Locatives in C++

CSE 326: Data Structures

January 5, 1998

The section of the Lewis and Denenberg textbook on Locatives is somewhat unusual, but necessary to understand some of the algorithms that arise later in the book. In order to help you understand locatives, the program fragment below implements Algorithm 1.6 in C++. For this to be helpful, you will have to look carefully at the differences between the textbook's algorithm and the function `LLInsert` below.

```
/****************************************************************
 * Implementation of Lewis and Denenberg's Algorithm 1.6 in C++,
 * using pointers to simulate locatives.
 ****************************************************************/

typedef int key;

struct node
{
    key  Key;
    node *Next;
};

typedef node *list;

list newlist(key K, list L)
// return a pointer to a new node with fields K and L
{
    list nodeptr = new node;
    nodeptr->Key = K;
    nodeptr->Next = L;
    return nodeptr;
}

void LLInsert(key K, list *P)
// Insert a node containing key K in sorted list *P if none exists already
// Note: P is a pointer to a pointer to a node
// Understand why one assignment is of the form P = ... and one *P = ...
{
    while (*P != NULL && (*P)->Key < K)
        P = &((*P)->Next);
    if (*P != NULL && (*P)->Key == K) return;
    *P = newlist(K, *P);
}
```