Respostas!

1 – C 2 – C 3 – C 4 – A 6 – D 7 – A 8 – A
9 – A 11 – D 13 – A 14 – D 15 – D 18 – B (a Propriedade Turma deveria estar em maiúsculo) 19 – B (Propriedade Raca deveria ser criada)

==========

1 - Sempre que uma classe é criada, o construtor é chamado. Uma classe pode ter vários construtores que usam argumentos diferentes. Em qual das alternativas um construtor se encaixa? a - classe; b - argumento; *c - método;

2 - Maria estava conversando com João sobre construtores e eles ficaram com a seguinte dúvida: Uma classe pode ter mais de um construtor? Para ajudar Maria e João escolha a opção correta: a - Uma classe só poder ter um construtor; b - Uma classe não pode ter construtores; *c - Uma classe pode ter n construtores; d - Uma classe pode ter até 20 construtores; e - nenhuma das alternativas

3 - Qual operador é utilizado para chamar um construtor de uma classe: a - this; b - public; *c - new; d - base; e - nenhuma das alternativas

4 - Yan estava desenvolvendo um aplicativo importante para uma empresa e ficou com uma dúvida: Quando um construtor é chamado em uma classe? Escolha a alternativa correta: *a - assim que uma nova instância do objeto é criada; Exemplo - new Class(); b - quando eu uso classe.create; Exemplo - Class.Create c - quando eu acesso um método/propriedade da classe; d - Quando eu chamo o método classe.GetConstructor(); e - nenhuma das alternativas

6 - Diogo estava desenvolvendo um aplicativo e quando foi criar um construtor ele não sabia quantos parâmetros poderia passar. Para ajudar o Diogo, responda quantos parâmetro um construtor pode receber: a - apenas um parâmetro, até mesmo porque pode virar bagunça; b - dois parâmetros; c - nenhum parâmetro; *d - Quantos parâmetros forem necessários para sua regra de negócio; e - nenhuma das alternativas

7 - Um construtor é um método, e com isso em mente, podemos fazer sobrecarga de métodos em orientação a objetos. Escolha a definição correta de sobrecarga de métodos: *a - A Sobrecarga permite criar mais de um método com o mesmo nome, mas parâmetros diferentes; b - A Sobrecarga permite criar mais de um método com o mesmo nome com parâmetros idênticos; c - A Sobrecarga permite criar apenas dois métodos com o mesmo nome e parâmetros idênticos; d - A Sobrecarga é uma gambiarra que não se aplica a métodos. e - nenhuma das alternativas

8 - Joãozinho estava desenvolvendo um aplicativo e precisava que o objeto "Carro", quando criado, já fosse instanciado com uma cor personalizada, mas ele estava tendo dificuldades. Para ajudar o Joãozinho, qual a alternativa correta, utilizando construtor para fazer isso: *a - new Carro("Vermelho"); b - new (Vermelho)Carro; c - new "Vermelho"; d - "vermelho", print(carro); e - nenhuma das alternativas

9 - Fabiola estava criando uma classe com muitas propriedades e utilizando um Construtor para inicializar todas, mas ela não sabia a ordem correta e ficou confusa. Tendo como base essa situação é correta afirmar que Fabiola poderia utilizar argumentos nomeados (Dependendo da linguagem) para facilitar a implementação? *a - Sim b - não c - talvez d - depende do contexto e - nenhuma das alternativas

11 - Um desenvolvedor estava implementando uma classe utilizando um construtor "new Carro("gol")" e viu a necessidade de criar mais propriedades nesse objeto, mas nem todas ele precisava inicializar na construção do objeto. Tendo essa situação em mente escolha a opção correto: a - Ele precisa sempre passar todas as propriedades no construtor do objeto mesmo sendo nulo; b - Ele precisa cria um construtor para cada propriedade do objeto; c - Ele não precisa se preocupar com isso, pois as propriedades se inicializam sozinhas com um valor igual a "ué to vazia"; *d - Ele precisa passar no construtor apenas as propriedades que ele quer inicializar; e - nenhuma das alternativas

