

Esercizio 1 del 27/4 /2016

Si tratta di realizzare un semplice pattern matching, ma con funzioni ricorsive.

Quindi come al solito il main deve dichiarare `int X[400]` e `int P[20]`, leggere `n_el` ($0 < n_el \leq 400$) e leggere da cin `n_el` interi in `X`, poi leggere `dimP` ($0 < dimP \leq 20$) e leggere `dimP` valori in `P`. Dopo questo il programma deve cercare i match di `P[0..dimP-1]` in `X[0..dim-1]`, considerando match contigui ed anche eventualmente sovrapposti.

Esempio: supponiamo che `X=[2,3,2,3,2,3,2,2,3,0,1,2,3]` e `P=[2,3,2]`, allora il pattern `P` ha 3 match in `X`: a partire dalla posizione 0, a partire dalla posizione 2, a partire dalla posizione 4 e basta. I 3 match condividono un elemento e quindi sono tutti parzialmente sovrapposti. In un caso come questo il vostro programma deve stampare su cout le seguenti stringhe:

match n.1 a partire dalla posizione 0

match n.2 a partire dalla posizione 2

match n.3 a partire dalla posizione 4

n. di match trovati= 3

end

Se invece il pattern fosse `P=[2,3]`, allora con lo stesso `X`, l'output da produrre sarebbe:

match n.1 a partire dalla posizione 0

match n.2 a partire dalla posizione 2

match n.3 a partire dalla posizione 4

match n.4 a partire dalla posizione 7

match n.5 a partire dalla posizione 11

n. di match trovati= 5

end

Le ultime 2 righe di output sono prodotte dal main.

Per realizzare le operazioni richieste il main deve invocare una funzione ricorsiva:

PRE=($0 \leq \text{dim}$, $0 \leq \text{dimP}$, $X[0..\text{dim}-1]$ e $P[0..\text{dimP}-1]$ sono definiti, $i \geq 0$, $\text{count_match} \geq 0$, $\text{vcout_match} = \text{count_match}$)

`int match(int *X, int *P, int dim, int dimP, int i, int count_match)`

POST=(trova tutti i match di $P[0..\text{dimP}-1]$ in $X[i..\text{dim}-1]$ ed esegue le corrispondenti stampe dove il numero del match parte da vcout_match , poi $\text{vcout_match}+1$, poi $\text{vcout_match}+2$ ecc., ritorna il valore finale di count_match)

Consiglio: conviene seguire lo schema già usato in precedenza per il pattern matching e che ora va coniugato ricorsivamente: la funzione ricorsiva match esamina tutti gli elementi di X e, per ciascuno di essi, invoca una seconda funzione (pure ricorsiva) che tenta il match a partire da quell'elemento.

Correttezza: scrivere la prova induttiva di correttezza della funzione match. In caso usiate una seconda funzione ricorsiva, scrivere la sua Pre e Post.