II Compitino di programmazione 13 Marzo 2013

INFORMAZIONI IMPORTANTI: Il programma da fare deve risiedere sul file esercizio1.cpp e deve leggere l'input da input1 e scrivere l'output su output1. Il comando di consegna è consegna esercitazione. Oltre a questo documento, nella home trovate un file esercizio1.cpp che contiene il main. Dovete scrivere le funzioni richieste in questo file.

Programmazione ricorsiva:
Si chiede di scrivere 2 funzioni crea e match che sono invocate dal main che è nel file esercizio1.cpp. Le funzioni devono soddisfare le seguenti specifiche.

La funzione crea deve essere ricorsiva e deve avere il seguente prototipo: void crea(nodo*& L, int dim, ifstream & INP). Essa deve leggere dim (dim >=0) interi da INP e contemporaneamente deve costruire e restituire attraverso il parametro L, passato per riferimento, una lista di dim nodi che hanno come campo informativo gli interi letti.

Esempio: *se dim è 3 e i 3 interi letti sono 0, 12 e 22, allora la lista restituita da crea deve essere L=0->12->22.*

Il tipo nodo è l'usuale struttura struct nodo{int info; nodo* next;};. Si richiede di scrivere una appropriata PRE_crea e POST_crea.

La seconda funzione ricorsiva da realizzare compie un'operazione di pattern matching tra un array P con dimP elementi definiti (leggere dimP >0 e i dimP interi da mettere in P è fatto dal main) e la lista L prodotta da crea. La funzione da fare ha il seguente prototipo: nodo* match(nodo* & L, int*P, int dimP) il suo scopo è spiegato dal seguente esempio. E' molto simile alla funzione F dell'esercitazione del 7/3, ma, diversamente da quella, match deve cercare solo match completi.

Esempio: *supponiamo che L= 2->3->3->1-> 0 e che P=[3,1,2] con dimP=3. In questo caso L non contiene un match completo di P e quindi match deve restituire 0 col return e il valore di L deve rimanere uguale alla lista originale. Se invece P=[2,3,0] allora L contiene un match completo di P e la funzione match deve restituire 2->3->0 col return (che è la lista dei nodi di L in cui è stato trovato il match di P) e quello che resta di L (dopo l'estrazione dei nodi del match) con il parametro L passato per riferimento e cioè: 3->1. Si noti che il match non deve necessariamente essere su nodi contigui. Nell'esempio precedente i nodi 3 e 0 di 2->3->0 non sono contigui in L.*

Quando un match completo di P in L esiste, chiameremo R(L,P) la lista ottenuta estraendo da L i nodi che forniscono il match di P (2->3->0 nell'esempio precedente), mentre con L-P denotiamo i nodi di L che non fanno parte del match (3->1 nell'esempio precedente). Match deve soddisfare le seguenti pre- e postcondizioni.

PRE_match=(L è una lista corretta, L=vL, P ha dimP elementi definiti con dimP>0).

POST_match(se match restituisce col return un valore R diverso da 0, allora esiste un match completo di P in L e R è R(vL,P) e il valore di L è vL-P) && (se match restituisce 0 allora non c'è alcun match di P in L e il valore di L è vL).

Si richiede la dimostrazione induttiva della funzione match rispetto a questa pre e postcondizione.

Attenzione: la condizione che dimP>0 nella PRE_match indica come segnalare al ritorno dalla ricorsione se il match è stato completato o no. Inoltre match non deve né creare né distruggere nodi.