CO3402 Object-Oriented Methods Assignment 2018-2019

Date issued:29/10/2018 **Date due:** 02**/**12/2018 (Midnight 11:59:59 PM)

Introduction

In the practical classes, you should have developed a template class implementing the stack abstract data type (ADT) using a linked list. To show you understand the concepts you have learnt from this, you will design a map ADT and implement it using a different representation.

A map is an associative container, linking a "key" to a "value" with no repetition allowed, which means that each key is associated with a single value. A map is like a look-up table, when you look up the key you get the corresponding value.

Key 1	Value 1
Key 2	Value 2
…	…
Key N	Value N

Table 1: How data are arranged in a map

A map might be used to represent a phone book, allowing you to find the phone number for a name. A map representing a phone book would allow you to insert and remove words, and to change definitions. You must decide the other operations that the map should provide. You may find ideas for the interface on the Internet and from the STL, but ensure that the interface is consistent with the abstract data type concept of a lookup table. Pay attention to the criteria for designing an interface and other relevant issues considered in the lecture notes.

Task Details

Note: for guidelines what to hand in and what the marker will be looking for, see the marking scheme.

Working INDIVIDUALLY:

1. Write a C++ class declaration (method headers & properties) for a map **abstract data type (ADT)**. The declaration should contain appropriate comments to explain the purpose of the methods and the properties. The STL documentation may give you ideas for the comments, but you must ensure that your comments are clear. Your class should satisfy the criteria for an abstract data type (for example, it should not define unnecessary or inappropriate methods). Your class should be as robust and flexible as possible so it can be used safely in different situations. Do **not** assume that any of the methods of any STL class are necessarily appropriate.

2. Implement the map class in C++ (i.e. write the method bodies). **The data structure to represent the map must use an array of elements allocated when the map is constructed. You are not supposed to use any existing facilities from STL(No stl::vector or similar facility should be used). This means all the Map implementation should be done by your code.** This is not necessarily the best representation to use, but that will give you something to discuss in your evaluation. Ideally, the map will not have a fixed upper size. Make sure your implementation behaves properly– check through the lecture notes for ideas about proper behaviour.

3. You map class should support **as many data types as possible**. **NOTE, both key and value in the map have data types.**

4. A test program that test All the operations of your map class with ALL the data types that you map supports.
5. Write a brief (1-2 pages) evaluation report of your map class: justify any design decisions (e.g. its interface, the use of friends) and evaluate the support provided by C++ for developing abstract data types and the use of this support for your map class. Identify any weaknesses in your implementation and discuss improvements and alternative representations, and compare your map class with the map class provided in the STL

What You Should Submit

You should submit a Zip file that contains the full visual studio project plus the evaluation reports via the links on Blackboard. This file should follow the name convention: YourStudentNumber.zip. **Using the default file name or random file name for the submitted file may lead to a severe delay for your work to get marked! Please also note, in the submitted project, DON'T expose your name anywhere.**

To allow for late submission (up to 1 week late for a maximum of 50%) work will be accepted up to 09/12/2018. However, late penalties will be applied to work submitted after 02/12/2018.

In the practical sessions in the weeks commencing from 3th, Dec 2018 and 10th, Dec 2018, you will need to show how your program works and be ready to answer questions from me.

Marking Scheme

Student G Number:		
Characteristics	**Mark Range**	**Marks and Comments to be used by Markers**
Very limited implementation with less than 4 methods or containing major bugs. No support for multiple data types. No support for dynamic data size. Simple Comments Simple test program	0-25	
Provides essential map facilities for more than one data type. No dynamic size support. No major bugs. Sensible comments & layout. Simple test program.	26-50	
Support more all system provided primitive data types or data types like string. Support Dynamic data size using array only Memory safe Sensible comments & layout. A test program that covers all methods and data types supported.	50-70	
Has a mechanism for supporting all data types (e.g. user defined classes) Support dynamic data size using array only Iterators are supported by the map class. Memory safe Sensible comments & layout. A test program that covers all methods and data types supported.	70-90	
Evaluation Report	10 Max	