



Introdução à Programação Orientada a Objetos (POO)



fundaçāo bradesco | escola virtual

Sumário

Apresentação	4
Módulo 1	6
Programação estruturada <i>versus</i> orientada a objetos	6
Programação estruturada <i>versus</i> orientada a objetos	7
Programação orientada a objetos	12
Comparação entre programação estruturada e a POO	18
Módulo 2	22
Os quatro pilares da programação orientada a objetos	22
Os quatro pilares da programação orientada a objetos	23
Abstração	23
Encapsulamento	25
Herança	28
Polimorfismo	29

Sumário

Módulo 3	31
Linguagens orientadas a objeto	31
Linguagens orientadas a objeto	32
Linguagem Java	32
Vantagens e desvantagens da linguagem Java	33
Linguagem C++	38
Diferenças e semelhanças entre Java e C++	41
Linguagem C#	42
Características da linguagem C#	44
Vantagens e desvantagens da linguagem C#	45
Fechamento	51
Referências	52

Apresentação

Olá!

Bem-vindo(a) ao curso **Introdução à Programação Orientada a Objetos (POO)**!

O objetivo desse curso é apresentar o conceito da programação orientada a objetos e as diferenças entre a programação estruturada e a POO. Além de falar sobre seus fundamentos e mostrar alguns exemplos de linguagens orientada a objetos.

Desejamos a você um bom aprendizado!



Vídeo

Confira o [vídeo](#) de apresentação do curso.



Perdeu algum detalhe? Confira o que foi abordado no vídeo.

Olá! Seja bem-vindo(a) ao curso de Introdução a Programação Orientada a Objetos.

Este curso está organizado em três módulos, de forma que facilite seu aprendizado sobre a Programação Orientada a Objetos.

No primeiro módulo, você verá os conceitos das diferenças entre a programação estruturada e a programação orientada a objetos (POO).

No segundo módulo, verá os quatro pilares da POO, ou seja, a abstração, o encapsulamento, a herança e o polimorfismo.

Por fim, no terceiro módulo veremos sobre as linguagens orientadas a objeto, como o Java, C++ e C#.

Preparado(a) para começar?



Módulo 1

Programação estruturada *versus* orientada a objetos

Programação estruturada *versus* orientada a objetos

Neste primeiro módulo, você conhecerá um pouco sobre **programação estruturada** (como funciona e qual a sua estrutura) e sobre **programação orientada a objetos** (como funciona e alguns conceitos).

Antes de entender as diferenças entre estas duas programações, é necessário entender os **conceitos** de cada uma.



Imagine que quando falamos dessas duas programações, estamos falando de duas pessoas diferentes. É importante saber que apesar dessas duas pessoas terem o mesmo objetivo de vida, que é programar, cada uma tem seu jeito particular de pensar.

Vamos entender melhor como é a programação estruturada e como ela funciona?

#PraCegoVer

Na imagem, há uma mulher e um homem em um café sentados na frente de um notebook. Eles estão discutindo sobre o conteúdo do monitor ao mesmo tempo em que tomam café. A mulher está apontando para a tela e o homem observando para o lugar apontado.

Programação estruturada

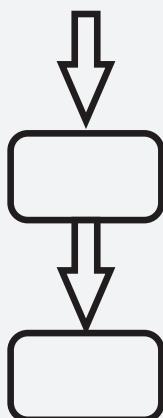
É preciso saber que este tipo de programação segue uma **lógica e sequência de pensamentos de uma máquina**. Ou seja, a sua lógica é direcionada à linguagem de máquina, já que ela realiza o que foi orientado pelo programador por meio de uma linguagem estruturada.

Para saber mais sobre isso, observe o recurso abaixo. As imagens contidas nele representam fluxogramas (símbolos para representação de regras e procedimentos lógicos que levam a solução de um problema) das interações já mencionadas:

sequência, decisão e iteração.

▼ Sequência

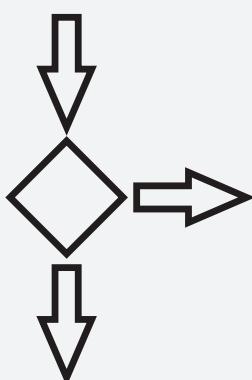
Segue uma sequência lógica para realizar determinada ação, ou seja, os comandos.



#Pracegover: Sequência – é composta por uma Seta no sentido vertical, com indicação para baixo, seguida de um retângulo vazio, outra seta no mesmo sentido que a anterior seguida de outro retângulo.

▼ Decisão

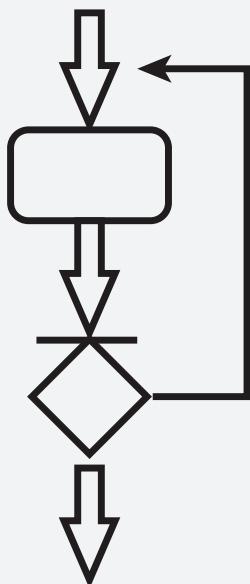
Uma ou mais decisões devem ser testadas pelo programa, por exemplo: *If Else*, *Else If*, *SWITCH-CASE*.



#Pracegover: Decisão - é composta por uma seta no sentido vertical, com indicação para baixo, seguida de um losango, depois outra seta também na vertical com indicação para baixo. Na ponta da lateral direita do losango há uma seta, na horizontal, com a indicação da esquerda para a direita.

▼ Iteração

É uma repetição que pode ser interrompida quando a condição for satisfeita, por exemplo: mostre na tela números de 1 a 100, o programa iniciará mostrando o número 1 na tela e só finalizará quando chegar ao número 100, que neste caso é a condição a ser satisfeita. As estruturas a serem utilizadas para realizar essa tarefa pode ser a *FOR*, a *WHILE* e a *DO WHILE*.



#Pracegover: Iteração - é composta por uma seta no sentido vertical, com indicação para baixo, seguida de um retângulo, depois outra seta também na vertical com indicação para baixo, seguida de um traço na horizontal e um losango. Depois do losango há uma seta, na vertical, com a indicação para baixo. Na ponta da lateral direita do losango sai uma seta que faz um traço em linha reta, depois vai no sentido para cima e depois para a direita até encontrar a primeira seta apontada na figura.

Diferente da programação orientada a objetos, a estruturada segue **processos** (sequência contínua de fatos) para conseguir atingir o seu objeto final, que é a **solução de um determinado problema**. Imagine que existe um programa a ser desenvolvido, onde várias etapas precisam ser cumpridas. Na programação estruturada esse programa é dividido em vários processos pequenos e depois que todos estão resolvidos, os juntamos para formar a solução final do problema.

Para entender de forma mais clara o conceito apresentado, observe o exemplo a seguir:



Em uma pizzaria, você trabalha como pizzaiolo, e o problema que precisa resolver é o de fazer uma pizza de calabresa. Muitos processos devem ser realizados para que você consiga realizar sua tarefa, não é mesmo?

#PraCegoVer

Uma mesa de madeira com três massas de pizzas ainda crus, mas com o recheio já colocado. Na pizza em destaque, uma mão está colocando uma fatia de calabresa nela.

Para isso você segue, passo a passo, cada uma das atividades, como:

- 1.** Preparar a massa;
- 2.** Colocar o recheio;
- 3.** Ligar o forno, se não estiver ligado;
- 4.** Colocar a pizza para assar;
- 5.** Retirar a pizza do forno depois de 10 minutos.

Pronto, você conseguiu preparar a pizza de calabresa corretamente.



#PraCegoVer

Mesa de madeira com uma tábua e um rolo também de madeira. Em cima da mesa também tem pedaços de massa de pizza e em cima da tábua uma massa sendo aberta por um pizzaiolo.



É assim que funciona a programação estruturada, ela é realizada por sequência como a preparação de uma pizza!

#PraCegoVer

Uma mesa de madeira com uma pizza de calabresa, queijo e rodelas de cebola vista de cima.

Saiba Mais



Para conhecer mais sobre a programação estruturada, as estruturas por meio de fluxogramas e como elas são utilizadas na programação, convidamos você a assistir o vídeo [Fluxogramas e programação estruturada](#).

Neste tópico, você compreendeu acerca do conceito de programação estruturada, a qual é fundamentada em processos destinados à resolução de um determinado problema. Porém, ainda é necessário conhecer a programação orientada a objetos. Desta forma, a seguir, vamos nos aprofundar nesse conceito.

Programação orientada a objetos

Agora que você compreendeu sobre a lógica por trás da programação estruturada, vamos entender como funciona a programação orientada?

Como você viu, as duas são um modo de programação, mas cada uma possui a sua própria lógica de funcionamento.



Enquanto a programação estruturada obedece a uma lógica de máquina, a **programação orientada a objetos** funciona de um modo mais similar ao **raciocínio humano**. Os seus códigos são organizados de acordo com objetos, e não processos, como no caso da programação estruturada.

#PraCegoVer:

Um plano de fundo preto com um cérebro em formato 3D com vetores. Tanto o cérebro quanto os vetores são azul claro.

Para ajudá-lo a compreender melhor, convidamos você a assistir o vídeo que preparamos sobre como surgiu a POO.



Vídeo

Confira o [vídeo](#) sobre a origem da programação orientada a objetos.

