Esame di Programmazione del 15/7/2019 **Parte iterativa**

Si chiede di scrivere una funzione iterativa che inserisca un nuovo nodo in un albero BST. Si ricorda che un albero BST soddisfa la seguente proprietà: per ogni nodo r dell'albero, il sottoalbero sinistro di r contiene nodi con campi info minori di r->info, mentre i nodi del sottoalbero destro di r hanno campi info maggiori di r->info. Non si verificherà mai che nodi diversi abbiano lo stesso campo info. Oltre al campo info, i nodi dell'albero da costruire hanno un campo intero num che dovrà essere instanziato, per ogni nodo r al numero di nodi dell'albero radicato in r (r compreso).

Esempio: sia dato l'albero BST [9,4]([3,2](_,[4,1](_,_)),[10,1](_,_)), dove ogni nodo è rappresentato dai suoi 2 campi, info e num. Per esempio la radice è [9,4] che indica che il campo info della radice è 9 e il campo num è 4. Infatti l'albero intero contiene 4 nodi. Si osservi che le foglie hanno ovviamente num=1. Si consideri ora la richiesta di inserire un nuovo nodo con campo info=8 in questo albero. Per mantenere la proprietà BST, il nuovo nodo dovrà diventare il figlio destro di [4,1]. Il cammino da percorrere è dettato dal confronto del valore 8 con i campi info dell'albero. Confrontando 8 con il campo info della radice che è 9, si decide che il nodo nuovo va inserito nel sottoalbero sinistro della radice e così via per i successivi nodi percorsi. La proprietà BST permette di far si che la discesa nell'albero sia senza possibili errori e quindi che sia possibile eseguirla con un ciclo iterativo.

L'albero BST ottenuto dopo aver aggiunto un nuovo nodo con info=8 è il seguente: [9,5]([3,3](_,[4,2](_,[8,1](_,_)),[10,1](_,_)). Si osservi che sono aumentati i campi num dei nodi attraversati durante l'operazione per tenere conto del nuovo nodo.

Funzione da fare: si chiede di realizzare una funzione iterativa:
void build_BST(nodo*&r, int x) che soddisfa la seguente coppia di PRE e POST condizioni:

PRE=(albero(r) è un albero binario benformato e BST in cui il campo num di ciascun nodo è corretto, Vr=albero(r), x è un qualsiasi intero)

POST=(albero(r) è ben formato, BST e con i campi num corretti ed è ottenuto da Vr aggiungendo il nodo con info=x come foglia)

Si osservi che albero(r) può anche essere vuoto.

Correttezza: specificare l'invariante del ciclo principale della funzione build_BST e dimostrare, usando l'invariante, che il ciclo è corretto.