Esercizio 3 assegnato il 20/2. Da consegnare entro il 24/2 a mezzanotte.

Si tratta di eseguire delle operazioni di pattern matching ma con funzioni ricorsive.

Viene dato un main che esegue le solite operazioni di input. Queste operazioni portano a definire un array T di interi che contiene n valori definiti (in realtà T può contiene più di n valori, ma gli elementi in più servono solo per affinare i test e vanno ignorati nell'esercizio) , un array P di dimP valori e i 3 interi lim1, lim2 e lim3. Come d'abitudine, nell'esercizio T va "visto" come un array int[lim1][lim2][lim3].

Vengono richieste due funzioni **ricorsive** (che sono invocate dal main dato) specificate come segue:

PRE=(T[0..n-1], n>0, è definito, P[0..dimP-1], dimP>0, è definito, e lim1, lim2 e lim3 sono >0)

void match_s(int*T, int n, int lim2, int lim3, int*P, int dimP, ofstream &OUT) che soddisfa la seguente coppia di pre- e post-condizioni:

POST=(T va "visto" come un array int X[lim1][lim2][lim3] e su "output", per ciascuno strato con qualche elemento definito, si deve scrivere il numero di match di P su quello strato; vanno considerati tutti i match anche sovrapposti tra loro;

PRE=(T[0..n-1], n>0, è definito, P[0..dimP-1], dimP>0, è definito, e lim1, lim2 e lim3 sono >0)

void match_ns(int*T, int n, int lim2, int lim3, int*P, int dimP, ofstream &OUT) che soddisfa la seguente coppia di pre- e post-condizioni:

POST=(T va "visto" come un array int X[lim1][lim2][lim3] e su "output", per ciascuno strato con qualche elemento definito, si deve scrivere il numero di match di P su quello strato; vanno considerati solo i match NON sovrapposti tra loro;

Si osservi che le 2 funzioni differiscono per la parte finale del nome (_s e _ns, per sovrapposti e non sovrapposti) a indicare che la prima deve considerare tutti i match mentre la seconda solo quelli non sovrapposti, come specificato nella parte finale delle 2 POST.

Consiglio: è molto conveniente che le 2 funzioni richieste ne invochino altre. **Tutte le funzioni dovranno essere ricorsive. Non è consentito l'uso di alcun ciclo for o while (a parte quelli del main).**

Correttezza: Ogni funzione deve avere una pre- e una post-condizione che descriva quello che la funzione calcola. Scrivete (come commento dopo il programma) una dimostrazione induttiva della correttezza di match_s rispetto alla sua PRE e POST.

Esempio: sia questo il contenuto di "input":
88
1 2 3 4 5 6 7 8 9 0
1 2 3 4 5 6 7 8 9 0
1 2 3 4 5

6 7 8 9 0
1 0 3 4 5 6 7 8 9 0
1 0 1 0 5 6 7 8 9 0

1 0 1 0 1 0 7 8 9 0
1 2 3 4 5 6 7 8 9 0
1 2 3 4 5

1 2 3 4 5
1 2 3 4 0 1 0 1

5 5 5
3 0 1 0

Quindi T va visto come un array [5][5][5] in cui 3 strati saranno completamente riempiti e il quarto contiene solo 13 elementi (2 righe complete e una riga con 3 soli elementi). Il pattern 0 1 0 deve venire cercato su tutti e 4 questi strati, facendo attenzione di considerare solo gli elementi definiti dell'ultimo. Il risultato che va scritto su "output" è che se si considerano tutti i match (anche sovrapposti), ce ne sono 0 sul primo strato, 3 sul secondo, 2 sul terzo e 1 sull'ultimo, mentre se si considerano solo i match non sovrapposti i numeri diventano 0, 2, 1 e 1.