UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CIÊNCIA DA COMPUTAÇÃO

FELIPE ARCHANJO DA CUNHA MENDES

RESENHA CAPÍTULO 5
LIVRO PROGRAMAÇÃO ORIENTADA A OBJETOS COM C++

CAMPO MOURÃO
2021

Resenha capítulo 5

O capítulo 5 inicia com um questionamento um tanto quanto interessante:

“Por que software não é como hardware? Por que todo desenvolvimento começa do nada? Deviam existir catálogos de módulos de software, assim como existem catálogos de chips: quando nós construímos um novo sistema, nós deveríamos estar usando os componentes destes catálogos e combinando-os, em vez de sempre reinventar a roda. Nós escreveríamos menos software, e talvez faríamos um desenvolvimento melhor . Será que alguns problemas dos quais todo mundo reclama - custos altos, prazos insuficientes, pouca confiabilidade - não desapareceriam? Por que não é assim?”

Neste sentido o livro questiona a necessidade de produzir partes de código em massa, assim como ocorre com peças de hardware, exaltando as vantagens de se reutilizar códigos. A partir daí explica como deve ser a forma de se modularizar códigos da forma correta para não haver erros.

Nos foi mostrado que um módulo deve ser feito de tal forma que ele se adapta ao estilo de código e a estrutura de dados por trás dele. Não só isso, mas também foi explicado que esses módulos devem ter o mínimo de repetições possíveis, uma vez que isso pode afetar o desempenho do seu código. Por fim foi dito que o método de instalação deste código, pelo usuário, deve ser feito da forma mais simples possível uma vez que o usuário não teria a obrigação de entender como funciona detalhadamente o algoritmo por trás do desenvolvimento do programa.

A partir daí foi explicado o conceito de heranças em C++. De acordo com o livro, as heranças são uma forma de modularização pois se trata de um recurso que você reutiliza classes base em novas subclasses filhas.

A estrutura de um herança se dá como segue:

```
class Caixa {
        public:
                int altura, largura;
                void Altura(int a) { altura=a; }
                void Largura(int l) { largura=l; }
};

class CaixaColorida : public Caixa {
        public: int cor;
        void Cor(int c) { cor=c; }
 };
```

Perceba, neste exemplo, que a classe Caixa é a chamada classe base ou classe pai. Nela foram criados alguns métodos e atributos. Posteriormente foi criada uma outra

classe, porém com a sintaxe de classe filha, uma vez que tal sintaxe está relacionada ao fato de a classe CaixaColorida herdar os atributos e métodos da classe Caixa.

Neste sentido, podemos utilizar a classe Caixa Colorida da seguinte maneira:

```
void main() {
        CaixaColorida cc;
        cc.Cor(5);
        cc.Largura(3); // herdada
        cc.Altura(50); // herdada
}
```

Perceba que a forma de instanciar uma classe filha é da mesma forma que herdar uma classe base comum.

Portanto, nota-se que além de funções e variáveis, uma boa forma de economizar e reutilizar código escrito nos algoritmos, a utilização de heranças em C++ é uma forma bastante útil para alcançar esse objetivo, indo de frente com a reflexão proposta no início do capítulo.