Workflow

Name:	03.02
Points:	3 pts
Deadline:	03/08
Prerequisite(s):	none

Main

1. Create a header file named **w0302.h** that defines a generic class named *StaticMultiset* that inherits the interface *SetInterface*. It will represent a multiset with a finite capacity, where a multiset is an unordered collection of objects. The class must contain

   - □ a private generic *Array* field named *data*.
   - □ a private int field named *size*.
   - □ its special member functions that should be public and the default size of *data* is 100 and the default value of *size* is 0.
   - □ a public overloaded constructor that takes an int parameter. If the parameter is at least 2, the function makes the parameter the size of *data*; otherwise, it makes 100 the size of *data*. Furthermore, it assigns 0 to *size*.
   - □ a public overridden `Insert()` method. It adds the parameter to the multiset if the set is not at capacity.
   - □ a public overridden `Remove()` method. It removes an instance of the parameter from the multiset if the multiset is not empty.
   - □ a public overridden `IsEmpty()` method. It returns true if the multiset does not have any members; otherwise, it returns false.
   - □ a public overridden `Contains()` method. It returns true if the parameter is in the multiset; otherwise, it returns false.
   - □ a public bool constant method named `IsFull()` that takes no parameters. It returns true if the multiset is at capacity; otherwise, it returns false.
   - □ a public int constant method named `Count()` that takes a constant generic reference parameter. It returns the number of occurrences of the parameter in the multiset.
   - □ a public string constant method named `ToString()` that takes no parameters. It returns a string of the elements of the multiset all enclosed in curly braces with each element separated by a comma.

Test

2. Create a cpp file named **main.cpp** that creates a *StaticMultiset* object and test each of its methods. The outputs of the methods that are not void must be displayed.