Perdeu algum detalhe? Confira o que foi abordado no vídeo.

Olá! Agora que você já sabe como funciona e qual a estrutura tanto da programação estruturada quanto da programação orientada a objetos, chegou o momento de compreender como surgiu a programação orientada a objetos. Vamos lá?

Ela foi desenvolvida na Noruega em meados de 1962 no Centro Norueguês de Computação (*Norwegian Computing Center* - NCC) da Universidade de Oslo.

Os pesquisadores Kristen Nygaard e Ole-Johan Dahl aceitaram o desafio de criar uma linguagem de simulação de eventos discretos.

De modo não surpreendente, eles decidiram batizar a linguagem de SIMULA. Mais tarde, devido a uma reformulação, ela foi chamada de SIMULA I e, em sua segunda e final versão, foi chamada de SIMULA 67. Esta última versão é reconhecida como a primeira linguagem de renome no que diz respeito ao universo da Orientação a Objetos.

Interessante não é mesmo?

Com isso, o surgimento da programação orientada a objetos aconteceu como forma de facilitar a programação, o que seria quase impossível com a programação estruturada.

De acordo com o vídeo, o modo de programar orientado a objetos surgiu da **necessidade de deixar a programação mais fácil**, possibilitando ao desenvolvedor entregar um software que satisfaça completamente o cliente. Algo muito difícil e às vezes, quase impossível, com a programação estruturada.

Para que isso ocorra, é necessário que você, como programador, ensine a máquina a pensar como os seres humanos, mostrando como o nosso mundo funciona.

Certo, mas como se faz isso?

Primeiro, você precisa compreender alguns conceitos! Vamos lá?

Objeto

São elementos do mundo real para o mundo da programação. O objeto é criado a partir de uma classe. É algo que se visualiza, se utiliza e assume um papel no domínio do problema. Um exemplo disso, é um Fusca 1964.

Atributos

São as características do objeto. Considerando o exemplo do Fusca 1964, os atributos deles são: ano de fabricação é 1964 e que ele tem um formato arredondado e o motor fica na parte traseira.

Métodos

São os comportamentos do objeto. Ainda de acordo com o exemplo, o Fusca pode se locomover, transportando pessoas, cargas etc.

Classes

São os agrupamentos que descrevem todos os objetos de um único tipo. Por exemplo, o fusca pertence à classe dos carros. Não importa a sua marca ou ano. O que importa é que são objetos que se encaixam nesta classe, que serve como molde para a criação dos objetos. É na classe que são moldados os objetos que serão utilizados do programa a ser desenvolvido.

Nas imagens apresentadas abaixo, temos exemplos de alguns objetos do mundo real que podem ser levados para a programação orientada a objetos.



#PraCegoVer: Duas imagens mostram objetos que podemos encontrar no mundo real. Na da direita, existem várias televisões em uma sala escura. Na da esquerda, existe um smartphone com um fundo colorido e claro.

Assim, você entendeu os conceitos fundamentais na programação de objetos, os quais são os objetos e as classes. Todavia, é preciso também compreender um outro conceito correlato: o de atributos.

Os **atributos** são simplesmente as **características de um objeto**. Ou seja, no nosso exemplo de carro, podemos ter atributos como modelo, placa, ano, cor, tamanho etc.



#PraCegoVer

Na imagem, há uma nuvem de palavras com os seguintes verbetes: bonita, engraçada, esperta, grande, gorda, educada, triste, grande e pequena.

Veja outro exemplo, reunimos alguns atributos que podem ser inseridos em uma classe de um objeto do tipo “pessoa”.

Nas palavras apresentadas, é possível notar alguns atributos relacionados à classe pessoa (bonita, engraçada, esperta, grande, gorda etc.), ou seja, nessa classe, é possível encontrar pessoas (objetos) com tais características (atributos).

Já os métodos são **todos os tipos de ação que um objeto pode realizar**. No caso de um carro podemos pensar em ações como dar ré, ir para frente, frear etc. No caso de uma pessoa, podemos pensar em ações como comer, correr, falar, andar, entre outras diversas ações que uma pessoa pode realizar.

Para que esses conceitos fiquem mais claros, assista ao vídeo que preparamos para você com mais um exemplo.

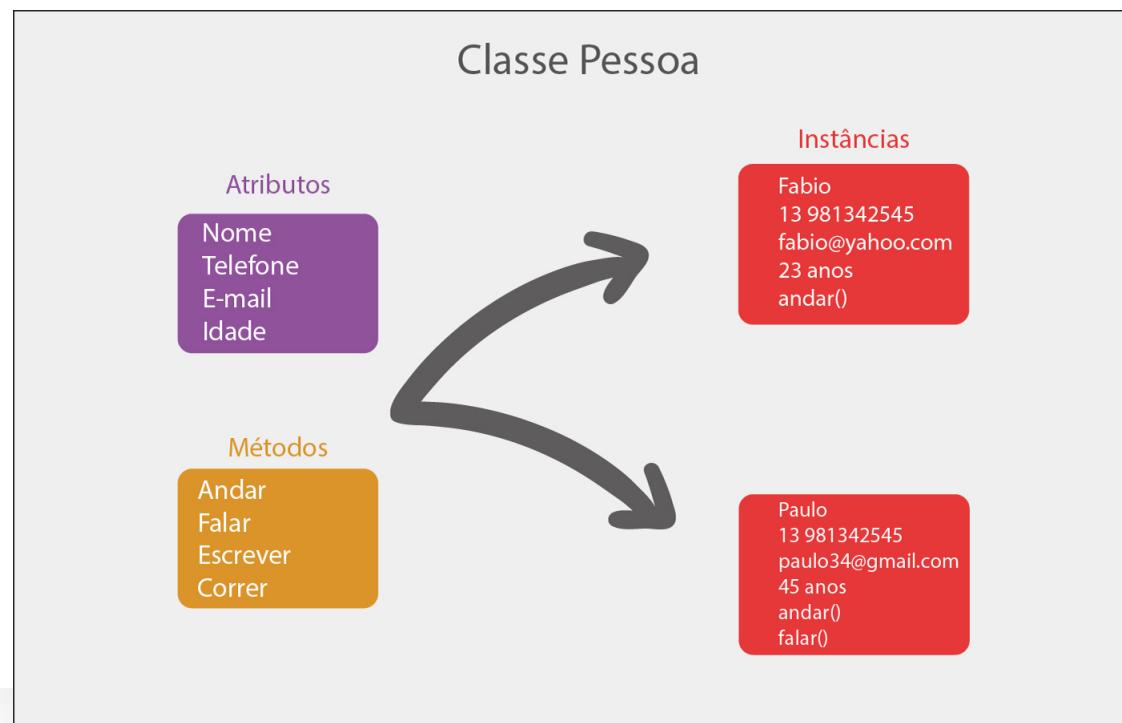


Vídeo

Confira o [vídeo](#) sobre conceitos de programação orientada a objetos.

Perdeu algum detalhe? Confira o que foi abordado no vídeo.

Olá! Agora que você já compreendeu os conceitos fundamentais da programação de objetos, confira neste vídeo um exemplo de aplicação que reúne o conceito de método e instância.



Observe a classe chamada “Pessoa”, que possui atributos como: nome, telefone, e-mail, idade. Lembre-se que os métodos dessa classe são as ações que essa pessoa pode realizar como: andar, falar, escrever e correr. Note que agora temos um novo conceito, as instâncias. As instâncias são novos objetos que são criados a partir de uma classe. De maneira simples, são objetos que tem o seu comportamento e estado definidos por uma classe.

Dessa forma, todos os novos objetos ou instâncias terão o mesmo conjunto de atributos definidos na classe, porém o conteúdo desses atributos é individual, ou seja, de cada objeto. Portanto, podemos observar que as instâncias Fabio e Paulo, apesar de serem compostas pelo mesmo conjunto de atributos, possuem conteúdos individuais.

Na prática fica mais fácil compreender essas questões, não é mesmo?

Com isso, pudemos constatar que os conceitos “Método” e “Instância” estudados aqui são essenciais para a programação orientada a objetos.

Continue seus estudos e se aprofunde cada vez mais.

Assim, no vídeo você viu o conceito de método e de instância, os quais ficaram ainda mais concretos por meio da exibição de um exemplo prático.

Saiba Mais



Indicamos ainda o vídeo: [O que é Programação Orientada a Objetos?](#), o qual é possível se aprofundar mais nesses assuntos sobre os métodos e suas vantagens, classes, objetos e atributos. Vale a pena você conferir!

Até este ponto, você estudou acerca do conceito de método e de instância, os quais, juntos com os de objeto, classe e atributos, são essenciais para a prática da programação orientada a objetos.

Agora que possui uma base teórica mais consistente em relação à programação

estrutura e à programação orientada a objetos, a seguir, acompanhe uma reflexão comparativa entre ambas.

Comparação entre programação estruturada e a POO

Neste tópico, iremos promover uma comparação entre a programação estruturada e a programação orientada a objetos, com o intuito de não só reforçar as características singulares de cada uma, mas também consolidar o que você já sabe em relação à elas.



