CdL in Scienze dell'Informazione, sede di Cesena

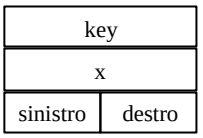
Sistemi per l'Elaborazione dell'Informazione 1

appello: 8 giugno 2000

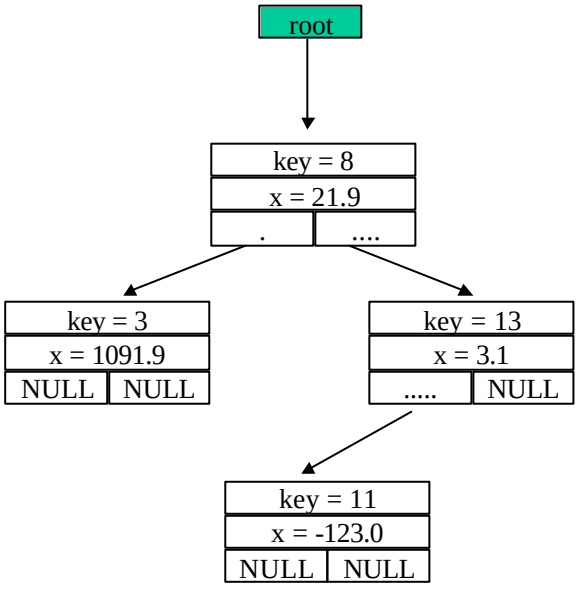
ESERCIZIO NUMERO 4

Data una struttura ad albero binario, implementata mediante la struttura dati NODO seguente:

```
typedef struct s_NODO{
  int          key;
  double       x;
  s_NODO       *destra;
  s_NODO       *sinistra;
  }  NODO;
```



Sia root la radice dell'albero (NODO *root), root assume valore NULL quando l'albero è vuoto, ogni nodo può avere due figli puntati dai puntatori destra e sinistra, che assumono valore NULL se il figlio corrispondente non esiste.
I nodi dell'albero sono ordinati secondo la chiave key.
Non esistono due nodi che abbiano la stessa chiave key.
L'albero può essere vuoto.



Implementare la funzione

int ***cerca***(int KEY, double *pX, NODO *pnodo);

che verrà chiamata ad. es. nel modo seguente:

```
int          ris, KEY;
double       X;
KEY = 13;
ris = cerca ( KEY, &X, root);
```

La funzione ***cerca*** scorre l'albero binario che discende dal parametro pnodo, alla ricerca di un nodo che abbia chiave key uguale a KEY.
Se tale nodo esiste, la funzione scrive il valore x del nodo nella locazione di memoria puntata da pX, e restituisce 1.
Se tale nodo non esiste, la funzione restituisce 0.
OSS. Il puntatore *pnodo* passato come terzo argomento alla funzione può essere NULL.
OSS. nell'esempio descritto la funzione restituisce 1 e assegna il valore 3.1 ad X.

SOLUZIONE dell'ESERCIZIO 4 (C)

```
int cerca( int KEY, double *pX, NODO *pnodo )
{
        if(pnodo)        {
                if( pnodo->key==KEY ) {
                        *pX=pnodo->x;
                        return(1);
                }
                else {
                        if( pnodo->key > KEY )
                                return (   cerca ( KEY, pX, pnodo->sinistra )   );
                        else
                                return (   cerca ( KEY, pX, pnodo->destra )   );
                }
        }
        else
                return(0);
}
```