Concurrent and Distributed Computing
Tutorial 2
CPS3227

Josef Magri
Adrian Debarro

February 24, 2017

Tutorial: C Language Refresher

In this tutorial we will be using a data structure called the *doubly linked list*. A doubly linked list is a linked data structure that consists of a set of sequentially linked records called nodes. Each node contains two fields, called links, that are references to the previous and to the next node in the sequence of nodes (Wikipedia). For the scope of this tutorial, the doubly linked list will be used to manage student records.

This tutorial is divided in three parts: (a) create a structure for storing student records, (b) create the doubly linked list to manage multiple student records and, (c) provide functionality for storing and retrieving student records to and from disk.

a. Create a structure for representing a single student record. Minimally, the structure must contain two pointers: `Next` pointing to the next student record in the linked list and `Prev`, which points to the previous entry. Note that in the case of head and tail nodes, `Next` and `Prev` cannot be both valid. For convenience the `struct` is given below:

```
typedef struct
{
  int Age;
  char *Name;
  float Grade;
  Level StudentLevel;
  StudentNode *Next, *Prev;
} StudentNode;
```

Note that `Level` is an enumeration; it is provided below:

```
enum Level
{
  GRADE_1 = 0,
  GRADE_2,
  GRADE_3,
  GRADE_4,
  GRADE_5
};
```

b. The data structure represents a collection of student records; provide functionality to modify the collection together with a shell user interface that allows a user to interactively do so. More specifically, the following functionality must be implemented:

Create Add a new student entry

Push Back Add a student entry to the end of the list

Insert Insert a student entry at a specified position in the list

Modify Modify the student entry at a specified position in the list

Delete Delete the student entry at a specified position in the list

Display Print the list contents as formatted output

c. Provide functionality for storing and retrieving the student records. In particular, implement the following two functions:

`Store` To persist the student record to a file called `students.lst`.

`Load` Clears the current linked list in memory and repopulates it from the student records stored in `students.lst`.

The student records should be stored in a text file the format of which is given below:

```
13,John,1,65
14,Tom,2,55
...
```