Dynamic List

Date assigned:	Monday, October 31, 2011
Date due:	Friday, November 11, 2011
Points:	50

We are re-implementing the List ADT used in the Static List project backed with a **doubly linked list** rather than an array. All of the functions operate the same; all of the ERROR codes are the same with the same precedence.

You can find the new **list.h** header file on Zeus in **/home/CS300Public/2011/dynamicList**.

In addition to implementing the list data structure, you must provide a Makefile and test driver (listDriver.c that produces an executable named listDriver) that thoroughly testes your list functions. You should be able to reuse your list driver from the Static List project, with perhaps a few modifications.

You may add any helper functions you need to list.c. You may not alter list.h in anyway.

1. Your code is to be written in C using Eclipse 3.7. Programs written in other environments will not be graded. Create an Eclipse project named cs300_dynamicList_PUNetID. This project should contain the directories: src, include, and bin. The driver, listDriver, should be created at the top level of the project, not in the bin directory.

2. The Makefile must contain the necessary targets to build the listDriver as well as a clean and dist target similar to the identically named targets in your Stack assignment. There is no testMe target required for this assignment. Typing **`make`** on the command line should build listDriver.

3. Submit a file called cs300_dynamicList_PUNetID.tar.gz into the CS300 Drop Box by 9:15am on the day in which the assignment is due. Submit a color, double sided, stapled packet of code by that same deadline. The packet should be in the following order:
   - Makefile
   - List Driver (.h then .c if you have both, otherwise just .c)
   - list.c (do not print list.h)
   - Any extra .h/.c pairs you have.

4. Test one function at a time. This will lessen your level of frustration greatly.

5. You are to use the coding guidelines from V6.0 of the coding standards.

Goals for this assignment:

1. Code and test your program one function at a time.
2. Write efficient/clean code
3. Use the debugger to effectively develop a correct solution
4. Thoroughly test your code.

The list.h header file as well as a list of ERRORCODEs that each function can produce are attached. Further, the ERRORCODEs are listed in order of precedence. If a function could produce multiple ERRORCODEs, the function must return the ERRORCODE highest on the list.

The meaning of the ERRORCODEs are given in the header file.

Since the interface for the list may be unclear, here is a very small example of how to walk a list and print out every element. For brevity, no error checking is done.

```
List sTheList;
DATATYPE theData;

// create and fill the list

lstFirst(&sTheList, &theData);

while( NO_ERROR == lstNext(&sTheList, &theData) )
{
  // print theData
}
```

There is only one deadline. I expect you to start this project this early and have questions soon. My solution to this assignment required over 1000 non-comment lines of code.

This project is significantly more difficult to debug than previous projects! Start early!