Revisão: Introdução à Programação Genérica

Prof. Tiago G. S. Carneiro

A programação genérica é uma forma de metaprogramação. Isto é, uma maneira de desenvolver códigos que irão manipular ou desenvolver outros códigos ou a si próprio. Na prática, a programação genérica permite que um código seja adaptado (customizado) e re-escrito automaticamente em tempo de compilação. Desta forma, evita-se que o programador precise reescrever manualmente trechos de código somente porque o tipo de seus argumentos mudou. Na linguagem C++, o conceito de *template* permite que as implementações de estruturas de dados independam dos tipos de objetos armazenados nestas estruturas.

O conceito de *iterator* pode fazer com que algoritmos independam da estrutura de dados que percorrem. O algoritmo *sort* da STL (Standad Library) é capaz de ordenar qualquer estrutura de dados. Veja como é sua assinatura: *sort(estDados.begin(), estDados.end())*. Para que ele funcione, o programador apenas deve informar o ínicio e o fim dos dados. Contudo, é essencial que a estrutura de dados a ser ordenada possua um iterador que permita o algoritmo *sort* percorre-la.

Portanto, a programação genérica promove o REUSO DE CÓDIGO ao permitir que implementações de estruturas de dados independam do tipos que armazenam e que algoritmos independam das estruturas de dados que percorrem. Ao evitar o trabalho manual, a programação genérica faz com que os programadores sejam mais produtivos.

Para responder as questões abaixo, estude os códigos “hello.h” e “hello.cpp” fornecidos com esta lista.

Questões:

1. Implemente as classes estudantes e disciplina, invente um conjunto mínimo de atributos com seus métodos *gets* and *sets*.

2. Utilize o conceito de *template* para implementar duas outras classes que poderão ser utilizadas para representar conjuntos de qualquer destes dois objetos. A primeira classe deve armazenar seus elementos de forma não ordenada. A segunda deve armazenar seus elementos de forma ordenada. Por exemplo, ConjNaoOrdenado<Estudante> e ConjNaoOrdenado<Disciplina> ou ConjOrdenado<int,Estudante> e ConjOrdenado<int, Disciplina>.

3. Implemente algoritmos *count, sum e average que* possam ser utilizado para calcular estatísticas sobre quaisquer dos conjuntos implementados na questão anterior.

4. Forneça códigos de testes que garantam a qualidade das implementações das questões anteriores. Utilize o método *assert* da STL nas implementações de todos seus testes.

BOM TRABALHO!