**Resumo dos conceitos/princípios em OOP**

António Nunes Mbala/ 1000022837  
Faculdade de Engenharia   
Universidade Católica de Angola  
Antonionunes988@gmail.com

***Abstrac***t – este tema tem

Como objetivo tornar os software mais flexível tanto na modificação como na implementação que possibilitam o surgimento de várias classes por intermédio de uma já existente.

1. INTRODUÇÃO

Os seres humanos utilizam mecanismos para pensar e ver o mundo são totalmente orientados e flexível. Para representar um software devemos utilizar uma notação que seja capaz de ajudar programador a decifrar o que deve ser juntamente com a linguagem com a linguagem de programação para que essa notação seja compreendida pelos analistas, responsáveis por identificar e modelar os requisitos do que será entregue.

1. CONSEITOS

**Encapsulamento –** Encapsulamento vem da palavra encapsular, que em programação orientada-se a objetos,e significa separar o programa em partes.

O encapsulamento tem como objetivo tornar o software mais flexível, ou seja fácil de modificar e de criar novas implementações, é uma forma eficiente de proteger os dados manipulados dentro da classe, além de determinar onde esta classe poderá ser manipulada.

Para ter acesso a algum métodos que estejam encapsulado usa-se o conceito de get e set.

SET: é feita uma atribuição a algum atributo, que determina ou diz o valor que algum atributo deve ter.

GET: permite recuperar os valor perdidos pelos atributos

**Herança –** A herança é a parte que se encarrega pela orientação a objetos que permitem criar novas classes a partir de uma classe já existente. Este mecanismo é muito importante porque promove o reuso e o reaproveitamento de código existente.

**Abstração -** É a definição de entidades do mundo real, e é onde são criadas criadas as classes. Essas entidades são consideradas tudo que é real, tendo como consideração as suas características e ações.

**Polimorfismo –** Em Poucas palavras Polimorfismo é uma técnica ou método que nos permite agregar ou utilizar à programação o uso de boas práticas, já testadas e aprovadas, certamente que vão tornar seu código mais robusto e flexível. Uma boa implementação deve não somente atingir seus objetivos de funcionamento , mas também ser de fácil aprimoramento e compreensão.