Esercizio 1 del 25/2, va consegnato corretto entro il 3/3 a mezzanotte

Si tratta di un esercizio ricorsivo diverso dagli esercizi visti finora. Abbiamo una array M ad una dimensione di lim2*lim3 booleani che va "visto" come un array bool [lim2][lim3]. In M i valori true vanno visti come varchi mentre i false vanno visti come ostacoli. Un cammino dalla riga 0 alla riga lim2-1 è una sequenza di lim2 posizioni dell'array che contengono tutte true e tali che l'elemento k del cammino (0<=k<lim2-1) è un elemento della riga k di M e se è l'elemento M[k][i], allora il successivo k+1-esimo elemento del cammino può essere solamente o M[k+1][i-1] o M[k+1][i] o M[k+1][i+1], naturalmente non sempre tutti questi 3 elementi esistono. Se i è 0 ovviamente non potrà esistere M[k+1][i-1] e similmente, se i=lim3-1 non esisterà M[k+1][i+1]. Vediamo un esempio di cammino.

Esempio: sia M come segue, con lim2=4 e lim3=7:
f f t f f t f
t t t f f f t
f t f f t f f
t f f t t f t
Cammini dalla riga 0 alla riga 3 sono [2,1,1,0] e anche [2,2,1,0]. Si osservi che ci sono anche vicoli ciechi, come [5,6] che non può venire prolungato ulteriormente Ovviamente i cammini che cerchiamo hanno tanti elementi quante sono le righe di M. Si osservi che nell'esempio abbiamo usato t/f per true/false, ma in "input" è necessario inserire 0 per false e 1 per true.

Si chiede di scrivere una funzione ricorsiva che cerca e scrive su "output" tutti i cammini che ci sono tra la prima e l'ultima riga di M. La funzione viene invocata da un main (che viene dato) che legge lim2 e lim3 da "input" e poi legge in M i successivi lim2*lim3 valori booleani 0/1 e poi invoca la funzione ricorsiva find_paths (che è da fare) che avrà il seguente prototipo, pre- e post-condizione:

PRE=(M ha lim2*lim3 valori bool definiti, lim2 e lim3 >0, path a lim2 elementi, e .... bla bla)

bool find_paths(bool* M, int lim2, int lim3, int* path, ofstream & OUT, ...)

POST =(se restituisce true c'è almeno un cammino in M che porta dalla prima all'ultima riga e "output" contiene tutti i cammini di questo tipo di M, se non c'è alcun cammino allora "output" resta vuoto o .... qualcosa di simile)

Attenzione: nel caso in cui non ci siano cammini, niente va scritto su "output". E' il main che in questo caso scrive su "output" un messaggio opportuno. find_paths potrebbe aver bisogno di un altro parametro. A voi di aggiungere, in questo caso, le opportune condizioni nella PRE e POST.

Consiglio: è meglio definire una ulteriore funzione ricorsiva cui dovrete associare opportune pre- e post-condizioni.

Correttezza: scrivere la prova induttiva della funzione find_paths.