#PraCegoVer

Mulher sentada em sua mesa de trabalho consultando um tablet segurado por sua mão direita e na frente de um computador. Na imagem estão grafismos de códigos transparentes que cobrem a imagem inteira em primeiro plano.

Então, preparado para aprender as diferenças entre elas? Observe a imagem a seguir!

A primeira coisa que precisamos ter em mente é que os dois tipos de programação possuem suas vantagens e desvantagens de uso pelos desenvolvedores. Além disso, determinar qual a melhor programação a ser utilizada pode depender do tipo de programa que será desenvolvido.



#PraCegoVer: É uma imagem que contempla a programação estruturada e a programação orientada a objetos. No lado esquerdo temos a programação estruturada com os dados globais divididos em cinco campos de procedimentos. Já no lado direito temos a programação orientada a objetos com três divisões de Dados Objeto e que cada uma delas contém dois métodos cada.

Perceba que na programação estruturada as funções são utilizadas globalmente na aplicação, já na orientada a objetos, essas funções são aplicadas aos dados de cada objeto.

Agora, conhecendo os dois tipos de programação, observe que a programação estruturada, quando realizada corretamente, tem a probabilidade de ter o desempenho superior ao da orientada a objeto. Isso ocorre, pois ela é procedural e realizada em sequência, onde cada linha de código é executada logo após a outra, sem desvios. Ao contrário do que ocorre na programação orientada a objetos.

Notou que é possível analisar claramente as diferenças entre esses dois modos de programar quando falamos de dados?

Deste modo, você percebeu que os procedimentos (funções) são os mesmos, o que muda é onde eles são declarados.

Como a programação estruturada possui uma linguagem mais parecida com a da máquina, ela permite que o programador utilize melhor o desempenho do hardware, resultando em um código mais eficiente.

Convidamos você a assistir ao vídeo, para saber mais sobre o hardware e as peças que o compõem. Vamos lá!



Vídeo

Confira o [vídeo](#) sobre o hardware e os principais componentes do computador.

Perdeu algum detalhe? Confira o que foi abordado no vídeo.

Olá! Vamos falar sobre hardware e os principais componentes do computador? O hardware nada mais é que todas as peças que compõem o computador, como: a Placa Mãe, que é o principal componente do computador, pois nela são ligados todos os outros componentes, como o processador, memória, fonte etc.

Portanto, ligado à Placa Mãe, está o primeiro componente que abordaremos: o processador, que é a Unidade Central de Processamento. É nele onde o computador realiza as instruções de um programa.

Enquanto a memória RAM e ROM guardam informações para que os programas consigam funcionar. Sendo a memória RAM de acesso aleatório, que guardam as memórias voláteis, ou seja, nada fica armazenado nela depois que o computador é reiniciado. É nela que programas e arquivos são armazenados temporariamente para serem utilizados.

Já a memória ROM oferece dados apenas para leitura, armazenando *firmwares*, que controlam as funções mais básicas do dispositivo.

O Disco Rígido (ou *Hard Disk* em inglês, mais conhecido por suas siglas HD), por sua vez, armazena todos os arquivos existentes no computador como textos, imagens, músicas, fotos entre outros.

Agora que você está por dentro dos principais componentes de um computador, confira na sequência os pilares da programação orientada a objetos

No vídeo, você conheceu mais sobre o conceito de hardware, sobretudo, observando os principais componentes do computador, isto é, a placa mãe, a memória RAM e ROM, o disco rígido e o processador.

Até aqui, analisamos as várias vantagens da programação estruturada. Mas você deve estar se perguntando, por que então utilizar a programação orientada a objetos?

Para trazer ainda mais benefícios quando falamos em desenvolver aplicações mais modernas. Pois, hoje em dia o hardware possui uma ótima capacidade de processamento. Logo, o desempenho da aplicação não é mais uma grande preocupação. Isso fez com que a POO se tornasse muito utilizada no mundo todo. Outra vantagem apresentada por esta programação é que ela possibilita reutilizar o código, além de ser um código muito próximo ao mundo real, o que fortaleceu o seu crescimento mundialmente.

Saiba Mais



No vídeo [PHP Orientado a Objetos - Programação Estruturada vs Orientação a Objetos](#), você pode complementar a definição dos dois paradigmas de programação estruturada e POO. Vale a pena conferir!

Parabéns! Você chegou ao final do módulo 1.

Aqui você compreendeu sobre a programação estruturada, como ela funciona e qual a sua estrutura, cuja lógica e sequência seguem os pensamentos de uma máquina.

Somado a isso, você também viu acerca da programação orientada a objetos, isto é, a programação próxima ao raciocínio. Além disso, você acompanhou brevemente a história desta última, bem como os conceitos de objetos, atributos, métodos, classes e instâncias.

No entanto, ainda é necessário entender os quatro pilares da programação orientada a objetos, assunto que você estudará no próximo módulo. Acompanhe!



Módulo 2

Os quatro pilares da programação orientada a objetos

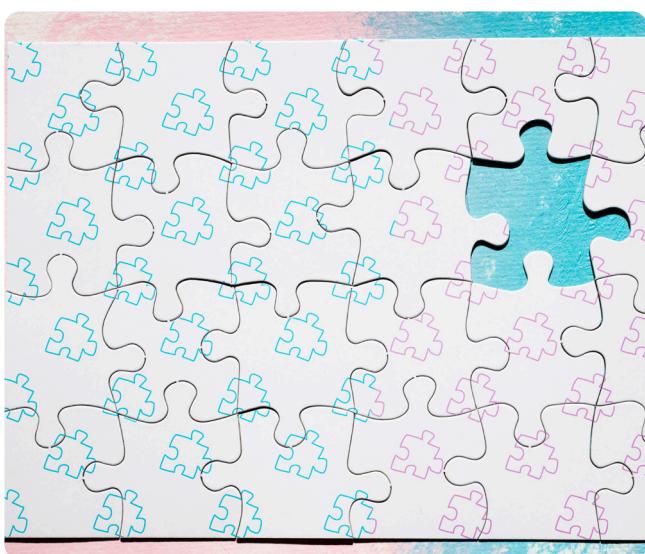
Os quatro pilares da programação orientada a objetos

No módulo anterior, você estudou sobre a diferença entre programação estruturada e a programação orientada a objetos.

Agora, neste módulo, você aprofundará mais seus conhecimentos, sobretudo, estudando sobre alguns elementos que o ajudará a entender melhor a programação orientada a objeto, explicando seus quatro mais importantes componentes: **abstração, encapsulamento, herança e polimorfismo**. Vamos começar?

Abstração

Na POO, damos o nome de **abstração** para o **processo de aproximar o mundo real do mundo da programação**, sendo o seu objetivo, simplificar um problema difícil. Para isso, a abstração leva em conta os aspectos importantes de um determinado ponto de vista e desconsidera os aspectos restantes.



Assim como representado na imagem, a abstração é a criação de uma classe abstrata, que é uma **classe incompleta**, como se fosse um quebra-cabeça. Assim, esta classe **não permite a criação de instâncias** e obriga a implementação de todos os métodos da classe assinados como '*abstract*'.

#PraCegoVer

Imagen de um quebra cabeças, mostrando que falta umadas peças, algo incompleto.

Resumindo, a abstração se concentra apenas nas informações que são importantes para o seu propósito. Dessa forma, ela mantém suas classes o mais simples possível, **concentrando-se apenas no que é importante** para uma determinada finalidade.

A abstração nada mais é do que você **abstrair coisas**, ou seja, quando você tem algo muito grande, mas não há necessidade de cadastrar todas as informações.

Vamos entender melhor este conceito a partir do seguinte exemplo:

Na prática



Vamos supor que você está desenvolvendo um código com a classe “ser humano”. Essa classe é composta por inúmeras variantes (atributos), como por exemplo: altura, peso, cor da pele, cor do olho, CPF, nome, endereço etc.

O objetivo do seu código é tratar o “ser humano” como cliente. Para cada cliente é preciso obter as informações de CPF, Nome e Endereço. As outras informações que pode obter na classe “ser humano”, como cor da pele, cor do olho, altura, peso etc. não são importantes para esta situação.

Portanto, podemos abstrair estas informações e considerar somente o que importa.



Saiba Mais

O vídeo Aula de [Java - abstract, classes abstratas](#), explica um pouco mais o que é abstração, e introduz sobre o tema do nosso próximo tópico, o encapsulamento! Vale a pena conferir!

Encapsulamento

Agora que já entendemos o que é abstração, vamos aprender sobre o encapsulamento. Ao encapsular algo, você está colocando um objeto dentro de um recipiente, igual a um remédio de cápsula.



#PraCegoVer: Na imagem, existem onze cápsulas de remédio em cima de uma superfície colorida.

Mas, afinal, qual é o propósito?

O encapsulamento é uma das **principais técnicas da programação orientada** a objetos. Quando você encapsula um objeto, você está criando uma **proteção** e um **padrão**. Com isso, o propósito é de **proteger** o desenvolvedor do código e o código do desenvolvedor.

Assim, quando você encapsula um objeto na POO, você está criando **moldes padrão** que fazem com que o conteúdo do objeto não importe. Você está determinando que o resultado será sempre o mesmo.

