**PESQUISA**

***Herança e Polimorfismo***

**DORIVAL**

**- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

**HERANÇA**

**É uma forma de compartilhar atributos, métodos e outros membros de classes entre si.**

**- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

**Tipos principais de classe que vamos poder encontrar dentro da herança**

**- Classe Base: Vai ser a classe responsável por conceder as características de uma outra classe.**

**- Classe Derivada: É a classe que vai herdar as características atribuídas na classe base.**

**- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

**O fato de as classes derivadas herdarem atributos das classes bases assegura que programas orientados a objetos cresçam de forma linear e não geometricamente em complexidade.**

**Cada nova classe derivada não possui interações imprevisíveis em relação ao restante do código do sistema.**

**Com o uso da herança, uma classe derivada geralmente é uma implementação especifica de um caso mais geral.**

**- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

**A classe derivada deve apenas definir as características que a tornam única.**

**- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

**EXEMPLO PRÁTICO: Código**

**Crie um projeto do tipo Console Application, dando a ele o nome de ExemploHeranca e dê OK.**

**Crie também uma classe dando a ela o nome Animal, esta classe será a classe pai ou classe base.**

**Nesta classe, adicione o seguinte código:**

**Texto

O conteúdo gerado por IA pode estar incorreto.**

**Neste código podemos encontrar 4 campos que vão definir os atributos que queremos dar a classe “ Animal “.**

**Consideremos a criação de um cadastro de cachorros do reino Animal. Entendemos que cada Cachorro é um Animal, mais nem todo Animal é um Cachorro.**

**Essa relação nos obriga a criar uma extensão da classe Animal a fim de que a mesma contenha os dados específicos referentes aos cachorros. Assim, a extensão criada acaba herdando os membros existentes na definição da classe Animal.**

**Texto

O conteúdo gerado por IA pode estar incorreto.**

**Note que nessa classe nós fizemos o uso de “ : “ ( Cachorro : Animal) fazemos isso para informar que a classe irá herdar as características da classe Animal.**

**Assim, temos a definição da nossa classe “ Cachorro “ que vai receber através do modo de Herança todas as caraterísticas da classe “ Animal”**

**- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

**Operador – Base :**

**Texto

O conteúdo gerado por IA pode estar incorreto.Esse operador nos dá acesso á classe Pai, assim ligando as classes! Então qualquer operador que possuir uma referência usando um Operador Base, dentro de uma classe filha, esta então será apontada para a classe Pai.**

**Na Filha, adicione os campos com o mesmo nome mais com valores diferentes (note o uso da instrução new) e crie o método que irá exibir os valores de ambas as classes em tela:**

**Texto

O conteúdo gerado por IA pode estar incorreto.**

**Agora no Program.cs, você devera instanciar a classe filha e chamar esse método de recém-criado:**

**Texto

O conteúdo gerado por IA pode estar incorreto.**

**Compile e tenha seu resultado!**

Tela de computador com texto preto sobre fundo branco

O conteúdo gerado por IA pode estar incorreto.

**- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

**POLIMORFISMO**

**- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

**Definimos Polimorfismo como um princípio a partir do qual as classes derivadas de uma única classe base são capazes de invocar os métodos que, embora apresentem a mesma assinatura, comportam-se de maneira diferente para cada uma das classes derivadas.**

**O Polimorfismo é um mecanismo por meio do qual selecionamos as funcionalidades utilizadas de forma dinâmica por um programa no decorrer de sua execução.**

**Com o Polimorfismo, os mesmos atributos e objetos podem ser utilizados em objetos distintos, porém, com implementações lógicas diferentes.**

**Texto

O conteúdo gerado por IA pode estar incorreto.**

**Na classe Vendedor temos o mesmo método, mais com a codificação diferente:**

**Texto

O conteúdo gerado por IA pode estar incorreto.**

**Vamos ver o mesmo na classe nomeada como Diretor**

**Texto

O conteúdo gerado por IA pode estar incorreto.**

**DIFERENÇAS ENTRE**

**HERANÇA E POLIMORFISMO**

**Herança é quando uma classe se aproveita do que já existe em outra classe**

**O Polimorfismo é quando nós usamos o mesmo método em varias classes em várias classes, com comportamentos diferentes.**

**A herança permite que uma classe aproveite atributos e métodos de outra classe. Isso ajuda a evitar repetição de código e facilita a organização. Por exemplo, se temos uma classe "Animal", podemos criar uma classe "Cachorro" que herda tudo o que "Animal" tem, e ainda pode ter coisas próprias.**

**O polimorfismo permite que diferentes classes usem o mesmo método, mas com comportamentos diferentes. Por exemplo, se várias classes têm um método chamado "falar", cada uma pode fazer isso de forma diferente: um cachorro pode "latir" e um gato pode "miar".**

**Esses dois conceitos estão ligados porque, com a herança, podemos criar métodos com o mesmo nome em classes diferentes, e o polimorfismo permite usar esses métodos sem se preocupar com qual classe exatamente está sendo usada.**

**Vantagens de cada uma:**

**A herança permite reutilizar código, já que uma classe pode herdar atributos e métodos de outra. Isso ajuda a evitar repetições e deixa o sistema mais organizado. Além disso, facilita a manutenção, pois mudanças feitas na classe base se refletem nas subclasses. A herança também permite estender funcionalidades, criando novas classes a partir de outras já existentes, sem precisar reescrever tudo do zero.**

**Já o polimorfismo oferece mais flexibilidade ao código. Ele permite que diferentes classes usem o mesmo método, mas com comportamentos próprios. Isso torna o sistema mais fácil de expandir e adaptar, pois novas classes podem ser incluídas sem mudar o que já foi feito. Além disso, o polimorfismo reduz o acoplamento entre as partes do sistema, facilitando a manutenção e promovendo um código mais limpo e organizado.**

**Referencias:**

**crieseusjogos.com.br**

**blog.grancursosonline.com.br**

**dcm.ffclrp.usp.br**

**dio.me  
devmedia.com.br**

**learn.microsoft.com**