Júlia Feliciano/E1000021989
Faculdade de Engenharia
Universidade Católica de Angola
juliapreeciosa18@gmail.com

No presente trabalho será feito um resumo geral dos fundamentos de **POO**. Será destacado os assuntos como: Abstração, Encapsulamento, Herança, Polimorfismo e Interfaces.

Palavras chaves: *abstração, encapsulamento, herança, polimorfismo, interfaces, classes, métodos, objectos.*

I. Introdução

A **Orientação a Objetos** é uma das várias formas de se programar. Muito usado para diminuir a complexidade do código, a raciocinar sobre o problema a ser resolvido e também no que diz respeito a reutilização.

II. Conceitos

A **abstração** é um processo de abstrair algo do mundo real e transformá-lo em objeto na programação com suas características e funcionalidades. A abstração reduz a complexidade do código e, ao mesmo tempo torna a sua estética agradável. Consiste em um dos pontos mais importantes dentro de qualquer linguagem Orientada a Objetos. São três pontos que devem ser levados em consideração nessa abstração: 1- uma ***identidade ou entidade*** ao objeto que iremos criar. 2- diz respeito as **características** ou ***propriedades*** do objecto. 3- definirmos as ***ações ou métodos*** que o objeto irá executar.

Outro conceito muito importante da POO é o **encapsulamento** das informações. Este permite que os atributos de uma determinada classe somente sejam modificados utilizando métodos que interajam com o mesmo. Assim, as modificações através dos métodos garantem que não há manipulações indevidas aos atributos. Basea-se em propriedades privadas, ligadas a métodos especiais chamados *getters* e *setters*, que irão retornar e setar o valor da propriedade, respectivamente. Essa atitude evita o acesso direto a propriedade do objeto, adicionando uma outra camada de segurança à aplicação.

O reuso de código é uma das grandes vantagens da POO. Muito disso se dá por uma questão que é conhecida como ***herança*****.** Essa característica otimiza a produção da aplicação em tempo e linhas de código. A herança é um mecanismo que permite que uma classe possa herdar o comportamento de outra classe, ao mesmo tempo em que novos comportamentos podem ser estabelecidos. A vantagem da herança é agrupar coisas comuns para poder reaproveitar código. As sub-classes (classes filhas) herdam da super-classe (classe pai). Ou seja, é a criação de uma nova classe a partir de uma classe existente.

Outro ponto essencial na programação orientada a objetos é o chamado **polimorfismo.** Como sabemos, os objetos filhos herdam as características e ações de seus "ancestrais". Entretanto, em alguns casos, é necessário que as ações para um mesmo método seja diferente. Em outras palavras, o *polimorfismo* consiste na alteração do funcionamento interno de um método herdado de um objeto pai. O polimorfismo está intimamente conectado à herança. Pode-se definir duas abordagens como: sobreposição(override) e sobrecarga(overload). A sobreposição ocorre quando duas ou mais classes derivadas de uma mesma superclasse podem invocar métodos que possuem a mesma identificação(assinatura), mas comportamentos distintos. Enquanto que a sobrecarga dos métodos ocorre quando em uma mesma classe, métodos possuem os mesmos nomes, mas assinaturas diferentes.

As **interfaces**, outro conceito importante e bastante utilizado são invólucros que promovem a interação de contatos externos, com ou sem passagem de dados, com um processamento interno. As interfaces são especificações ou contratos. São responsáveis por descrever as assinaturas dos métodos e não como eles serão implementados. Além disso, a utilização das interfaces permite que simulemos a herança múltipla, pois uma classe pode implementar uma ou mais interfaces. Assim, uma interface é a reunião das assinaturas dos métodos, sem definições reais. Isto indica que uma classe tem um conjunto de comportamentos (vindo da interface), além dos que recebe de sua superclasse.

Referências bibliográficas

1-Claro D. B. e Sobral J. B. M. PROGRAMAÇÃO EM JAVA. Copyleft Pearson Education. Florianópolis, SC.

2- Programação Orientada a Objetos Flávio de Oliveira Silva

Pilares da POO (Programação Orientada a Objectos)