Observe abaixo a definição dos conceitos:

Conceito de encapsulamento

É a ação de **ocultar partes** independentes da implementação, permitindo construir partes invisíveis ao mundo exterior. Ok, mas se no fim eu estou ocultando detalhes do código, como que ele vai funcionar? A POO permite que você converse com esta cápsula, trocando informações entre o mundo externo e o objeto por meio de mensagens. Portanto, ao enviar mensagens para essa cápsula, você vai **obter uma resposta, sem precisar entrar nela.**

Mas não podemos esquecer de um importante detalhe! Para que haja esta troca de mensagens, precisamos desenvolver uma interface. É a interface que permite esta comunicação com o mundo externo.

Conceito de interface

É uma lista de serviços fornecidos por um componente. É o contato com o mundo exterior, que define o que pode ser feito com um objeto de determinada classe. Ou seja, a interface é uma classe composta apenas por métodos (não possui atributos).



Atenção

Encapsular não é obrigatório na POO, mas é uma ótima prática para produzir classes mais eficientes.

Anteriormente, você aprendeu sobre abstração, certo? Que nada mais é do que simplificar uma classe, se concentrando apenas em informações importantes e relevantes para o propósito do código.

No encapsulamento esse conceito é muito importante, já que os métodos da classe encapsulada serão abstratos, ou seja, os métodos abstratos são previstos ali, mas não são implementados na interface.

Para entender melhor esse processo, pense, por exemplo, nas máquinas de café expresso nas padarias.



#PraCegoVer: Na imagem, há uma máquina de café vista pela diagonal. No fundo, existem vários copos empilhados

Como ela é feita, não é algo que precisamos saber para poder obter o resultado do processo (nossa café quentinho). Por isso, a máquina é um objeto encapsulado, onde os ingredientes e o mecanismo do preparo do café ficam escondidos atrás da interface externa da máquina, composta por botões de comando (como ligar, desligar, tipos de bebida etc.) que auxiliam na preparação.



Saiba Mais

No vídeo [Pilares da POO: Encapsulamento](#), você terá uma explicação complementar sobre esse pilar da POO. Vale a pena conferir!

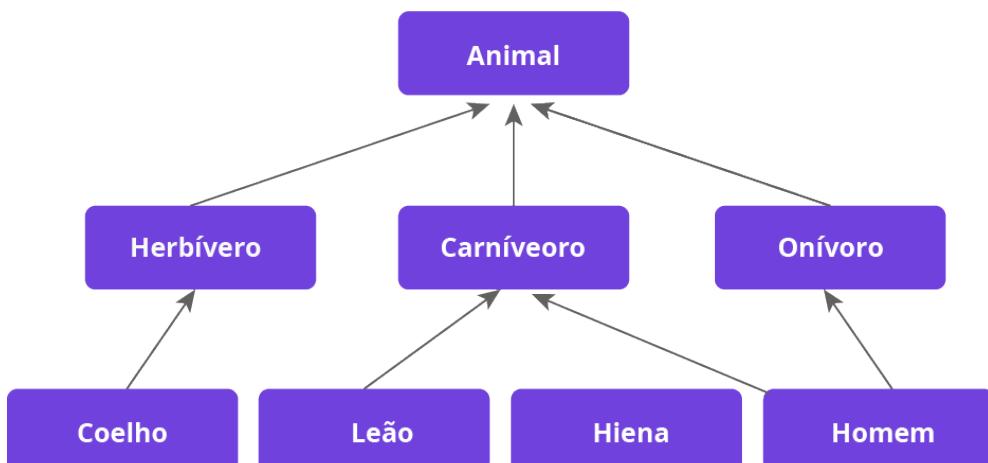
Neste tópico, você compreendeu o conceito de encapsulamento e como ele é aplicado na POO. Contudo, a seguir, vamos aprender sobre outro conceito muito relevante para a programação orientada a objetos, a herança.

Herança

Outro pilar da POO é a **herança**. Assim como no mundo real, a herança na POO também diz respeito à **ação de herdar**. Ela nada mais é do que um objeto poder ser criado em uma outra classe, **levando consigo todos os atributos já existentes** em sua classe de origem.

A herança é uma maneira de **reutilizar o código** já existente em uma nova classe. Desta forma, o código é aprimorado com novas e melhores capacidades. Ao utilizar este pilar, os programadores economizam tempo de desenvolvimento de um programa, já que eles reutilizam códigos já testados e aprovados.

No organograma abaixo, mostra um exemplo de herança. Ao analisá-la, percebe-se que as classes herbívoro, carnívoro e onívoro podem herdar quaisquer atributos necessários da classe animal: tamanho, raça, cor etc. Da mesma forma, as classes leão, hiena, homem e coelho podem herdar atributos das classes herbívoro, carnívoro ou onívoro.



#Pracegover: A imagem contém um organograma, sendo a leitura feita de baixo para cima. Onde na primeira linha contém quatro caixa com as informações: Coelho, Leão, Hiena, Homem.

Coelho tem uma seta indicada para a caixa Herbívoro e este por sua vez, uma seta para a caixa Animal.

Leão e Hiena possuem setas indicadas para a caixa Carnívoro que leva a seta para a caixa Animal.

Homem tem uma seta indicada para a caixa Onívoro, que por sua vez também leva a seta para a caixa Animal.

Saiba Mais



No texto sobre [Herança](#), de autoria de Evandro Eduardo Seron Ruiz, disponibilizado pela Universidade de São Paulo (USP), você encontra informações complementares sobre a herança, além de conhecer alguns tipos e ver como ela se comporta dentro de um código escrito com a linguagem Java, que ainda vamos conhecer neste curso.

Polimorfismo

E o polimorfismo? Você sabe o que é e como funciona?

O polimorfismo é um pilar da POO que é utilizado para que **duas classes façam uso do mesmo método**, implementando-o de formas diferentes. Ele permite que o programador desenvolva o código de forma ampla ao invés de perder muito tempo no desenvolvimento de códigos específicos. Ou seja, o polimorfismo permite que sistemas sejam escritos de forma a processar objetos que compartilham a mesma superclasse (classe já existente), como se eles fossem parte direta dela.

Por exemplo, na superclasse Animal, estabelecemos o método “emitir o som do objeto animal”, ou seja, os objetos pato, cachorro e gato devem emitir um som ao comando do método, mas cada um fará isso de um jeito diferente.

Agora, vejamos outro exemplo, onde foi desenvolvido um programa para o estudo que simula o modo de locomoção de animais com as classes Peixe, Anfíbio e Pássaro. Cada uma dessas classes é extensão da superclasse Animal, que possui o método “mover” e controla a localização destes seres. Para simular a locomoção, o programa manda a mesma mensagem (mover) a cada objeto. Porém, cada uma das classes (tipo de animais) responde ao comando de um jeito diferente: o peixe nada, o anfíbio pula e o pássaro voa.

Observe que nos exemplos apresentados, o polimorfismo foi este processo de implementar métodos (emitir som e mover) na superclasse Animal, para que as classes (pato, cachorro e gato; peixe, anfíbio e pássaro) possam obedecer a um mesmo comando, mesmo que de maneiras diferentes.



Saiba Mais

Sugerimos o vídeo sobre [Polimorfismo em Java](#), onde você encontrará mais informações sobre o polimorfismo e alguns exemplos práticos do dia a dia que o ajudarão a fixar melhor esse conceito. Confira!

Parabéns! Você chegou ao final do módulo 2.

Aqui, a partir de exemplos concretos, você compreendeu os quatro principais conceitos da programação orientada a objetos, os quais são a abstração, o encapsulamento, a herança e o polimorfismo. Vale ressaltar que eles são essenciais não só para o domínio da POO, mas também para a sua futura atuação profissional nessa área.

Contudo, para que isso ocorra plenamente, é relevante que você aprenda as linguagens orientadas a objeto, as quais veremos no próximo módulo. Vamos lá!



Módulo 3

Linguagens orientadas a objeto

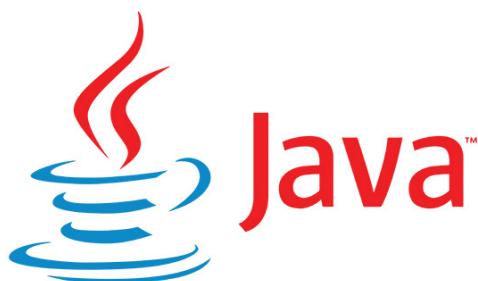
Linguagens orientadas a objeto

Conforme você viu no módulo anterior, foi possível entender os quatro pilares da POO. Porém, ainda é necessário aplicar isso às linguagens utilizadas na orientação ao objeto.

Desta forma, neste módulo, você descobrirá algumas linguagens de programação que utilizam o paradigma orientado a objetos, como Java, C# e C++.

Linguagem Java

Você já ouviu falar em programação Java alguma vez na vida? Esse nome é muito conhecido por ter sido escolhido devido a uma ilha da Indonésia, onde há uma produção de café exótico.



#PraCegoVer: na imagem, há a logo da linguagem de programação Java, com um simbolismo de uma xícara de café fumegante. Ao lado direito a escrita Java.

