Esercizio a tempo del 26/4/2018 consegna entro le 23:55

Scopo di questo esercizio è scrivere un programma che cerchi un match completo di un pattern in un testo e lo stampi in un formato particolare. Il testo è composto da una sequenza di lettere di lunghezza dimT <= 400 e viene memorizzato nell'array char T[400]. Il pattern è una sequenza di lettere di lunghezza dimP <= 50 che viene memorizzato nell'array char P[50]. Cerchiamo un match completo di P in T, cioè in cui tutti i caratteri di P[0..dimP-1] devono venire trovati in T, ma i match dei successivi elementi di P non devono essere necessariamente in elementi successivi di T. Infatti ci possono essere dei "buchi" che saltano dei caratteri di T tra il match di un carattere P[k] ed il match di P[k+1]. Vediamo un esempio.

Esempio. Se P = abbac è il pattern e T = cabccbacbacbbac è il testo, il primo match completo che esiste è rappresentato nel modo seguente: c[ab]cc[bac]bacbbac
Questo significa che per trovare il primo elemento di P dobbiamo saltare 1 elemento di T, poi dal secondo elemento di T, è possibile di matchare 2 elementi di P, dopo di che c'è un buco di 2 elementi di T, dopo il quale è possibile matchare i restanti 3 elementi di P. Questa sequenza di salti e match la rappresentiamo in questo modo: (1,2)(2,3), cioè, salto 1, prendo 2, salto 2 e prendo 3. L'esercizio chiede di cercare il primo match completo di P in T con buchi. Significa il match che matcha gli elementi di P appena possibile. Vediamo un altro esempio.
Supponiamo che: P sia bbabc e T sia lo stesso di prima. Allora il primo match completo è:
ca[b]cc[ba]c [b]a[c]bbac, chè è rappresentato dalla sequenza di coppie: (2,1)(2,2)(1,1)(1,1).

Per rappresentare le coppie nel programma dato inseriamo una struttura coppia con 2 componenti (salta e prendi) per contenere i valori delle coppie dell'esempio. Si tratta di realizzare una funzione ricorsiva F che soddisfa alla seguente specifica:

L'esercizio richiede di scrivere una funzione **ricorsiva** F che deve soddisfare la seguente specifica:

PRE_F=(T ha dimT elementi definiti e P ne ha dimP, X ha gli elementi che ci servono)
bool F(char* T, char* P, int dimT, int dimP, coppia *X)
POST_F=(restituisce true sse esiste un match completo (con eventuali buchi) d P in T e, se il match c'è, X contiene le coppie che corrispondono al match)

F deve usare 2 funzioni **ricorsive** match e not_match che soddisfano le seguenti specifiche:

PRE_m=(T ha dimT elementi definiti e P ne ha dimP)
int match (char* T, char*P, int dimT, int dimP)
POST_m=(deve restituire la lunghezza del massimo prefisso di P che matcha a partire da T[0] in posizioni contigue, cioè senza buchi)
Si osservi che match può restituire 0.

PRE_n=(T ha dimT elementi)
int not_matc(char*T, char x, int dimT)
POST_n=(restituisce il numero di elementi di T da saltare per arrivare a trovare x in T, partendo da T[0])

Attenzione: Prendere sul serio le PRE e POST date. Scrivere funzioni che le rispettino. Osservate anche il file dato che contiene la definizione della struttura coppia e imposta anche la stampa di X. Osservate che nel main X è inizializzato con dei valori speciali che servono nella stampa per

riconoscere la parte di X da stampare da quella che non contiene coppie utili. Questo serve nella funzione ricorsiva stampa (da fare) che deve stampare in modo appropriato le coppie contenute in X (solo qualora un match completo sia stato trovato).
Ponete molta attenzione a considerare tutti i casi base delle funzioni ricorsive e valutate bene quale sia l'ordine appropriato dei casi base.

Correttezza: Scrivere le prove di correttezza delle funzioni F, match e not_match rispetto alle pre e postcondizioni date.