

Gebze Technical University
Department of Computer Engineering
CSE 241/505
Object Oriented Programming
Fall 2021
Homework # 6
Inheritance, Templates, STL
Due date
Dec 30th 2021

In this homework, you will write a templated class hierarchy for a simple container class hierarchy.

The class **Iterable<T>** is an abstract class with the following pure virtual member functions.

<u>empty</u>	Test whether container is empty
<u>size</u>	Return container size
<u>erase</u>	Erase element pointed by the given iterator
<u>clear</u>	Clear all content
<u>begin</u>	Return iterator to beginning
<u>end</u>	Return iterator to end
<u>cbegin</u>	Return a constant iterator to beginning
<u>cend</u>	Return a constant iterator to end

The class **GTUVector<T>** derives from the base class and implements all of the functions appropriately for a vector class including functions for add and operator[].

The class **GTUSet<T>** derives from the base class and implements all of the functions appropriately for a set class including functions for search, add, delete, intersect, union.

The class **GTUArray<T, SIZE>** derives from the base class and implements all of the functions appropriately for a fixed size array class such as STL array.

Here are some common features of the classes

- All classes will keep their data using dynamic memory techniques with shared_ptr STL pointers. Do not use regular pointers or STL container classes as data members.
- All classes should implement move semantics functions.

- There should be a class hierarchy for the iterators. The classes should use derived class iterators from a base iterator class. The classes **GTUIterator** and **GTUIteratorConst** implement iterator operators such as *****, **->**, **++**, **--**, **=**, and **==**.
- Your classes should be compatible with range based for loop, **std::find**, **std::for_each** and **std::sort** for array and vector.

You will also implement the following global functions from the STL library which will accept

Write your driver program to test the all the classes and all of their functions. Do not forget to test the compatibility with all std algorithms.

Notes

- Use separate header and implementation files for each class.
- Use name spaces.
- Do not forget to define and test exceptions.