Essa linguagem de programação foi criada nos anos 90, baseada na linguagem C. Ela foi desenvolvida pela empresa Sun Microsystems, que tem James Gosling como seu CEO. Mas atualmente a empresa Oracle comprou a linguagem.

O diferencial da linguagem Java é ser multiplataforma, ou seja, um mesmo programa pode funcionar perfeitamente em Windows, Linux, Android e iOS. Atualmente ela é muito utilizada em desenvolvimento da Internet das Coisas (IoT), aplicações para celular, além de jogos on-line, aplicativos para Android, páginas da Internet, documentos interativos etc.

Observe no código a seguir, um exemplo de uma classe desenvolvida em linguagem Java. Note que esta linguagem faz uso dos pilares da POO.

Dentro da repetição "While", é solicitado que a cada volta no loop (comando de repetição), seja retirado um número x, e depois este resultado deve ser mostrado na tela `System.out.println(x)`.

```
public class Exercicio {  
    public static void main(String args[]) {  
        int x = 4;  
        do {  
            x--;  
        } while (x > 5);  
        System.out.println(x);  
    }  
}
```

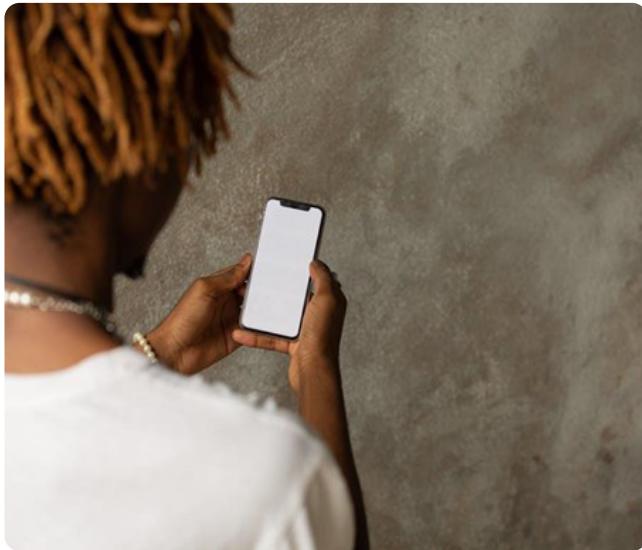
Este exemplo mostra uma classe pública, chamada "exercício", criada de forma a mostrar na tela os números decrescentes, começando pelo número 4 .

Perceba que a variável "x" começa valendo 4

Neste tópico inicial, você conferiu as características básicas e a breve história da linguagem Java, além de entender que ela consegue atender várias plataformas. Adiante, você continuará estudando sobre ela, sobretudo, em relação às suas vantagens e desvantagens.

Vantagens e desvantagens da linguagem Java

A linguagem Java possui diversas **vantagens** que tornam a programação muito mais rápida, fácil e eficiente. Vamos conhecê-las:



Portabilidade: pode ser executada em várias plataformas como Windows, Mac, Linux, Unix, entre outras.

#PraCegoVer

Na imagem, um homem está sendo enquadrado de costas e por cima do seu ombro direito. Em suas mãos, ele segura um smartphone.

Multi-Threaded: suporta multiprocessamentos, ou seja, permite realizar mais de um processo ao mesmo tempo.



#PraCegoVer

Na imagem, existem quatro tablets e quatro smartphones no chão, enquanto uma mulher interage com eles.



Criação de jogos: quando falamos em desenvolvimento de jogos, Java é a primeira linguagem a ser lembrada, já que ela possui vários recursos que facilitam a produção de inúmeros tipos de jogos.

#PraCegoVer

Na imagem, um joystick está sendo enquadrado sob uma mesa de madeira.

Gratuidade: a linguagem e seus ambientes de desenvolvimento são totalmente gratuitos.



#PraCegoVer

Na imagem, uma mão está depositando uma moeda em um cofrinho em formato de porco sobre uma mesa de madeira. Ao fundo, há um notebook aberto.



Utilização: a sua utilização atualmente tem amplitude mundial, sendo aplicada até mesmo em bancos e instituições do governo.

#PraCegoVer

Na imagem, duas mãos de um programador estão sendo enquadradas enquanto digitam em um notebook. De modo sobreposta, há imagens de um circuito elétrico e alguns códigos binários.

Independência: Java é independente de qualquer plataforma, pois ela se conecta em diversas aplicações e sistemas.



#PraCegoVer

Na imagem, há um cybercafé com vários usuários sentados e digitando em seus respectivos computadores. Ao fundo, há uma vitrine que dá para uma rua arborizada.

Apesar da linguagem Java possuir muitas vantagens, como toda a linguagem, ela também possui algumas **desvantagens** que vale a pena você conhecer:



Estrutura da linguagem

complexa: para que seja realizada uma simples instrução em seu programa, é necessário que muitas linhas de códigos sejam programadas e escritas.

#PraCegoVer

Na imagem, um programador de perfil está de terno e digita em um notebook. De modo sobreposto, existem várias linhas de programação e vários códigos binários.

Reutilização do código: como a linguagem faz uso do *Bytecode* (um formato de código que fica entre o código fonte manipulado pelo programador e o código de máquina), a segurança do código fica comprometida, já que facilita para que outros programadores, sem a permissão, reutilizem o código e recuperem o código fonte original.



#PraCegoVer

Na imagem, há um fundo escuro enquanto que há um cadeado brilhante no meio.

Saiba Mais



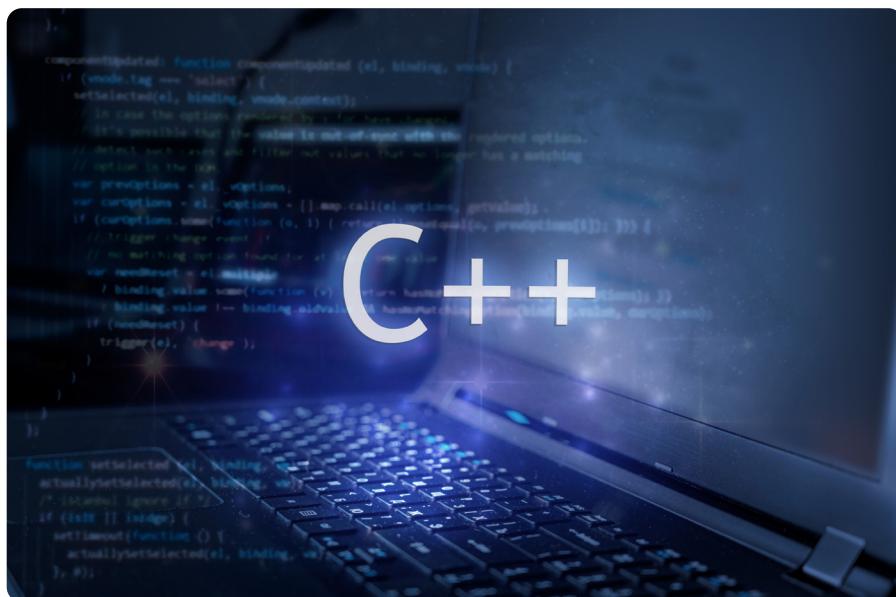
Gostou da linguagem? Quer saber um pouco mais em um curso para iniciantes?

Aqui na Escola Virtual da Fundação Bradesco disponibilizamos o curso [Linguagem de Programação Java – Básico](#) para quem quer começar a programar na linguagem Java. Nele, você aprenderá como preparar o ambiente de programação com todos os recursos necessários, além de saber sobre fluxos de controle, métodos e outros recursos da linguagem.

Até aqui você entendeu tanto as vantagens, quanto as desvantagens da linguagem de programação Java. Contudo, ela não é a única que pode ser usada na POO. A seguir, você aprenderá mais sobre a linguagem C++. Vamos lá?

Linguagem C++

Essa linguagem é muito conhecida e utilizada desde 1990, principalmente no universo acadêmico, devido ao seu **grande desempenho!**



#PraCegoVer: Na imagem, há uma tela escura com um notebook aberto e, no meio, com letras claras, a palavra “C++”.

Ela foi criada no início de 1980, por Bjarne Stroustrup, no Bell *Laboratories*. A C++ possui **diversos recursos que embelezam a linguagem C**, principalmente trazendo a possibilidade da programação orientada a objetos. Com isso, surge a possibilidade da linguagem poder ser utilizada em grandes projetos com diversos programadores colaborando simultaneamente.

C++ é uma linguagem de programação multiparadigma, onde podemos desenvolver um mesmo programa orientado a objetos (POO) ou procedural (linguagem estruturada).

Ela nada mais é do que um desvio da linguagem C, sendo sua aplicação a principal diferença entre as duas:

Aplicação Linguagem C

Utilizada em programas a nível de sistema, que são os responsáveis por operações para gerenciar o próprio computador.

Aplicação Linguagem C++

Utilizada para o desenvolvimento de programas que ajudam a resolver problemas específicos como uma calculadora, um sistema de gerenciamento, jogos etc.

Outra grande diferença entre as linguagens é que a C++ possui um desempenho, que de acordo com especialistas, é igualável ao desempenho da linguagem C, pois o C++ é uma linguagem leve, por poder ser utilizada em qualquer plataforma e gerenciar a memória manualmente.

