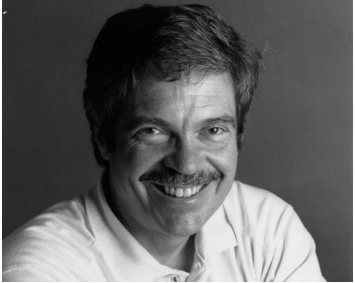


HISTÓRIA DA ORIENTAÇÃO Á OBJETOS		HISTÓRIA DA ORIENTAÇÃO Á OBJETOS
Como os paradigmas da programação <b>mudaram</b> com o passar do tempo até chegar na POO?		<p><b>1º Programação de Baixo Nível (1950):</b> Feita somente pelos engenheiros que entendiam do baixo nível específico para determinado tipo de computador;</p> <p><b>2º Programação Linear:</b> Programação de médio e alto nível, onde a programação seguia o código de cima para baixo sem desvios;</p> <p><b>3º Programação Estruturada:</b> Programas de médio e alto nível que permitiam desvios de fluxo no código fonte para as funções;</p> <p><b>4º Programação Modular:</b> Programas de alto nível que encapsulavam módulos que poderiam ser referenciados;</p> <p><b>5º Programação POO:</b> Veio em substituição rápida da programação modular;</p>
O que significa a <b>POO</b> ?		<p><b>POO:</b> significa Programação Orientada a Objetos, em inglês é OOP (Object Oriented Programming).</p>
Quem é considerado o <b>pai da Linguagem POO</b> ?		<p>O homem considerado o "Pai da POO" é <b>Alan Kay</b>, um Biólogo e Matemático que desenvolveu a POO implementando conceitos da área de matemática e biologia para criar a POO, nas palavras do próprio Alan Kay: "O computador ideal deve funcionar como um organismo vivo, isto é, cada célula deve funcionar a fim de desempenhar um objetivo, mas elas também devem reagrupar-se a fim de desempenhar outras funções a fim de resolver um problema". Estudando o funcionamento da POO vamos perceber que isso tem tudo a ver com o que Alan Key quiz dizer.</p>
- reservada para a questão acima -		
Qual é o <b>conceito fundamental</b> da POO?		<p>A POO parte do conceito de que "<b>qualquer objeto da vida real pode ser explicado e recriado por através da Linguagem Orientada a Objetos</b>". Por exemplo um controle remoto, apesar do controle remoto ter vários componentes que o completam, cada deles desempenha sua função individual, para gerar o "controle remoto". A Linguagem Orientada a Objeto parte desse mesmo princípio, de que podemos encapsular elementos que desempenha sua função individual, assim como as células, e que podemos reagrupar esses elementos para gerar resultados e objetos diferentes, criando várias aplicações diferentes a partir de métodos já existentes e até desenvolver sistemas inteiros.</p>
Qual foi a <b>primeira linguagem de programação</b> á usar a POO?		<p>A primeira linguagem de programação a usar a POO foi a "<b>SmallTalk</b>", desenvolvida pelo próprio Alan Kay e sua equipe de desenvolvimento. Apesar de ser a primeira, o "SmallTalk" já contava com muitas ferramentas da POO atual, como: Classes, Objetos e Atributos.</p>
Quais são <b>algumas linguagens</b> que utilizam a POO?		<p>Nós podemos dividir a linguagens que utilizam POO em 2 grupos:</p> <ul style="list-style-type: none"><li>- <b>Linguagens Baseadas em Classes</b> (também conhecidas como "<b>Baseadas em Objetos</b>" ou "<b>Multi-Paradigma</b>"), que são linguagens que não são totalmente orientadas á objetos, mas utilizam conceitos da POO mesclados a outros paradigmas da programação, algumas delas são: <b>C++, PHP, JavaScript, Python, Ruby, C#, VisualBasic, Swift;</b></li><li>- <b>Linguagens Orientada a Objetos</b>, são linguagens totalmente orientadas a objetos, ou seja, que não podem ser programadas em outro paradigma. Um bom exemplo disso é o <b>Java</b>;</li></ul>
Quais as vantagens de usar <b>Linguagens Orientadas a Objeto</b> ?		<ul style="list-style-type: none"><li>- <b>Confiabilidade:</b> devido ao isolamento das classes e métodos podemos gerar softwares seguros, pois se alterarmos uma parte do software, outras partes não serão alteradas;</li><li>- <b>Oportuno:</b> como cada parte pode ser desenvolvida separadamente, não precisamos desenvolver uma parte primeiro para depois desenvolver a outra, podemos desenvolvê-las na ordem que quisermos, até mesmo em paralelo, aproveitando melhor o nosso tempo;</li><li>- <b>Manutenível:</b> podemos aplicar manutenções e atualizações no software com facilidade, pois cada classe é isolada, se aplicarmos alterações a uma dessas classes não vamos causar conflitos entres as outras partes;</li><li>- <b>Extensível:</b> podemos gerar novas classes e métodos o quanto quisermos, caso desejemos melhorar o sistema, ou seja, podemos extendê-lo o quanto nós quisermos;</li><li>- <b>Reutilizável:</b> podemos reutilizar classes para outras partes do código ou para códigos futuros;</li><li>- <b>Naturalidade:</b> a transformação dos objetos para o código fonte é muito natural, pois a POO permite essa transformação com facilidade;</li></ul>
- reservada para a questão acima -		<p>Para facilitar podemos usar a sigla <b>COMERNada</b>, levando em consideração somente as 6 primeiras letras, que são um acrônimo para as vantagens do POO.</p>
- reservada para a questão acima -		