LAddRow Insert row(s) of empty cells into a list

#include <Lists.h> **List Manager Package**

short	**LAddRow(***count*, *rowNum*, *theList* **)**;	
short	*count* ;	how many rows to insert
short	*rowNum* ;	where to start inserting
ListHandle	*theList* ;	handle leading to a ListRec
	returns	row number of the first inserted row

LAddRow inserts one or more rows of empty cells into a list. If drawing is on, the list display and the vertical scroll bar (if any) are updated.

count is the desired number of rows to insert.

rowNum specifies where to start inserting rows. Rows are inserted *before* this row. For instance, if *rowNum*=3 and *count*=1, then rows 3...*n* are renumbered as rows 4...*n*+1. Thus, the cell that used to be called (0,3) is now called (0,4), and so forth.

If *rowNum* > ListRec.dataBounds.bottom (i.e., greater than the current height), exactly *count* rows are added to the bottom of the list. The row where they were actually added is returned.

theList is a handle leading to a variable-length ListRec structure. It is a value previously obtained via **LNew**.

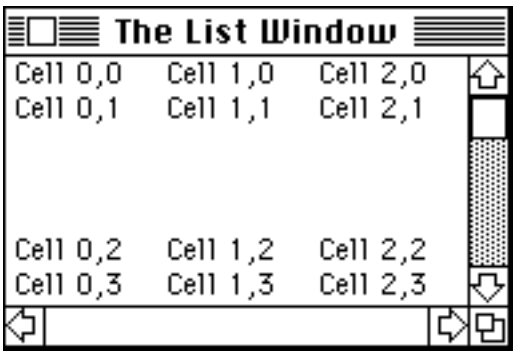
Returns: a short; the row number of the first new row inserted. When inserting within the array bounds, this simply returns *rowNum*. But, if you attempt to insert beyond the current bounds, the return value is the current vertical size of the list (i.e., ListRec.dataBounds.bottom).

Notes: **LAddRow** increases the size of the ListRec structure by (*count* * ListRec.dataBounds.right) * 2 bytes. ListRec.dataBounds.bottom is increased by *count*.

For instance, after:

```
LAddRow( 3,2, theList );          /* insert 3 rows at row 2 */
```

The list shown in the **LNew** example would look like:



Note that if there are no rows or columns (as when *rDataBnds* is empty when you call **LNew**), you must insert at least one column or row (via **LAddColumn** or **LAddRow**) before starting to store cell data via **LSetCell**.