Mas, como já vimos anteriormente, toda linguagem também tem suas **desvantagens**. Vamos conhecer algumas?

Atenção



A primeira coisa que podemos citar como desvantagem é o **gerenciamento manual da memória**. Contudo, tenha atenção: mesmo que possa ser visto como uma vantagem, ele também pode causar problemas, já que é um processo que não é fácil. Se não for realizado com eficiência, ele pode trazer problemas na aplicação a ser desenvolvida.

Em função disso, é relevante frisar que tal gerenciamento manual pode gerar **insegurança**, ou seja, tal fato pode fazer com que ela não seja tão segura como as outras linguagens, por exemplo: a Java. A capacidade que a linguagem tem de realizar basicamente tudo, faz com que o risco de algo ilícito ou errado acontecer aumente.

Observe o código a seguir, que apresenta uma classe criada em linguagem C++, com o nome “minha classe”. classe”. Note que o *Loop For* é quem vai ajudar na realização dos cálculos.

```
#include <iostream>
#include <string>
class Minha Classe
{
public:
    int n1, n2, n3;
    void Calcular(int);
} void Minha Classe::Calcular(int x = 1)
{ n1 = 0 ;
n2 = 1;
std::cout << n1 << " " << n2;
for (int i = 1; i<= x-2; i++) {
n3 = n1 + n2;
std::cout << " " << n3;
n1 = n2;
n2 = n3;
}
}
int main()
```

```
{  
    int numero = 5;  
    Minhaclasse objeto;  
    objeto.Calcular();  
}
```

Veja que a C++ é uma linguagem de programação que possui suas origens na linguagem C, mas com a facilidade e maior performance da orientação a objeto. Ela é uma das poucas linguagens que podemos programar em POO ou estruturada.

Vamos agora conhecer as diferenças e as semelhanças entre Java e C++.

Diferenças e semelhanças entre Java e C++

Agora que você conheceu ambas as linguagens, que tal ver no que elas se assemelham e se diferem? Confira abaixo:

Design

As duas linguagens são bem diferentes. Em compensação, ambas são muito boas em relação a sua performance, apesar da C++ ter a tendência de ser um pouco mais rápida, já que o Java necessita ser interpretado no momento da execução do código. Esse processo acaba tornando a linguagem Java um pouco mais lenta, mas, de qualquer forma, as duas linguagens têm um bom desempenho, mas de forma diferente.

Popularidade

As duas linguagens são muito utilizadas mundialmente. Porém, a C++ já era utilizada e reconhecida antes da Java entrar no mercado por volta de 2012.

Atualmente, a Java é a linguagem vista como mais popular, apesar do C++ ter sido desenvolvida para uso geral. Cada uma tem sua legião de fãs.



Saiba Mais

Convidamos você a assistir o vídeo sobre [Introdução ao C++](#), onde conhecerá um pouco mais sobre essa linguagem. Confira!

Neste tópico, você estudou sobre as principais características, bem como as vantagens e desvantagens da linguagem C++, principalmente, voltada à POO. Adiante, veremos uma terceira linguagem: a C#.

Linguagem C#

Ok, mas como se lê este nome? C o quê?

O nome é pronunciado como *C Sharp!* E a escolha deste nome causou uma grande confusão entre as pessoas, que acreditavam na semelhança do nome com o C++. Porém, está linguagem seria uma atualização do C++, como se fosse um C+++, entendeu?

Mas calma aí, que não é bem esse o caminho!



#PraCegoVer: Na imagem, há um homem no centro com a mão na bochecha e com cara de dúvida.

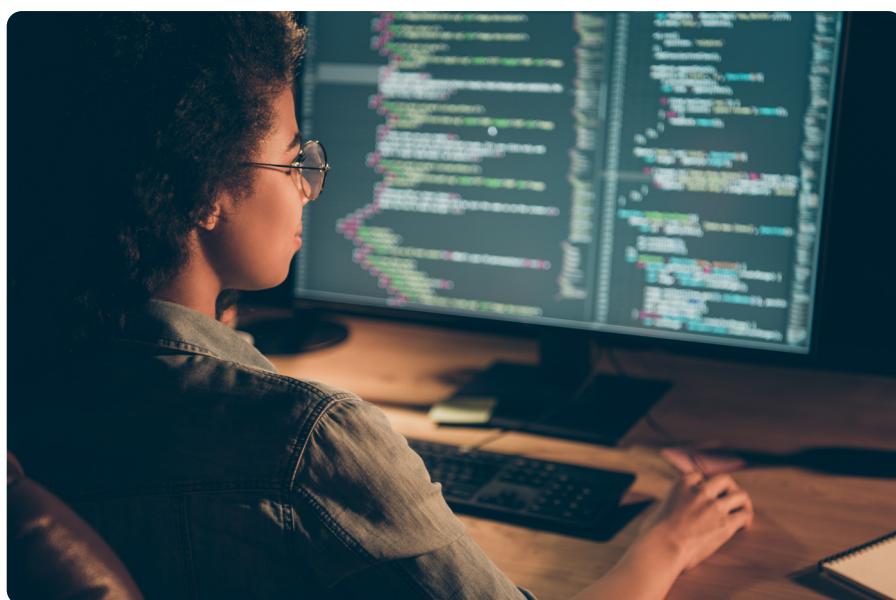
Na verdade, o nome da linguagem foi escolhido, pois o símbolo # é utilizado para se referir ao sustenido, no meio musical. O sustenido indica meio tom acima de uma determinada nota musical. E já que o C# possui uma sintaxe elegante, expressiva e orientada a objetos, foi o nome escolhido pela Microsoft quando foi desenvolvida como parte da plataforma .NET.

A C# foi criada com base na linguagem C++, mas inseriu também alguns percursos de outras linguagens, como a Java e *Object Pascal*. A C# é uma linguagem multiparadigma e de tipagem (que não realiza conversões automaticamente, não permitindo que um mesmo dado seja tratado como se fosse de outro tipo).

A linguagem C# tem suas raízes em C, C++ e Java, adaptando os melhores recursos de cada linguagem para que seja possível acrescentar novas capacidades próprias.

Como ela foi criada com base em linguagens utilizadas mundialmente e muito bem desenvolvidas, os programadores podem aprender facilmente e de forma agradável.

Pois, apesar da sua sintaxe parecer um pouco complexa, ela é muito simples para pessoas que já são familiarizadas com linguagens C, C++ ou Java.



#PraCegoVer: Na imagem, há uma programadora de costas sentada em frente à mesa de escritório e desenvolvendo linhas de programação no computador à sua frente.

Além disso, é interessante comentar que essa linguagem foi desenvolvida por um grupo de pessoas que ajudou na criação, destacando-se Anders Hejlsberg, o mesmo que ajudou no desenvolvimento das linguagens Turbo Pascal e Delphi. Por eles não focarem em sua compatibilidade, a linguagem foi desenvolvida do zero, sendo a maioria das classes do *framework* .NET em C#.

No código abaixo, analise uma classe desenvolvida em C#. Observe que as diferenças no código são poucas, mas a lógica da POO é a mesma, pois temos o objeto “cachorro” e os atributos “nome” e “idade”.

```
public class Cachorro
{
    public Cachorro(string nome, int idade)
    {
        Nome = nome;
        Idade = idade;
    }
    public string Nome { get; protected set; } =
"Caninha";
    public int Idade { get; set; } = 1;

    public void IncrementaIdade() => Idade++;
    public int IdadeDoCachorroEmAnos() => Idade * 7;
}
```

Desta forma, você viu uma breve introdução voltada à linguagem C#, observando como ela foi criada e desenvolvida. Agora, no próximo tópico, vamos nos aprofundar nela, entendendo as suas principais características.

Características da linguagem C#

A linguagem C# foi desenvolvida em 2000 pela Microsoft e baseada em diversas outras linguagens, entre elas a C++ e a Java. Ela agrupa as principais vantagens dessas duas linguagens, mas suas implementações foram melhoradas e novos recursos surgiram. Com isso, a linguagem se tornou muito interessante para desenvolvedores que desejem mudar para a Microsoft .NET (plataforma única para criação e execução de aplicações e sistemas).

A C# possui uma sintaxe muito simples que se assemelha muito a do Java e do C. Como a linguagem faz o uso do *framework* .NET, é possível a elaboração de diversas aplicações. Além disso, o ambiente de desenvolvimento gerencia os recursos, permitindo que o desenvolvedor não precise se preocupar com isso. Dessa forma, ele pode apenas se concentrar em seu trabalho principal, que é o desenvolvimento lógico.

Assim, você acompanhou as principais características e singularidades da linguagem C#, sobretudo, no que tange à aplicabilidade dela na programação. Apesar disso, ainda é relevante que você saiba não só as suas vantagens, mas também as desvantagens. Confira a seguir.

Vantagens e desvantagens da linguagem C#

Como em qualquer outra linguagem de programação, a C# possui suas vantagens e desvantagens ao ser escolhida para o desenvolvimento de qualquer programa.

Vamos conhecer algumas vantagens que vale a pena serem destacadas:



Compatibilidade total com o Windows: é compatível com o ambiente Windows sem necessitar de nenhum programa ou configuração especial para que seja executada no ambiente. Isso serve tanto para ambientes Web, desktop, além de serem facilmente instalados em rede.

