Indian Institute of Technology Bombay
Department of Electrical Engineering
EE-717 Advanced Computing for Electrical Engineers

Assignment 4

Submission Deadline: October 1, 2015 (Thursday), 11:55 pm (IST)

The goal of this assignment is to implement linked lists using pointers, to come up with an algorithm for a problem on an array and to find components of a graph.

Part 1:

Solve the programming problems given on the platform. First problem is on using pointers to implement linked lists. Problems 2 is an operation on a given array. Problem 3 is on connected components of a graph. The deadline for Part 1 is on the platform and it is October 8 Thursday at 11:55pm.

Start part 2 only after part 1 is over.

Part 2:

In the accompanying tarball placed in moodle you will see testcases for the linked list problem. Problems 2 and 3 do not have any Part 2 component.

In the subsequent sections, we assume that you are using a Unix based environment. The following commands were tested in Ubuntu 14.04 LTS. However, any compiling environment with standard libraries installed should be ok.

2.1 Download

Download "assignment4.tar.gz" from the moodle page. Open a terminal. Use the following command at the command prompt ($) to uncompress the file:

```
$ tar xvf assignment4.tar.gz
```

Note that anything that follows the "$" sign should be typed on the command prompt of the terminal.

This should create a directory called *assignment4.* Move to the directory by typing

```
$ cd assignment4
```

2.2. What is in the directory?

You will see a .c file called llist_template.c. You will also see a directory with the input files named input<num>.txt. These are input files on which you have to test your program.

1) Run

```
$ make llist
```

to build llist executable and

There is a file called *Makefile* which is used by programmers on Unix like platforms to automate compilation of large projects. Any edit that you do to the source file llist_template.cpp must be followed by running the appropriate *make <name>* on the command line to create a new executable.

As of now, the executables do not do any work. Once you are done with the submission on the platform, cut and paste the appropriate code into the functions in the respective files. Specifically, do not overwrite the main functions in the template. This part of the experiment assumes that you have already tested correctness against the testcases provided on the platform. The input testcases are for a larger set.

2) Check if you have md5sum installed. It is a utility to make one way hash of a given file.

```
$ which md5sum
```

If you get an output like “/usr/bin/md5sum” then you have it installed. Otherwise, go ahead and install it using

```
$sudo apt-get install coreutils
```

We need md5sum installed so that you can go to step 3.

3) The file runLList.sh is a script to run the executables on all inputs.

Run

```
$ ./runLList.sh
```

You will notice that the script will run the executables with all the inputs and will print a string that looks like gibberish. An example string might look like:

```
c86faa20cd3e17974dbf5af6f3918f0b -
```

Note that the executables should be filled up with your actual code. Running the script on the empty template is not going to give meaningful results.

2.3. What to submit?

A google form is set up at http://goo.gl/forms/Lj255AgvCs where you can put in the string that is printed by the shell script for each input file.

You will be asked to enter one string per input file. Enter the string without the "-" sign at the end. Ensure that you do not leave any characters out of the string that is printed by the shell script. You will have to login into your gmail account for the submission of google form.

Caution: If your program does not terminate on some input file, subsequent files may not get a chance to run and you may have incorrect strings for all the files. It is necessary that your programs terminate for all the input files before you generate the strings for each input file.