```
public class QSort{
/* ------------ Stato oggetti: Statica ---------------------- */
/* ------------ Operazioni statiche ------------------------- */

public static boolean ord(int t1, int t2){
    return (t1<t2);
}
public static int separatore(myIntMutSeqRW u) throws IllegalArgsException{
    if (u==null) throw new IllegalArgsException("separatore");
    return u.val();
}
public static boolean eqOrd(int x, int y){
    return ((x==y)||(ord(x,y)&&ord(y,x)));
}
private static Pair divisione(myIntMutSeqRW inf, myIntMutSeqRW sup, int a)throws IllegalArgsE
    boolean met = false;
    myIntMutSeqRW inf2 = inf;
    myIntMutSeqRW sup1 = sup;
    Pair pair;
    met = met || (sup1 == inf2);
    while (!met) {
        int x = inf2.val();
        while ((inf2!=sup) && (ord(x,a)||eqOrd(x,a))) {
            inf2 = inf2.succ();
            x = inf2.val();
            met = met || (sup1==inf2);
        }
        int y = sup1.val();
        while ((sup1!=inf) && ord(a,y) && (!eqOrd(x,a))) {
            sup1 = sup1.pred();
            y = sup1.val();
            met = met || (sup1==inf2);
        }
        if (!met) {
            sup1.swap(inf2);
        }
    }
    pair = new Pair();
    if (inf2 == sup1) sup1 = sup1.pred();
    pair.left = sup1;
    pair.right = inf2;
    return pair;
}


static void QSort(myIntMutSeqRW cInf, myIntMutSeqRW cSup)throws IllegalArgsException{
    /* vedi Versione3 C: ordina gli elementi della sottosequenza [cInf,cSup],
       lasciando invariati tutti gli altri, se presenti.
    */
    if (cInf == cSup) return;
    {
        int a = separatore(cSup);
        Pair sup1inf2 = divisione(cInf,cSup,a);
        QSort(cInf,sup1inf2.left);
        QSort(sup1inf2.right,cSup);
    }
}
}

class Pair{
```

```
    myIntMutSeqRW left;
    myIntMutSeqRW right;
}
```