#PraCegoVer

Na imagem, há um notebook aberto com tela azul sobre a mesa. Atrás dele, há uma janela que dá para um jardim.

Facilidade de se aprender a linguagem:

como visto anteriormente, principalmente para quem já tem familiaridade com outras linguagens de POO, é uma das vantagens da C#. Encontrar programadores especializados pelo mundo se torna algo mais fácil e rápido, além dela ser relacionada com a Java, fazendo com que o desenvolvedor possa trabalhar com as duas ao mesmo tempo, agregando valor ao aplicativo a ser desenvolvido.



#PraCegoVer

Na imagem, dois programadores um do lado do outro vistos de perfil estão digitando em seus respectivos notebooks.



#PraCegoVer

Na imagem, há um símbolo de uma nuvem. Ao fundo, há um circuito elétrico de cor escura.

Compatibilidade com desenvolvimentos para nuvem (armazenamento de dados fora da máquina, na Internet): a sua segurança, estabilidade e facilidade que proporciona os aplicativos em nuvem, crescem a cada ano. Por isso, atualmente, um desenvolvedor deve estar atualizado quando o assunto é desenvolvimento em nuvem.

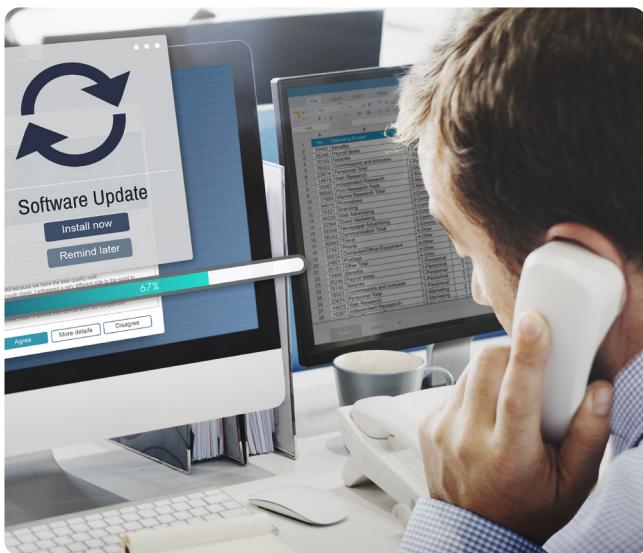
Segurança: é uma das principais vantagens do C#. Nessa linguagem não se consegue utilizar uma variável não inicializada, já que isso pode causar estragos em uma aplicação. No C# se isto ocorrer, uma notificação será enviada ao desenvolvedor.



#PraCegoVer

Na imagem, as mãos de um programador estão sendo enquadradas e vistas de frente em relação à tela, enquanto digita em um teclado. De maneira sobreposta, há uma imagem de um globo terrestre.

Apesar da linguagem ser muito bem estruturada e utilizada mundialmente, como todas as outras, também possui suas **desvantagens**, observe-as abaixo:



Falta de suporte para o .NET:

algumas estruturas .NET deixaram de receber suporte pela Windows depois de atualizações do seu sistema operacional.

#PraCegoVer

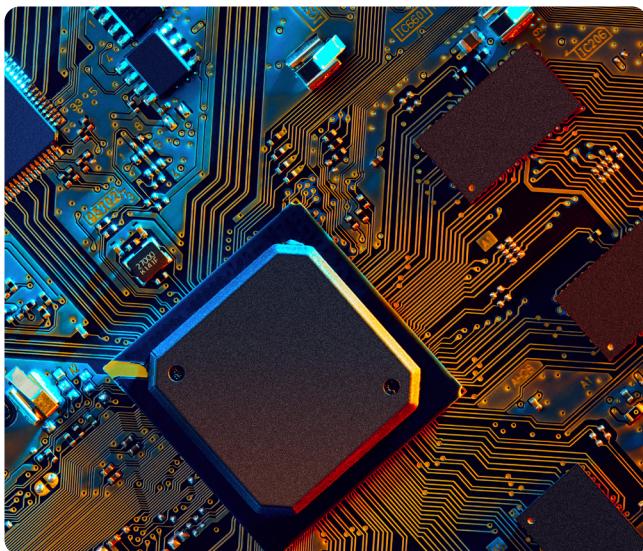
Na imagem, um telefonista visto sobre o seu ombro esquerdo está sendo enquadrado, enquanto usa um telefone e olha para uma tela de computador sobre a mesa de seu escritório.

Suporte somente Windows:

aplicativos desenvolvidos na linguagem só funcionam em ambiente Windows. Isso quer dizer que se o servidor que o aplicativo será rodado tiver Linux como sistema operacional, será necessária a instalação do Windows.

**#PraCegoVer**

Na imagem, há um notebook aberto e ligado sobre uma mesa de escritório. Ao lado dele, há um mouse.

**Perdas de espaço de memória:**

ocorre quando um espaço na memória, que já não é mais utilizado, não é liberado para uma nova locação. Apesar do .NET possuir um recurso chamado coletores de lixo, que realiza o serviço de tentar liberar esse espaço, esse problema ainda traz muita preocupação. Quanto mais uma aplicação cresce, mais espaços perdidos na memória são criados.

#PraCegoVer

Na imagem, um processador em um circuito elétrico está sendo enquadrado e visto de cima.

Compilação: Apesar de ser uma vantagem, também pode ser uma desvantagem. Como o código precisa ser compilado a cada nova linha de código criada, o aparecimento de bugs (erros no código desenvolvido) pode ser frequente, se uma pequena modificação no código não for cuidadosamente testada.



#PraCegoVer

Na imagem, um símbolo de atenção na cor vermelha está em destaque, enquanto várias telas de erro estão sobrepostas a ele.

Saiba Mais



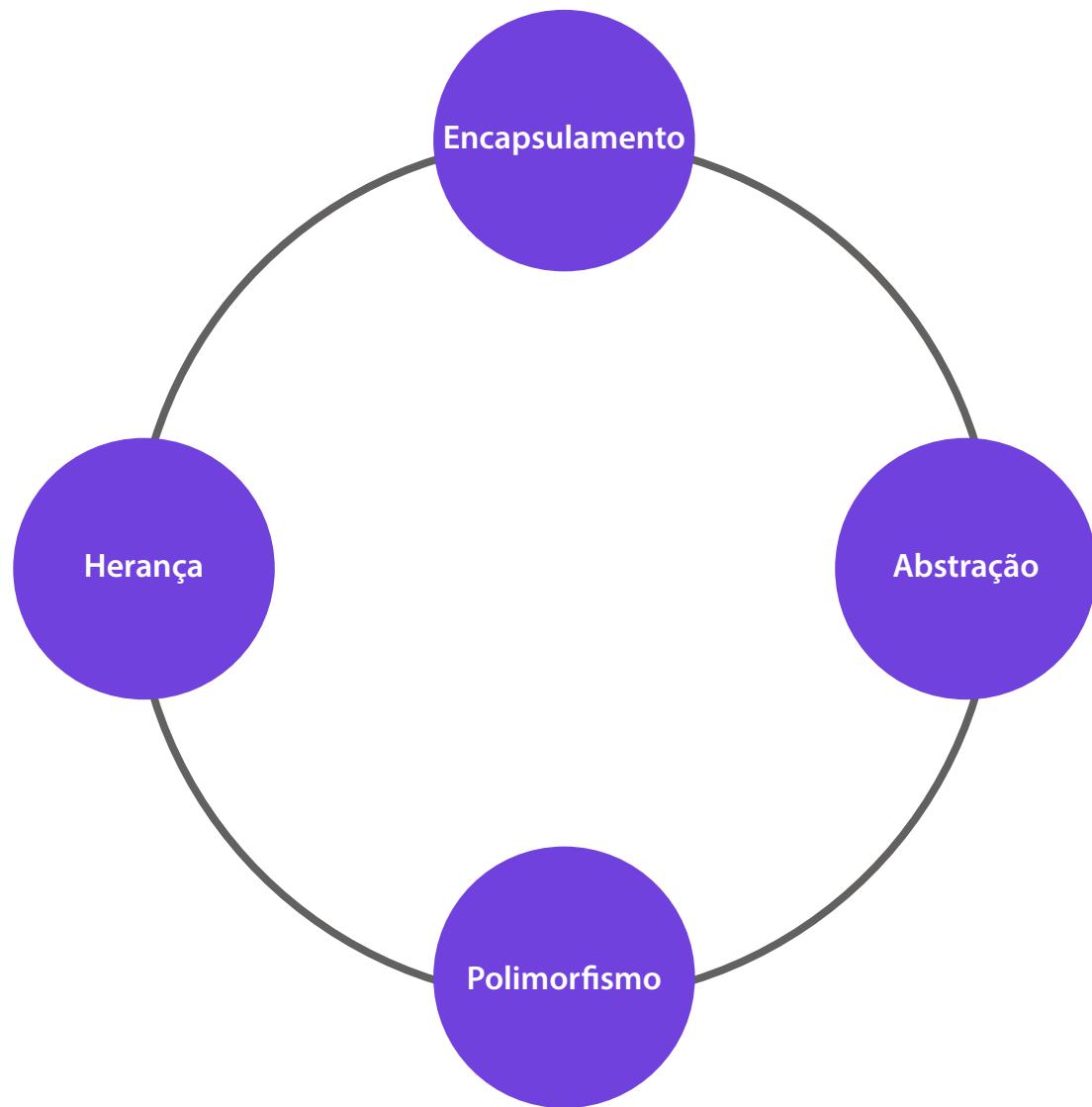
Gostou da linguagem? Quer saber mais sobre a sua estrutura e funcionalidades em um curso para iniciantes?

