Esercizio n.1. (sbarramento)

Si definisca una funzione ricorsiva C che, dato in input un vettore di interi v, un indice di inizio del vettore i, un indice di fine j e un intero diff, controlla se, per ogni elemento, la differenza tra un elemento e il precedente è minore di diff.La funzione restituisce 1 se la proprietà è vera, 0 altrimenti. La prima chiamata della funzione si suppone avvenga con i=0, j = n-1, se gli elementi sono n.
La funzione ha il seguente prototipo:

int **DiffSucc**(int* v,int i,int j, int diff)

Non si dimentichi di precisare pre e post condizioni!

Esempio: v = 1,4,6,-2,-1
diff = 5
Poichè 4-1 < 5, 6-4 <5, -2 -6 <5 e -1-2 <5 la funzione restituirà 1.

Esercizio n.2.

Si definisca una funzione C che restituisce il numero di nodi contenti nel'albero di input, -1 se quest'ultimo è vuoto. Un nodo e' contento se nell'albero in esso radicato ci sono piu' nodi figli sinistri che destri. La funzione ha il seguente prototipo

int **contenti**(TreePtr tPtr);

Sol. Es. 2

```
int contenti(TreePtr  tPtr)
{ int diff, nCont = 0;
if (!tPtr) return 0;
diff = contAux(tPtr,&nCont);
printf("la differenza e' %d",diff);
return nCont;}


int contAux(TreePtr  t, int* nCont)
/*post:  calcola in nCont il numero dei nodi contenti in t, restituendo la
differenza tra il numero dei nodi figli sinistri e quello dei figli destri.
prec: t!=NULL*/
{int diff;
if (t->left)  diff =  contAux(t ->left,nCont)  + 1;
/* rientro da un figlio sinistro, quindi incremento di 1*/
else diff = 0;
if (t->right) diff += contAux(t ->right,nCont) - 1;
/* rientro da un figlio sinistro, quindi decremento di 1*/
if  (diff > 0)  (*nCont)++;
return diff;}
```