CSCD 340
Lab 1

One of the most important data structures that will be used is a Linked List. It is important that we develop a very generic linked list that we can use during the course of the quarter.

Node Specifics

- void pointer for data
- next pointer
- prev pointer
- Declared in the header file named linkedList.h

LinkedList Specifics

- Node pointer for head (Dummy Head)
- int size
- Declared in the header file named linkedList.h
- NOTE: The List will **NOT** be circular

Specific Information

- You will not use gets – use scanf, fscanf, fgets
- For the specific functions I have provided documented .h files. The comments are Javadoc style. Look through the .h files.
- I have provided the menu functions and the file util functions. These are provided as .o files.
- I have provided a Makefile. If you are using a 32 bit system you will need to comment out the line pertaining to 64 bit .o files and uncomment the 32 bit line.
- You must create
  - books/books.c
  - linkedlist/linkedList.c
  - linkedlist/listUtils.c
- I have provided .h files. You CANNOT change them in any fashion. If you need to add anything add it to the requiredIncludes.h
- The purpose is to create a generic linked list with a void * and function pointers and the type should not matter. In our case the type is a book.

  A book contains
  - char * title
  - char * ISBN
  - int pages

    - An array of Authors
        - An array of name structure
        - int total for the number in the array

        - A Name structure contains
            - char * first
            - char * last

    - Publisher reference
        - char * name
        - char * city

- You will create an output file that shows the run of the program using valgrind. This output file should show that no memory is being leaked.
- I have provided unchangeable cscd340Lab1.c

FILE SPECIFICATIONS

# of Books in the file
Title
ISBN
Pages
Pub Name
Pub City
Number of Authors
One Author per line

TO TURN IN

There are multiple turn ins for this lab.

1. Your stubbed out methods ensuring all code compiles will be submitted via git commit to your repository on GitHub. This is required by the date and time specified in canvas).

2. A zip file, in Canvas, by the required due date containing:
    - all files required to compile and run your code
    - all input file(s) used to test your program program
    - all output file(s)
    - a test run named cscd340Lab1val.txt – testing all aspects including showing your code is leak free.
    - We should be able to download the zip and compile your code, and then run your code.
    - Name your zip, your last name first letter of your first name lab1.zip (Example: steinerslab1.zip)