CS 135 Data Structures
Assign 1

Milestone: Wed Mar 27

ACTIVITIES

1. Create a new class that implements the **SortedList** ADT that supports the following functionality:

   - **void putItem(T item)**
     inserts an item into the list and the list remains sorted
     (i.e. this function finds the proper location in the list to always keep the list in order)
   - **T getItem(int index)**
     returns the item found at a provided index
   - **T pop()**
     remove and returns the last item in the list
   - **T pop(int index)**
     remove and return the item found at index
   - **void remove (T item)**
     removes the first instance of item from the list;
   - **bool contains()**
     returns true when item is in the list; false otherwise.
   - **void getCount(T item)**
     returns the size of the list.
   - **void printList()**
     Displays (cout) the contents of the list.

   Create a simple driver function that demonstrates this functionality
   (including any special cases that need to be considered)

   Notes:

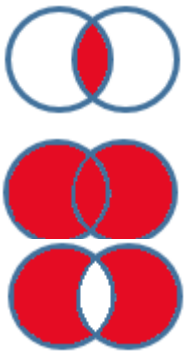
   a. Your implementation should use a linked list.

   b. With the exception of putItem, this implementation is *essentially* the same as the List ADT that you have already coded. I would suggest that you re-implement these functions instead of copy-pasting them from the previous task. a) The extra practice will do you good! b) You may discover some savings! (hint consider the need for "end").

   c. Your implementation should use a templated class. This will allow the SortedList to store multiple types. However, you need to be mindful of a specific dependency: when you put an item into the SortedList, it must be able to compare the item to other entries. Therefore, T is restricted to types that work with the relational operators <, <=, ==, >, >=, != So you should expect a SortedList to immediately work with int and string. You can also make a SortedList to hold user-defined types (i.e. PlayingCards), but you would need to take some extra steps to tell C++ how to use the relational operators with that type. (We won't do that…. yet).

2. Create new class that implements a **Set** ADT that supports the following functionality:

   - **void putItem(T item)**
     inserts an item into the Set; ensures there are no duplicates.
     (i.e. this function finds the proper location in the list to always keep the list in order)
   - **void remove(T item)**
     removes an item from the Set (if it exists)
   - **void intersection (Set other)**
     prints* the Set that is the intersection between this Set and the other Set
   - **void union (Set other)**
     prints the Set that is the union between this Set and the other Set
   - **void difference(Set other)**
     prints the Set that is the difference between this Set and the other Set
   - **void getCount()**
     returns the size of the set.
   - **void printSet()**
     Displays (cout) the contents of the Set.



   Create a simple driver function that demonstrates this functionality
   (including any special cases that need to be considered)

   Notes:

   a. Start by Copy-Pasting the code from SortedList into new files (Set.h and Set.cpp).

   b. Your implementations should be very similar to the previous version. Pay attention what needs to change.

   c. All of your implementations should be at least O(n) efficient.

* We really want to return instead of print, but that can wait….