Scritto di Programmazione del 7/9/2016

Data una lista concatenata Q, vogliamo smembrarla in tante sottoliste quanti sono i valori diversi dei campi info di Q.

Esempio 1. Se Q= 2->1->0->10->10->2->0 la smembriamo nelle 4 sottoliste: 2->2, 1, 0->0 e 10->10.

Inoltre vogliamo gestire queste sottoliste in una lista concatenata di nodi definiti come segue:

struct nodoF{FIFO fi; nodoF* next;}

L'idea è che ogni nodoF contiene nel suo campo fi una struttura FIFO che, come sappiamo, è capace di gestire una lista concatenata. Si vuole costruire una lista di nodoF dove ogni nodoF gestisce una delle sottoliste estratte dalla lista originale.

Esempio 2. Vogliamo costruire una lista W di 4 nodoF per contenere le 4 sottoliste di Q dell'esempio 1. Precisamente, il primo nodoF di W deve gestire la sottolista 0->0 di Q, il secondo nodoF di W la sottolista 1, il terzo la sottolista 2->2 e il quarto la sottolista 10->10. Insomma vogliamo che in W le sottoliste di Q siano inserite per valore di campo info crescente.

Le cose date.

Nel programma che viene dato trovate quanto serve per leggere da cin un valore intero dim maggiore di 0 e poi costruire una lista i cui nodi contengono nel campo info i successivi dim valori presenti su cin.

Esempio 3. La lista Q dell'esempio 1 verrebbe costruita con cin contenente 7 seguito dai sette valori 2, 1, 0, 10, 10, 2, 0.

Nel programma dato trovate anche le dichiarazioni di nodo, nodoF e FIFO con rispettivi costruttori, la funzione push_end su FIFO e le funzioni di stampa stampa_L e stampa_F che stampano, rispettivamente, una lista di nodo ed una lista di nodoF, stampando una sottolista alla volta.

Da fare 1)(11 punti) Dovete scrivere una funzione **iterativa** nodoF*smembra(nodo* Q) che deve soddisfare le seguenti pre e post-condizioni.

PRE=(Q è lista corretta di nodo)

POST=(restituisce una lista concatenata di nodoF che contiene le sottoliste di Q come mostrato nell'esempio 2, cioè in ordine di campo info crescenti)

La funzione smembra deve invocare una funzione **ricorsiva** nodoF* inserisci (nodoF*W, nodo*a) che soddisfa le seguenti pre e post-condizioni:

PRE=(W è una lista corretta di nodoF, a è un nodo con campo next=0, vW=W)

POST=(restituisce una lista di nodoF ottenuta da vW aggiungendo ad esso il nodo a) questa asserzione richiede maggiori spiegazioni. Dobbiamo distinguere 2 casi:

i) se in vW c'è già un nodoF che gestisec una sottolista con nodi con campo info=a->info, allora a andrà aggiunto in fondo a questa sottolista.

ii) se (i) non si applica allora sarà necessario aggiungere un nuovo nodoF a vW che gestisca la sottolista composta solo dal nodo a e questo nuovo nodoF andrà inserito nel posto di vW tale che prima di lui ci siano le sottoliste con campo info minori di a->info e dopo di lui quelle con info maggiore di a->info.

Esempio 4. Consideriamo ancora la lista Q= 2->1->0->10->10->2->0 dell'esempio 1. All'inizio delle operazioni di smembramento di Q, la lista di nodoF W da costruire, sarà vuota. Quindi quando cercheremo di inserire in W il primo nodo, chiamiamolo a, di Q (che contiene 2) saremo nell caso (ii) visto prima. Quindi a W verrà aggiunto un nodoF che gestisce la lista col solo nodo a. Il secondo nodo di Q contiene 1 quindi saremo di nuovo nel caso (ii) ed essendo 1 minore di 2, il nuovo nodoF va aggiunto in W al primo posto. Questo nuovo nodoF gestisce la lista che consiste del solo secondo nodo di Q. Col terzo nodo di Q (che contiene 0) succederà la stessa cosa appena vista. Dopo averlo inserito, W conterrà 3 nodoF, il primo che gestisce un nodo con 0, il secondo un nodo con 1 e il terzo un nodo con 2. Il quarto nodo di Q contiene 10 che è diverso e maggiore dei campi info visti finora. Quindi dovremo aggiungere un nuovo nodoF in W (siamo ancora nel caso (ii)), ma il nuovo nodo dovrà essere all'ultimo posto per mantenere l'ordine crescente. Il quinto nodo di Q contiene di nuovo 10 e quindi siamo nel caso (i), cioè l'ultimo nodoF di W dovrà venire modificato in modo da gestire una sottolista con 2 nodi entrambi con campo info=10. L'inserimento in W dei successivi nodi di Q corrisponde sempre al caso (i) e quindi W non aumenterà in lunghezza, ma le sottoliste già presenti con info uguale a 2 e 0 si allungheranno di un nodo.

Da fare 2)(7 punti) Dovete scrivere una funzione ricorsiva nodo* LO(nodoF* W) che soddisfa alle seguenti pre e postcondizione:

PRE=(W è lista corretta di nodoF, vW=W)

POST=(produce una lista di tipo nodo composta concatenando le sottoliste gestite dai campi info dei nodi di vW nell'ordine in cui sono in vW; i nodoF di vW sono tutti deallocati)

Esempio 4. Riprendendo l'esempio 2, dalla W di quell'esempio LO deve costruire la lista:

0->0->1->2->2->10->10

LO può usare funzioni ausiliarie che non contengano né iterazione, né ricorsione.

Attenzione: tutte le operazioni richieste non comportano mai la creazione o la distruzione di nodi di tipo nodo. I nodi della lista costruita dal main vengono prima inseriti in W e poi nella lista prodotta da LO.

Correttezza: (3+5 punti)

1) Dare l'invariante del ciclo principale della funzione smembra.

2) Scrivere la prova induttiva della correttezza della funzione inserisci.