CpSc 2120 — Goddard — Spring'18

Assignment 2

Due 8pm Tuesday 13 March

EFFICIENT RANGE-SET STORAGE. You are to produce a class RangeSet that maintains a ***set of nonnegative integers***. The storage and display of the set should be in terms of ***ranges***—a ***range*** is a group of consecutive integers.

Task 1

The assignment is to produce a class RangeSet that maintains a set of integers, with functions to update and display the set in terms of ranges. The data must be stored in a linked list using struct RNode. A skeleton for this is provided.

The linked list must be stored in increasing order with one node per range. (I found it easier to have a dummy node, but it's up to you.)

Your class should initially have the following three public functions:

- Default constructor
- The mutator void addLonelyRange(int min, int max): This adds all integers from min to max inclusive to the set. If the range is empty or contains a negative, it should print a suitable error message and not change the list. Important: This function may assume that the added range does not overlap nor extend any existing ranges; and it does not have to do any error-checking thereof.
- The accessor void dump( ): this prints out the set as a series of ranges. If there is only one integer in a range, only one number should be displayed. An endl should be printed.

You may add private helper functions as needed. For example,

```
RangeSet S;
S.addLonelyRange(22,33);
S.addLonelyRange(1,1);
S.addLonelyRange(77,78);
S.dump( )
```

should produce

```
1,22-33,77-78,
```

Task 2

Add the following public functions to your class:

- The accessor bool isInSet(int val): this returns whether the value val is in the set or not
- An equality tester (overloaded ==) to compare two RangeSet's
- A suitable destructor

Task 3

(Challenging!)

Add the following two public functions to your class:

- bool deleteValue(int val): This deletes the integer val from the set. For example, if the set starts with just the range 1:100 and we delete 14, then the linked list must now have two nodes, one with the range 1:13 and the other with the rnage 15:100. It returns whether val was found or not.
- void addRange(int min, int max): This adds the integers from min to max inclusive to the set. Unlike the earlier add command, this may overlap or extend existing ranges. This function should leave the linked list in minimal form; that is, with the fewest possible ranges. For example, if the set starts with 1:3, 5:7 and 11:23 and we add 4 thru 9, the linked list must now have two nodes, one with 1:9 and the other with 11:23. Again print suitable error message if range is empty or contains a negative.

Comment: I found it a little easier to change RNode so that I had a doubly linked list. Further I also created a little private routine that removed a specified node from the list.

Other Instructions

You may NOT use any data structure from the C++ Standard Template Library or other such Data Structures package.

You should write (but not submit) a test program that vigorously tests your class.

You are to work independently, but can ask questions from the lecturer and lab instructor(s). Late submissions will be significantly penalized.

Submit your RangeSet and RNode files using handin.