CpSc 2120: Algorithms and Data Structures

Instructor: Dr. Brian Dean
Webpage: `http://www.cs.clemson.edu/~bcdean/`
Handout 2: Lab #1

Fall 2014
MWF 9:05-9:55
Vickery 100

1 Linked List Practice

The goal of this lab is to help everyone get back up to speed with programming in C/C++ and to re-familiarize ourselves with linked lists. This is intended mostly to be a review of material you should have seen in earlier courses.

In the directory:

`/group/course/cpsc212/f14/lab01`

you will find three files, `main.cpp`, `intset.cpp`, and `intset.h`. Make a copy of these files in your own directory space. You will need to modify and submit `intset.cpp` and `intset.h`; you should not need to modify the file `main.cpp`.

If you look at `intset.h`, you will find the definition of a simple class that implements a data structure for representing a set of integers. The members of this class allow you to perform the following operations:

- `find(key)` : Return true if key is present in the set, false otherwise.
- `insert(key)` : Insert a new integer into the set. It is an error to call this function with a value of key that is already present in the set.
- `remove(key)` : Remove an integer from the set. It is an error to call this function with an integer not present in the set.
- `print()` : Print the contents of the set in sorted order.

The code in `intset.h` and `intset.cpp` currently provides a straightforward implementation of these functions using an array as the underlying representation of the set. Elements are stored in sorted order within the array, allowing the `find` function to run very quickly using binary search. If the memory allocated for the array fills up due to a large number of *insert* operations, a new array is allocated of twice the original size.

In this lab, your goal is to replace the underlying implementation of the integer set with one based on a linked list instead of an array. The linked list should be maintained in sorted order. If you like, you can use a doubly-linked list instead of a singly-linked list.

You will need to change most of the code in `intset.cpp`, and some of the code in `intset.h`. However, you should not change the definition of any of the public member functions (e.g., `find`)

in `intset.h`, so that code that uses your integer set class should not need to be changed in any way (this is one of the benefits of object-oriented software design – the ability to change underlying implementation of a class while keeping the same public interface intact, making the change largely transparent to anyone using the class).

2 Compiling and Running

To compile on the Unix lab machines, use the command:

```
g++ -o main main.cpp intset.cpp
```

You can then execute your code by running:

```
./main
```

Recall that the "-o main" part of the command line tells the compiler what to name your executable file, in case you want to give it a different name.

3 Testing

The file `main.cpp` provides a simple command-line interface for testing your code. It allows you to type, at the command line, commands like:

```
insert 7
remove 3
find 2
print
quit
```

These commands invoke the corresponding methods from the integer set class, so you can use them to debug your work. You may also want to put several of these commands in a text file (e.g., `input.txt`) and execute them all at once using

```
./main < input.txt
```

since this way you won't need to re-type the same commands over and over during testing.

4 Submission

To submit your work, please upload your modified versions of `intset.h` and `intset.cpp` on `handin.cs.clemson.edu`, where we will be turning in all of our work this semester. You should submit your work under the main CPSC 2120-001 course (not the course number of the lab section you are attending).

Final submissions are due by 11:59pm on the evening of Friday, August 29, just in case students have a small amount of extra work to finish after the end of lab. No late submissions will be accepted.

5 Grading

All labs are graded on a scale of zero to ten points, where the point breakdown depends somewhat on the particulars of the lab. For this lab, you will receive 10 points for code that is fully correct, and varying amounts of partial credit for incorrect code. Zero points will be awarded for code that does not compile, so make sure your code compiles on the lab machines before submitting!