliamo effettuare il match tra le righe di T e le colonne di T nel modo seguente:

per ogni riga si devono considerare i match (contigui ed eventualmente non completi) della riga su tutte le colonne di T in modo da determinare il match di lunghezza massima assoluta (massima per tutte le colonne) e per tale match si vuole produrre una tripla di interi che rappresenta, la colonna in cui il massimo match occorre, l'indice dell'elemento della colonna da cui il match inizia e la lunghezza del match.

Esempio: Sia R=8, C= 6 e T il seguente array. Si ricordi che T è ad una dimensione, ma lo rappresentiamo con R righe e C colonne per facilitare la comprensione dell'esempio essendo questo il modo di "vedere" T nell'esercizio:

T=

0 0 0 1 0 1	le triple per ciascuna riga sono queste:	2 0 5
1 1 0 1 0 0		3 0 5
0 1 0 0 1 0		0 0 6
0 0 1 1 0 1		0 2 3
1 1 0 0 1 0		4 4 4
0 1 0 1 1 1		3 2 6
0 0 0 1 0 1		2 0 5
0 1 0 1 0 1		3 2 4

Consideriamo la prima riga. La tripla corrispondente 2 0 5 significa che il massimo match della prima riga è nella colonna 2, a partire della posizione 0 ed ha lunghezza 5. E' facile constatare, esaminando T, che questo match c'è. Il fatto che questo sia il match di lunghezza massima della prima riga su tutte le colonne di T è invece più difficile da controllare. Comunque il punto è aver capito cosa viene richiesto. Prendiamo ora l'ultima riga di T, la sua tripla 3 2 4 dice che il massimo match è nella colonna 3, a partire dalla posizione 2 ed ha lunghezza 4.

I valori di R, di C e gli R*C interi da leggere in T di questo esempio li trovate nella vostra home nei input 1 e input2 (sono uguali). Gli output prodotti dai vostri programmi, se corretti, dovrebbero contenere la corrispondente sequenza di triple a destra di T.

Si osservi che per una riga ci potrebbero essere più match della stessa lunghezza massima su diverse colonne o anche su una stessa colonna. In questo caso, si deve scegliere sempre il match sulla colonna di

indice minimo e, per match di uguale lunghezza sulla stessa colonna, si deve scegliere quello che inizia nella posizione più piccola della colonna.

Per contenere le triple che caratterizzano i match massimi, nei file esercizio1.cpp ed esercizio2.cpp trovate la dichiarazione del tipo struttura TRIPLE. Esso è dotato di un opportuno costruttore. Il risultato del programma che viene richiesto è il riempimento di un array di R posizioni di tipo TRIPLE. Nell'esempio appena visto Q , alla fine dell'esecuzione, deve contenere i valori TRIPLE che contengono le triple rappresentate a destra di T.

Parte iterativa: nel main contenuto in esercizio1.cpp c'è l'invocazione della funzione iterativa It0 che va realizzata in modo da rispettare le seguenti pre- e post-condizioni:
PRE_It0=(T ha R*C elementi definiti, R e C sono >0)
void It0(int*C, int R, int C, TRIPLE* Q)
POST_It0=(alla fine Q contiene R valori TRIPLE che rappresentano i match massimi delle R righe di T nelle colonne di T).

It0 **dovrebbe** invocare una funzione iterativa It1 il cui compito è descritto come segue:
PRE_It1=(T ha R*C elementi definiti, 0<=r<R e 0<=c<C)
TRIPLE It1(int*T, int r, int c, int R, int C)
POST_It1=(la funzione restituisce il valore di tipo TRIPLE che rappresenta il massimo match della riga r nella colonna c)[1]
A sua volta It1 dovrebbe invocare un'altra funzione che potrebbe gestire il match della riga r a partire da un certo punto della colonna c.

Si chiede di realizzare It0 in modo che calcoli l'array Q corretto. L'uso di It1 ed eventualmente di una terza funzione dovrebbe risultare in una soluzione più semplice e quindi più apprezzata in termini di valutazione. Si chiede di associare ad ogni ciclo di It0 un invariante e di delineare la dimostrazione di correttezza di It0.

Parte ricorsiva: in esercizio2.cpp, oltre al main, ci sono 2 funzioni ricorsive R0 e R1. La prima è invocata dal main e invoca la seconda. Per entrambe le funzioni sono specificate pre- e post-condizioni.
Si chiede di realizzare la funzione ricorsiva R2 che è invocata da R1. R2 dovrebbe servirsi di una quarta funzione ricorsiva, diciamo R3. Per R2 ed eventualmente R3, vanno specificate pre- e post-condizioni. Si chiede inoltre di dimostrare induttivamente che R2 è corretta rispetto alle vostre pre- e post-condizioni.

[1] In caso ci fossero più match massimi nella stessa colonna, va restituito quello che inizia prima nella colonna.