**1.**

peso: 20%

**Com base nas classe Carro e CarroCorrida, indique o resultado do método Main() ou de seu comportamento.**

**class Carro**

**{**

**private int velocidade {get; set;}**

**public virtual void Acelerar()**

**{**

**velocidade = velocidade + 1;**

**}**

**public void Desacelerar()**

**{**

**velocidade = velocidade - 1;**

**}**

**}**

**class CarroCorrida : Carro**

**{**

**public override void Acelerar()**

**{**

**velocidade = velocidade + 10;**

**}**

**}**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**Carro c = new Carro();**

**CarroCorrida cc = new CarroCorrida();**

**c.Acelerar();**

**c.Acelerar();**

**c.Desacelerar();**

**cc.Acelerar();**

**Console.WriteLine(c.velocidade);**

**}**

**}**

* 1
* Erro: propriedade inacessível devido ao seu nível de proteção (private)
* 11
* 0
* 10

A alternativa correta é "Erro: propriedade inacessível devido ao seu nível de proteção (private)". O modificador de acesso **private** na classe pai (**private int velocidade {get; set;}**) não permite que classes filhas acessem o atributo velocidade da classe pai. O compilador, portanto, acusará erro e encerrará a compilação, não executando o programa.

**2.**

peso: 20%

**Com base na classe PedidoCompra, indique o resultado do método Main() a seguir.**

**class PedidoCompra**

**{**

**public int numero {get; set;}**

**public string produto {get; set;}**

**public int qtde {get; set;}**

**public double preco {get; set;}**

**public double desconto {get; set;}**

**public double total {get; set;}**

**public void CalcularPedido(int q, double p, double d)**

**{**

**this.qtde = q;**

**this.preco = p;**

**this.desconto = d;**

**this.total = q \* p - d;**

**}**

**}**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**PedidoCompra p = new PedidoCompra();**

**p.CalcularPedido(3, 20.00, 5.00);**

**Console.WriteLine(p.total);**

**}**

**}**

A alternativa correta é "55". No código do exemplo, temos a implementação da classe **PedidoCompra** com atributos e método públicos. Dentro do método **Main**, temos o objeto p, instância da classe **PedidoCompra**, e a declaração dos valores das variáveis usadas no método **CalcularPedido**, sendo **q** = 3, p = 20.00 e **d** = 5.00. O resultado de **CalcularPedido** é armazenado na variável **total** e é mostrado no console (Console.Writeline). O valor de total é **q\*p-d**, ou seja, 3\*20.00-5.00 = 55.

**3.**

peso: 20%

**Verdadeiro ou Falso**

O paradigma de programação orientada a objetos (POO) surgiu como uma alternativa para aproximar a representação dos itens do mundo real para o mundo da programação. Indique se a afirmativa é verdadeira ou falsa.

* Verdadeira.
* Falsa.

A alternativa é verdadeira. O principal objetivo da programação é solucionar problemas do mundo real. Representar objetos do mundo real na programação pode facilitar a solução de problemas reais. Esse conceito é chamado de abstração e é um dos pilares da POO.

**4.**

peso: 20%

**Com base nas classes Forma e Triangulo, indique o resultado do método Main() a seguir.**

**class Forma**

**{**

**public double base {get; set;}**

**public double altura {get; set;}**

**public double area {get; set;}**

**public virtual void CalcularArea()**

**{**

**area = base \* altura;**

**}**

**}**

**class Triangulo : Forma**

**{**

**public override void CalcularArea()**

**{**

**area = base \* altura / 2;**

**}**

**}**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**Triangulo t = new Triangulo();**

**t.base = 6;**

**t.altura = 4;**

**t.CalcularArea();**

**Console.WriteLine(t.area);**

**}**

**}**

A alternativa correta é "12". O polimorfismo ocorre quando um método da classe pai é reescrito ou sobrescrito (override) ou recebe argumentos diferentes (sobrecarga) na classe filha. No exemplo, o método **CalcularArea (area = base \* altura;)**, da classe pai Forma, é reescrito na classe filha **Triangulo (area = base \* altura / 2;)**. Note a presença das palavras-chave **virtual** na classe pai e **override** na classe filha, necessárias para o **override**.

**5.**

peso: 20%

**Os modificadores de acesso são palavras-chave utilizadas para especificar a acessibilidade de uma propriedade ou de um método cujo acesso pode ser mais ou menos restritivo.**

Considerando os modificadores de acesso existentes na linguagem C#, indique aquele que permite acesso apenas dentro da própria classe.

* Public
* Protected
* Default
* **Private**
* Security

A alternativa correta é "Private". O modificador de acesso private permite que o tipo ou membro possa ser acessado somente na mesma class ou struct.