Project 5:

USE OF TEMPLATES

OBJECTIVE

- Learn to create a complete template class.
- Modify the template class design for augmenting its flexibility.

TO BE HANDED OVER

- table_t.h, test_tabla.cpp
- makefile (without parameters should compile all programs)

PREVIOUS EXERCISES

- Write the *test_tabla.cpp* for section 1.

DEVELOPMENT

In this project we are going to transform a normal class in a template class. We will also practice with some of the concepts seen in theory classes about templates.

1. Test program

First you should write a program (test_tabla.cpp) for testing the *Tabla* class, which implements a hash table with chaining. The code for this class can be found in Aula Virtual. The program has to do the following steps:

1. It has to create a hash table of the adequate size for storing 200 *alumnos*, it has to generate the 200 *alumnos* and insert them in the table.
2. It has to look for one key that exists in the table and another key that does not exist. The program has to show the *alumno* or show a message "Student does not exist".

2. Conversion to template class

The Tabla class has to be converted to a template class, so a user will be able to use it with any type of key or value (do not worry about the hash function in this section). As the class is now a template, all the code should be included in the header file (table_t.h). Test the class with the previous program.

Store a copy of the Tabla class for project 6.

3. Collision list

In the actual version, we are using a vector for the *collision list*. Modify the class to allow the selection of the container to be used as a *collision list* using a *template template parameter* (see Ch5.4 of "C++ Templates: The Complete Guide"). The class should admit at least, a vector and a list. Indicate with a program comment which operations should be supported by the container. Test the program with other types of containers.

4. Hash function

In this section, you will define hash functions for the most common types (*int* and *string*). Modify the design of the class so that the functions are selected automatically.

Sometimes, an application needs a particular hash function because the general one does not work well. The design of the class should also allow modifying the hash functions from the main file, so we can use a different hash function if needed.

Test all modifications of the class in the main program.