CS 3363, Organization of Programming Languages: Programming Assignment 1

Assignment name: ML insertion sort.

Due: Saturday, February 11, 2023, at 11:59 PM, U.S. Central Time.

Write a file named '**p1.sml**' that contains a function named '**rinsort**', which accepts two parameters: a **real list** and **comparator function**. It must return the sorted **list** of type **real** using an insertion sort, in the order indicated by the comparator function. The comparator functions to be supported are **op<**, **op<=**, **op>=** and **op>**. If the **op<** or **op<=** operator is passed, the list is to be sorted in ascending order (low to high), while the other two operators indicate that the list is to be sorted in descending order. Your solution must be recursive.

There may be from zero to an arbitrarily large number of **real** values in the input **list** parameter, and any value in the **list** may appear any number of times up to the number of entries in the **list**. Your solution must use recursion for both the sort and the insertion. If a value appears more than once in the input list it must appear only once in the output list.

Your **rinsort** function must use a separate function to perform insertions. You must name this helper function '**rinsert**'. *You may not embed this function within the* **rinsort** *function*. The **rinsert** function must accept three parameters: a **real** value to be inserted into the second, a **real list** parameter and a comparison operator. It must return a properly sorted **real list** in the order indicated above for the passed comparator. (Note: Separation of the two functions allows independent testing of **rinsort** and **rinsert**.) The **rinsert** function must be recursive.

Remember that an empty **list** and a **list** of one element are both definitionally sorted. Correct insertion of an item into a sorted **list** produces a sorted **list**.

The way your **rinsort** function should work is that if it is passed an empty **list** it will return that **list**. If the **list** is non-empty, it should call **rinsert** to insert the head of the list into the list returned by a call to **rinsort** using the tail of the **list** passed as its parameter.

Again, both **rinsort** and **rinsert** functions must be recursive.

Code that does not compile and run will receive no credit.

You must document your code. Use the file heading requirements from the syllabus, as well as providing an documentation of how and why your code works within the two functions.

Submit the file '**p1.sml**' to the **handin** folder for this class on the department's csx server using the following command on csx.

'**handin cs3363-rlchurc program1 p1.sml**'

Programs that do not include proper documentation, including both header and in-line comments, will be penalized up to 20 percent. If the csx server is not accessible when you try to submit your solution, submit to Canvas, but resubmit to csx as soon as you are able.