CS262, Lab Assignment 10:
Linked List

Due: Sunday, April 24 at 11:59 pm ET

Description:

In this assignment you will write a program to create and manipulate a simple linked list.
For each node in the linked list,

- you will generate a random number,
- create a node that holds that number,
- and insert that node into its sorted position (ascending) in the linked list.

Once the nodes are inserted, you will traverse the list, printing the value held in each node. Then you will clean up the list (deleting all the nodes) and exit the program. You will also use valgrind to help you check for memory errors in your code.

Instructions:

The source file will be named `lab10_<username>_<labsection>.c`

> Define a **struct** appropriate for holding:
> - A random integer
> - A "next" pointer to reference a separate instance of the struct

Use the **`typedef`** keyword to define a new *typename* for the previous struct.

The program will get 3 integers as CLA (i.e. `argc` should be 4)

1. A random number seed (`S`)
2. The number of random numbers to generate (`N`)
3. The Maximum Possible Value of the Random numbers generated (`Max_val`)

- The program will use a pointer to the head of the linked list.
  - The head of the list will be accessed through a pointer, and the data contained in the head of the list will be used for sorting purposes.
- Your program ***must*** contain the following functions:
  - `insertNodeSorted()` - You may implement this either of two ways.
    (1) Use the head of the list as one parameter and the integer value as the second. The function will allocate memory for the node, initialize it and then insert it in the proper location (sorted in ascending order) in the list.
    (2) Pass the head of the list as one parameter and a previously allocated and initialized node as the other. The existing node is inserted in the proper location (sorted in ascending order) in the list.
  - `printList()` - Takes the head of the list as its only parameter, traverses the list, printing out the data in sorted order.
  - `deleteList()` - Takes the head of the list as its only parameter, traverses the list, deleting all nodes.

- The basic algorithm of your program is
  - Seed the random number generator (`S`)
  - Use a loop (the number of times is based the value `N`) that will:
    1. Generate a random number from [0, `Max_val`] /* inclusive */
    2. Print the random number
    3. **Create a node** that contains that random number
    4. **Insert the new node** in sorted order into the existing list
  - Print the sorted linked list
  - Delete the linked list

Include input validations to avoid segmentation fault when the parameters are not given by CLA
Validate the input values for each parameter (`S,N,Max_val`) are > 0

Testing using Valgrind

Valgrind is a suite of programs that you can use to improve your programs. For this assignment, you will be using a tool called Memcheck.

- Read a quick-start tutorial at this link: http://valgrind.org/docs/manual/quick-start.html
- Read the Overview for Memcheck at this link: http://valgrind.org/docs/manual/mc-manual.html

When compiling your program with `gcc` don't forget to use the `-g` option and run your exe as:

```
valgrind --leak-check=yes lab10_<username>_<labsection>
```

The Heap, Leak and Error Summaries of valgrind should show that there are no errors or issues.

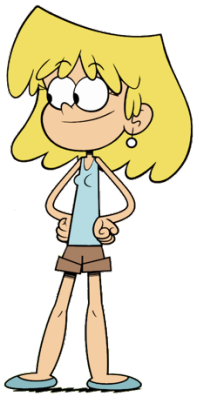
Makefile:

Modify the Makefile you used for the previous assignment so that it works for this assignment.

Submission:

You will submit a typescript file (showing that your program compiles without errors)

1. Create a typescript file named `lab10_typescript_<username>_<labsection>`
2. Show that you are logged onto Zeus
3. Compile the code using your `Makefile`
4. Run the code given these options:
   ```
   valgrind --leak-check=yes lab10_<username>_<labsection>
   valgrind --leak-check=yes lab10_<username>_<labsection> -1 2 4
   valgrind --leak-check=yes lab10_<username>_<labsection> 262 15 100
   ```
5. Delete the executable using the make clean command
6. End the typescript
7. Be sure your directory ONLY contains the *source file*, *script* and `Makefile`
8. Create a `tarfile` of your `lab10_<username>_<labsection>` directory
9. Submit the `tarfile` to Blackboard
10. Verify that your submitted `tarfile` can be extracted and it's the right `tarfile`.



Congratulations! You have completed your assignment