COMP 2401 -- Tutorial #6

Linked Lists

Learning Objectives

After this tutorial, you will be able to:

- create and manipulate singly linked lists

Tutorial

1. Download the file `T06.tar.bz2` from the tutorial page in *cuLearn*. Extract and read through the tutorial files.

   This week's tutorial documents are compressed using another common compression tool: `bzip2`. Read the `man` page for `bzip2` (and `bunzip2`) and extract the files needed for this tutorial.

   Remember when compiling that there are more than one source file. You may also want to include the `-g` *flag if you plan on using* `valgrind` *to debug (highly recommended). Be sure to also use the* `-Wall` *flag to alert you to any additional warnings in your code.*

2. Create a `StudentNode` type for use in a linked list which is compatible with what is already written.

3. Draw a memory map diagram (like the ones seen in class) which represents a call and return from the `addStudent()` function. Your diagram should include the function call stack, heap, and all relevant pointers.

4. Implement the functions `addStudent()` and `printList()` as prototyped.

   When adding a student, it should be to the front of the list.

5. Write a function `appendStudent(StudentList *stuList, StudentNode *a)` which adds a student to the end of a list, always in the last position. Remember to consider all possible cases (empty list, inserting at the beginning, at the end, etc.)

Exercises

1. Write a function `popStudent(StudentList *stuList)` which deletes the student in the first position from the list.

2. Write a function `deleteStudent(StudentList *stuList, int pos)` which deletes the student in the given position from the list.

3. Alter the `StudentList` and `StudentNode` types so that the list becomes doubly linked. Make any necessary changes to the list functions so that they work correctly.