# COMP 116 Data Structures

Lab #7

In this lab, we will do a few timing tests to compare the behavior of `ArrayIntList` and `CircArrayIntList`. In the process, you will have to use the various methods of the classes I programmed in front of you in class.

1. First, download Lab07.zip from the schedule lab of the course website, and decompress the content in the folder that contains your CLion projects. The file contains a project folder with all the files necessary to use lists, array lists and circular array lists.
2. Look at the function timingTest1 and its two helper functions. The first helper is used to run the tests with `ArrayIntList`, the second helper is used to run tests with `CircArrayIntList`. From that file, you should be able to understand how to use the timing functions from the `ctime` library. All the other tests should follow the same format, that is, running the tests on lists of size 10000, 20000, 40000 and 80000 for both `ArrayIntList` and `CircArrayIntList`.
3. Run the first test on your computer and adjust the number of repetitions in both helper functions so that both helpers use the same number of repetitions, and the first test for the ArrayIntList takes about 100 milliseconds. Write the times in the table on the other side of this sheet.
4. Write the code for `timingTest2`. Each test should first create a list of the appropriate size (before starting the timer), then within the timed zone, call the methods `getSize, isEmpty, peekHead, peekTail, at` and `operator[]`.. Call the `at` and `operator[]` on the index in the middle of the list. All these methods are *really* fast, so you will probably need to crank the number of repetitions way up so that the first test takes about 100 milliseconds.
   Run your test and note the results in the table.
5. Write the code for `timingTest3`. Within the timed zone, each test should create an empty list and insert `size` elements in the list using `insertAtTail`. If you need to do multiple repeats, remember to reset the list to an empty list at the beginning of each repetition.
   Run your test and note the results in the table.
6. Write the code for `timingTest4`. Within the timed zone, each test should create an empty list and insert `size` elements in the list using `insertAtHead`. As we have seen in class, operations at the head are quite slow for `ArrayIntLists`, so you may not need any repeats for the test to last more than 100 milliseconds.
   Run your test and note the results in the table.
7. Write the code for `timingTest5`. Each test should first create a list of the appropriate size (before starting the timer), then within the timed zone, remove all the elements form the list using `removeHead`. Most likely, no repeats will be needed for this test either.
   Run your test and note the results in the table.

**Results** (all times should be in milliseconds)

| | Repeat # | `ArrayIntList` | | | | `CircArrayIntList` | | | |
|---|---|---|---|---|---|---|---|---|---|
| Size | | 10000 | 20000 | 40000 | 80000 | 10000 | 20000 | 40000 | 80000 |
| Test 1 | | | | | | | | | |
| Test 2 | | | | | | | | | |
| Test 3 | | | | | | | | | |
| Test 4 | | | | | | | | | |
| Test 5 | | | | | | | | | |

Given the running times for each of these functions we have discussed in class, can you justify the times you found in the table? In particular, can you figure out the big-O running time of each of the tests?

When you are done, write your name on the sheet and hand it to the lab instructor.

Name: ______________________________________________