Aqui na Escola Virtual da Fundação Bradesco disponibilizamos o curso [Linguagem de Programação C# - Básico](#), elaborado para quem quer começar a programar nessa linguagem. Confira!

Em virtude da programação orientada a objetos ser mais eficiente para a programação em geral, ela foi rapidamente conhecida e utilizada pelo mundo todo.

Como vimos, são muitas as diferenças entre o modo de programar estruturado e a orientada a objetos, sendo que cada um tem sua utilidade e eficácia. Também conhecemos as diferenças entre cada uma das linguagens e como elas podem ajudar a programar em diferentes projetos.

Na orientação a objetos, alguns pilares devem ser conhecidos a fundo para que você consiga fazer o uso de suas ferramentas com eficiência. Por isso, você conheceu os principais que são:



#praCegoVer: uma imagem com quatro círculos onde contém os seguintes dizeres: encapsulamento, abstração, polimorfismo e herança

Já que o paradigma de programação orientada a objetos se tornou o mais utilizado, muitas linguagens que fazem o uso desse modo de programar apareceram no mercado, como: a Java, a C#, a C ++ entre outras.

Fechamento

Parabéns! Você chegou ao final do curso de Introdução à Programação Orientada a Objetos. Esperamos que você tenha tido um excelente aprendizado, proporcionando novos conhecimentos.

Como você viu, essa programação é muito eficiente e, em função disso, é muito utilizada no mundo todo. É importante uma compreensão adequada dos conceitos e dos pilares dessa programação, bem como das vantagens e desvantagens das linguagens utilizadas nela, pois isso garante o uso eficiente das suas ferramentas ofertadas.

Até a próxima!

Referências

AGUIAR, Luiz Jayones. **Programação em C++: Algoritmos, estruturas de dados e objetos.** 2. ed. Porto Alegre: McGraw-Hill Interamericana do Brasil Ltda., 2008.

BR.CCM.NET. **POO – Herança.** 16 dez. 2020. Disponível em: <https://br.ccm.net/contents/414-poo-heranca>. Acesso em: 23 mar. 2021.

BOZ JUNIOR, Geraldo. **Fluxogramas e programação estruturada.** 2015. (9m11s). Disponível em: <https://www.youtube.com/watch?v=kcic0UK-MMY>. Acesso em: 14 jan. 2021.

CANAL TI. **O que é programação orientada a objetos.** 2018, (3m24s). Disponível em: <https://www.youtube.com/watch?v=ojEHf3q7akE>. Acesso em: 18 fev. 2021.

CFB CURSO. **Curso de C++ #01 - Introdução ao Curso,** 2015, (16m51s). Disponível em: <https://youtu.be/nuqkr-ey86y>. Acesso em: 19 jan. 2021.

CURSO EM VÍDEO. **Curso POO Teoria #06a - Pilares da POO:** encapsulamento. YouTube. Disponível em: <https://www.youtube.com/watch?v=1wYRGFXpVlg>. Acesso em: 15 jan. 2021.

DEITEL, H.M. **C# Como programar:** Net e Web services. São Paulo: Person, 2003.

DEITEL, Paul. **Java:** como programar. 10. ed. São Paulo: Pearson, 2017. 935 p.

ESTUDO NA WEB. **Entenda o Conceito Polimorfismo | #5.** 2015. (6m44s). Disponível em: <https://www.youtube.com/watch?v=DfzAPafCgNY>. Acesso em: 22 mar. 2021.

FELIX, Rafael. **Programação Orientada a Objeto.** 1. ed. São Paulo: Person, 2016. 179 p. Disponível em: <https://plataforma.bvirtual.com.br/Acervo/Publicacao/128217>. Acesso em: 21 jan. 2021.

FERREIRA, Kevin. **Herança, Polimorfismo e Encapsulamento.** 28 jan. 2015. Disponível em: <https://medium.com/@kelvinferreira/heran%C3%A7a-polimorfismo-e-encapsulamento-c-te-ajudando-a-colocar-a-teoria-na-pr%C3%A1tica-com-3d-5d7be226d>. Acesso em: 23 mar. 2021.

FREEPIK. **Recursos gráficos para todos.** Freepik, 2021. Disponível em: <https://br.freepik.com/>. Acesso em: 21 de out. de 2021.

FUNDAÇÃO BRADESCO. **Curso Linguagem de Programação Java – Básico.** Disponível em: <https://www.ev.org.br/cursos/linguagem-de-programacao-java-basico>. Acesso em: 17 jan. 2021.

FUNDAÇÃO BRADESCO. **Curso Linguagem de Programação C# - Básico.** Disponível em: <https://www.ev.org.br/cursos/linguagem-de-programacao-c-basico>. Acesso em: 17 jan. 2021.

HERINQUE, Carlos. **Aula de Java 032** - abstract, classes abstratas, 2013, (5m47s). Disponível em: https://www.youtube.com/watch?v=M-08A2_Ep4E. Acesso em: 23 de mar. 2020.

INFOESCOLA. **Disco Rígido.** Disponível em: <https://www.infoescola.com/informatica/disco-rigido/>. Acesso em: 23 mar. 2021.

I.ZST.COM.BR. **Memória RAM.** Disponível em: <https://i.zst.com.br/images/os-8-melhores-processadores-intel-em-2019-photo760663649-44-17-13.jpg>. Acesso em: 23 mar. 2021.

LEITE, Thiago e Carvalho. **Orientação a Objetos:** aprenda seus conceitos e suas aplicabilidades de forma efetiva. São Paulo: Casa do código. 2016.

MACHADO, Rodrigo Prestes. **Desenvolvimento de Software III:** Programação de Sistemas Web Orientado a Objeto. 3. ed. São Paulo: Bookman, 2016.

MACORATTI. Disponível em: http://www.macoratti.net/18/08/c_recurs1.htm, 2021. acessado em 09 dez. 2021

MACORATTI, J. P. **C#** - Recursos da Linguagem que você deveria conhecer. Macoratti, 2019. Disponível em: www.macoratti.net/18/08/c_recurs1.htm

MARIANO JUNIOR. **Abstração e Encapsulamento - Orientação a Objetos.** YouTube, 25 jan. 2017. (3m24s). Disponível em: <https://www.youtube.com/watch?v=X5UkVuJ5np8>. Acesso em: 16 jan. 2021.

MORETTO, Luiz Augusto. **Programação Orientada a Objetos: Classes Abstratas.** Mar. 2018. Disponível em: <https://morettic.com.br/wp2/poo/poo-classes-abstratas/>. Acesso em: 23 mar. 2021.

PEXELS. **As melhores banco de imagens gratis e vídeos gratuitos compartilhados por criadores talentosos.** Pexels, 2021. Disponível em: <https://www.pexels.com/pt-br/>. Acesso em: 21 de out. de 2021.

PORTARI, Sergio. **Aula introdução a POO.** Disponível em: <http://www.sergioportari.com.br/wp-content/uploads/2017/08/Aula01-Introducao-a-POO.pdf>. Acesso em: 23 mar. 2021.

RUIZ, Evandro Eduardo Seron. **Herança.** Universidade de São Paulo, 2008. Disponível em: <https://dcm.ffclrp.usp.br/~evandro/ibm1030/constru/heranca.html>. Acesso em: 15 jan. 2021.

SINTES, Anthony. **Aprenda Programação Orientada a Objetos em 21 dias.** 1. ed. São Paulo: Pearson, 2002.

SLIDEShare.NET. Métodos. Disponível em: <https://pt.slideshare.net/>. Acesso em: 23 mar. 2021.

SHUTTERSTOCK. **Transforme ideias em conquistas.** ShutterStock, 2021. Disponível em: <https://www.shutterstock.com/pt/>. Acesso em: 21 de out. de 2021.

TECHTUDO. **Memória ROM.** Disponível em: <https://www.techtudo.com.br/noticias/noticia/2015/10/conheca-os-tipos-de-memoria-rom-e-escolha-o-ideal-para-voce.html>. Acesso em: 23 mar. 2021

VIEIRA, Marcos Alves. **PHP Orientado a Objetos- Programação Estruturada vs Orientação a Objetos.** 2020, (2m18s). Disponível em: <https://www.youtube.com/watch?v=MsYRGp7oWTM>. Acesso em 16 jan 2021.

WIKIPÉDIA. **Placa mãe.** Disponível em: <https://pt.wikipedia.org/wiki/Placa-m%C3%A3e>. Acesso em: 23 mar. 2021.

ZOOM. **Processador.** Disponível em: <https://www.zoom.com.br/processador/deumzoom/melhor-processador-intel>. Acesso em: 23 mar. 2021.

