Lab 02 - Classes & Generics

Instructions:

- **The lab requires completing a few tasks.**
- **Your submissions must be submitted to the GitHub repository in your Lab02 directory.**
- **You need to use the classes *Mask* and *Object* from the header files 'Mask.h' and 'Util.h'. Its documentation is provided in the directory.**
- **You can only use the libraries *iostream*, *string*, *sstream*, *cctype*, *cmath*, *stdexcept*, *iomanip*, and the header file 'Util.h'.**
- **Cheating of any kind is prohibited and will not be tolerated.**
- **Violating or failing to follow any of the rules will result in an automatic zero (0) for the lab.**

TO ACKNOWLEDGE THAT YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS ABOVE, AT THE BEGINNING OF YOUR SUBMISSION(S), ADD A COMMENT THAT CONSISTS OF YOUR NAME AND THE DATE. PROVIDE THE NAME OF YOUR TEAMMATE IF YOU ARE WORKING IN A GROUP.

Grading

Task	Maximum Points	Points Earned
1	3	
2	2	
Total	5	

Note: solutions will be provided for tasks colored blue only.

Task 1

Create a header file named '`DekaTuple.h`' that defines the generic class *DekaTuple* that publicly inherits *Object* and contains

- ☐ A private generic array field named *data* of size 10.
- ☐ A public default constructor that assigns the default generic value to each element of *data*.
- ☐ A public copy constructor.
- ☐ A public overloaded assignment operator.
- ☐ A public empty destructor.
- ☐ A public generic type reference method named `GetValue()` that takes an int parameter. It returns the element of *data* whose index equals the parameter if the parameter is between 0 and 9, inclusively; otherwise, it throws the error message `"out of bound"`.
- ☐ A public constant generic type reference constant method named `GetValue()` that takes an int parameter. It returns the element of *data* whose index equals the parameter if the parameter is between 0 and 9, inclusively; otherwise, it throws the error message `"out of bound"`.
- ☐ A public bool constant method named `Contains()` that takes a constant generic type reference parameter. It returns true only if an element of *data* is equal to the parameter; otherwise, it returns false.
- ☐ A public int constant method named `Count()` that takes a constant generic type reference parameter. It returns the number of times the parameter appears in *data*.
- ☐ A public overridden method `ToString()` from *Object* that returns a string of a list of the elements of *data* separated by a comma all enclosed in parentheses.

Hint: Make *data* an *Array* object.

Task 2

Create a cpp file named '`main.cpp`' that

- ☐ Includes the header files `"Mask.h"` and `"DekaTuple.h"`.
- ☐ Defines a void function named `RemoveDuplicates()` that takes a *Mask DekaTuple* reference parameter. It voids elements of the parameter until there are no duplicates.

  Example: If

  ```
  data = (2,5,2,4,7,8,1,7,3,6)
  ```

  then after the invocation `RemoveDuplicates(data)`, `data` will possibly be `(0,5,2,4,0,8,1,7,3,6)` where zero represents a voided value.
- ☐ Defines its main function that
  1. Starts with the statement `srand(time(nullptr));`
  2. Declares a *Mask DekaTuple* object.
  3. Invokes `RemoveDuplicates()` with the object as its argument.
  4. Displays the object.

Etra Credit

Create a header file named '`Extra.h`' that defines the class *LetterSet* that publicly inherits *Object* and contains

- ☐ A private bool array field named *data* with a size of 26.
- ☐ A public default constructor that assigns false to each element of *data*.
- ☐ A public copy constructor.
- ☐ A public overloaded assignment operator.
- ☐ A public empty destructor.
- ☐ A public bool method named `Insert()` that takes a char parameter. It assigns true to the element whose position is the same as the parameter's position in the alphabet and returns true if the parameter is a letter; otherwise, it returns false.
- ☐ A public bool method named `Remove()` that takes a char parameter. If the parameter is a letter and the element whose position is the same as the parameter's position in the alphabet is true, it assigns false to the corresponding element and returns true; otherwise, it returns false.
- ☐ A public bool constant method named `Contains()` that takes a char parameter. It returns the element whose position is the same as the parameter's position in the alphabet if the parameter is a letter; otherwise, it returns false.
- ☐ A public bool constant method named `IsEmpty()` that takes no parameters and returns true only if all elements of *data* are equal to false; otherwise, it returns false.
- ☐ A public bool constant method named `IsFull()` that takes no parameters and returns true only if all elements of *data* are equal to true; otherwise, it returns false.
- ☐ A public short constant method named `Count()` that takes no parameters and returns the number of elements of *data* that are equal to true.
- ☐ A public void method named `Clear()` that takes no parameters and assigns false to each element of *data*.
- ☐ A public overridden method `ToString()` from *Object* that returns a string of uppercase letters separated by a space that corresponds to the elements of *data* that are true.