Compito di Programmazione del 20 marzo 2012

Teoria:

(i) Data la seguente funzione ricorsiva, inserire appropriate PRE e POST

```
//PRE= ??
int F(nodo* a) {
if(!a) return 0;
 if (a->left && a->right) return F(a->right)+F(a->left);
 return 1+ F(a->left)+F(a->right);
} //POST=??
```

(ii) Considerate il seguente programma. In caso pensiate sia corretto, spiegate cosa calcola e cosa stampa. Se pensate sia sbagliato, spiegate in dettaglio il motivo.

```
int** f(int *& p){int**x=&p; p[0]--;  p++; return x; }

main() {int b[]={2,3,4,5},*q=b+1; **f(q)=*q; cout<<b[0]<<b[1]<<b[2]<<b[3];}
```

Programmazione: Sia per la programmazione iterativa che per quella ricorsiva, il problema da considerare è quello di eliminare da una lista concatenata alcuni dei nodi che hanno campo info uguale a un valore y dato. Per la programmazione iterativa vanno mantenuti i primi k nodi con campo info=y e distrutti i successivi (se ci sono), mentre per la programmazione ricorsiva vanno mantenuti gli ultimi k nodi della lista e distrutti i precedenti (se ci sono). Si osservi che nel caso k sia 0 allora tutti i nodi con info=y andranno eliminati. Se invece la lista non contiene più di k nodi con info=y, allora la lista non deve essere modificata.

Esempio. Se la lista L è 3->2->3->2, k=2 e y=3, allora la lista non va cambiata in nessun caso essa contiene solo k=2 nodi con info=3. Se invece L= 3->2->3->2->3 ->3, k=2 e y=3, allora, se si devono mantenere i primi k=2 con info=3 ed eliminare i successivi nodi con info=3, la lista risultante sarà, 3->2->3->2, mentre se vanno mantenuti gli ultimi k=2 nodi con info=3 ed eliminati gli altri, allora la lista risultante sarà: 2->2->3 ->3.

Vediamo nel dettaglio i 2 esercizi.

Attenzione: Si osservi che nessun nuovo nodo va creato. I nodi eventualmente eliminati dalla lista iniziale dovranno sempre essere deallocati.

Esercizio iterativo: la funzione iterativa da realizzare deve avere il seguente prototipo:
void F(nodo*L,int k, int y) ed essa deve soddisfare alla seguente coppia di asserzioni:

PRE=(L è una lista corretta e non vuota, k >0, e y è definita, sia vL=L)

POST=(L è una lista ottenuta da vL nel modo seguente: sia N il n. di nodi di vL con campo info=y, allora, se N-k>0, L è ottenuta da vL eliminando gli ultimi N-k nodi con campo info=y, altrimenti L=vL) &&(tutti i nodi eliminati da vL sono stati deallocati)

Si richiede di trovare un invariante per il ciclo principale della funzione F. Per scrivere questo invariante può essere utile avere una funzione che scandisca le iterazioni effettuate (che corrispondono ai nodi della lista esaminati) e può essere anche utile la notazione vL[c]= i primi c nodi di vL. L'invariante richiesto può venire ricavato con la ricetta di indicizzazione dalla POST data.

ATTENZIONE: è indispensabile osservare, meditare e usare le assunzioni contenute nella PRE. Esse permettono di semplificare F.

Esercizio ricorsivo: la funzione ricorsiva da realizzare deve avere il seguente prototipo:
int G(nodo*&L,int k, int y) e soddisfare la seguente coppia di asserzioni;:

PRE=(L lista corretta e event. vuota, k>=0, y definita, vL=L)

POST=(G restituisce il numero N di nodi con info=y presenti in vL) && (L si ottiene da vL come segue: se N è il numero dei nodi di vL con info=y, allora, se N-k>0, L è ottenuto da vL eliminando i primi N-k nodi con campo info=y. Se invece N-k<=0, allora L=vL) &&(i nodi eliminati da vL sono tutti deallocati)