CISC 2200

Lab3

We have studied ADT list and an example implementation using a fixed size array. In this lab, you will extend and improve the implementation (unsorted.h, unsorted.cpp), and then use the list to write a program that manages appointments (enter, delete, sort, search for appointments).

To complete the assignment:

1. First modify UnsortedType based upon the following detailed instructions:

• Modify the UnsortedType class to use a dynamically allocated array to store the items, you need to add an int field to remember the size of the array.
• Modify zero-parameter constructor so that it allocates an array of size 10 (default size)
• Add a constructor that takes an int as parameter, specifying the size of array to allocate
• Add a destructor for the class (why do you need to do so?)
• Modify IsFull( ):

It will return true, if and only if, the array is filled and an attempt to allocate larger array fails. This means that if array if not full, or if array is full but we could allocate a larger array, then the method return false.

• Overload index operator (i.e., []) to return the reference of i-th element in the list
• Modify PutItem( ) so that it can be called to put an item into a "full" list:

1. Allocate a larger array (which can be double of current size)
2. Copy current array elements to the new array
3. Delete current array
4. insert item into new array
5. set info points to new array

2. write a class named **Appointment** that has the following member fields:

• Year, Month, Day (You could use the DateType from chapter 2.) The date of the appointment will be used as key to compare two appointments.
• a string description of the appointment

Write a few member functions that allow one to input, output, and compare two appointments' key.

To use Appointment as ItemType, you can do it in either of the following two ways:

• Wrap Appointment inside an ItemType class
• Give the Appointment class an alias, ItemType
typedef Appointment ItemType;

Either way, you need to make sure that ItemType class implements CompareTo( ) method, as required by UnsortedType. This method compares two items' key value, i.e., two appointment objects' date.

3. Write the **main** function that runs in a loop that allows the user to choose a command:

- Display: display all appointments
- Add: add a new appointment:
  if there is already an appointment on the date, reports error.
- Search: search for an appointment on a user specified date
- Delete: delete an appointment on a user specified date
- Quit: to quite the program

When a command is chosen, the program shall prompt the user to enter relevant info, for example, to add a new appointment, the program prompts the user to enter the date and the description, and then create an Appointment object, call PutItem() to add the item into the list.