Esercizio 1 del 13/1/2020

Si tratta di un esercizio di pattern matching in cui il pattern deve essere trovato nel test per intero, ma non in posizioni del testo necessariamente contigue.

Esempio 1. A=[1,0,0,2,1,3,2,0,2,0,1,1,2,0] e P=[0,0,0,1,0]. Esiste un match di P nelle seguenti posizioni di A:

-i primi 2 valori di P, cioè 0,0, sono nella posizione 1 e 2 di A,

-il terzo valore di P, cioè 0, è nella posizione 7 di A,

-il quarto valore di P, cioè 1, è nella posizione 10 di A

-il quinto elemento di P, cioè 0, è nella posizione 13 di A.

Diremo che questo match ha larghezza pari a 13 in quanto inizia nella posizione 1 e finisce nella 13. Questo significa che, visto che P ha lunghezza 5, tra la prima e l'ultima posizione del match ci sono 8 posizioni che non contribuiscono al match, cioè, dalla 3 alla 6, la 8 e la 9, e la 11 e la 12.

C'è anche un altro match di P in A:

-la prima posizione di P è nella posizione 2 di A

-la seconda posizione di P è nella 7 di A

-la terza posizione di P è nella 9 di A

-le ultime 2 posizioni di P sono come prima, cioè nella 10 e 13 di A.

Visto che questo nuovo match inizia dalla posizione 2 e finisce nella 13, come nel primo match, esso avrà lunghezza pari a 12 e quindi minore di quella del primo match.

Si chiede di scrivere un programma che dato A e P (dichiarate e lette nel main dato) calcoli il match a larghezza minima di P in A. Va notato che, a differenza dell'esercizio 2 della settimana scorsa, in questo esercizio si cercano solo match dell'intero pattern P.

Supponiamo che il match a larghezza minima inizi nella posizione i di A e abbia larghezza x, allora il programma dovrà stampare:

"il match migliore inizia in i e ha larghezza x"

se un tale match non esiste, il programma deve stampare:

"nessun match trovato"

Consiglio: conviene usare una funzione ausiliaria che calcola un match a partire da una certa posizione di A.