13 - Sobre construtor e classe, o que é verdadeiro? *a - O construtor precisa ter o mesmo nome da classe; b - O construtor não precisa ter o mesmo nome da classe; c - A classe precisa ter o mesmo nome da construtor; d - O construtor e a classe são coisas diferentes e não se relacionam; e - Nenhuma das alternativas;

14 - Marcelo criou dois construtores com o mesmo nome e com os mesmos parâmetros e ele não conseguia compilar a aplicação, pois apontava erro. O que Marcelo deveria fazer: a - Adicionar nomes diferentes aos construtores, pois não podemos ter mais de dois construtores com o mesmo nome e parâmetros diferentes; b - O erro ocorre porque os construtores têm o mesmo nome da classe e isso não pode; c - Adicionar mais um construtor idêntico, pois uma classe deve ter no mínimo três construtores; *d - Remover um dos construtores ou passar parâmetros diferentes, pois podemos ter quantos construtores quisermos, desde que, tenham o mesmo nome da classe com parâmetros diferentes; e - nenhuma das alternativas

15 - Thiago queria inicializar a classe carro passando a cor no construtor. Como a estrutura da classe deveria ser para ele conseguir fazer isso?

```
a - pulic class Carro
    {
        public Carro()
        {
        }

        public string Cor { get; set; }
    }

b - pulic class Carro
    {
        public string Cor { get; set; }
    }

c - pulic class Carro
    {
        public Carro()
        {
        }
    }

*d - pulic class Carro
    {
        public Carro(string cor)
        {
            Cor = cor;
        }

        public string Cor { get; set; }
    }
```

```
e - pulic class Carro
    {
        public Carro()
        {
            string cor;
            Cor = cor;
        }

        public string Cor { get; set; }
    }
```

18 - Alexandre gostaria de criar um objeto passando como parâmetros algumas informações na construção do mesmo. Escolha a melhor implementação da classe para que isso seja possível: a - pulic class Curso { public Curso() { }

```
        public string Nome { get; set; }
        public int Horas { get; set; }
        public string turma { get; set; }
    }

*b - pulic class Curso
    {
        public Curso(string nome, int horas, string turma)
        {
            Nome = nome;
            Horas = horas;
            Turma = turma;
        }

        public string Nome { get; set; }
        public int Horas { get; set; }
        public string turma { get; set; }// faltou estar em maiúsculo (Turma)
    }

c - pulic class Curso
    {
        public string Nome { get; set; }
        public int Horas { get; set; }
        public string turma { get; set; }
    }

d - pulic class Curso
    {
        public Curso()
        {
        }
    }
e - pulic class Curso(string nome, int  horas, string turma)
    {
        public Curso()
        {
        }
    }
```

19 - Mariazinha tinha uma classe "Animal" que recebe um parâmetro "nome" e precisou herdar essa classe na classe "Gato" e ela ficou como deveria implementar o construtor da classe filha e chamar o construtor da classe pai. Escolha a opção correta da implementação (Essa sintaxe pode variar de linguagem. A linguagem utilizada no exemplo é do C#):

```
a - pulic class Gato: Animal
    {
```

```
        public Gato(string nome, string raca)
        {
            Nome = nome;
            Raca = raca;
        }

        public string Nome { get; set; }
        public string Raca { get; set; }
    }

*b - pulic class Gato: Animal
    {
        public Gato(string nome, string raca) : base(nome)
        {
            Raca = raca;
        }

        public string Nome { get; set; }// faltou trocar Nome por Raca
        public string Cor { get; set; }
    }

c - pulic class Gato: Animal
    {
        public string Nome { get; set; }
        public string Raca { get; set; }
    }

d - pulic class Gato: Animal
    {
        public Animal(string nome)
        {
            Nome = nome;
        }

        public Gato(string raca)
        {
            Raca = raca;
        }

        public string Nome { get; set; }
        public string Raca { get; set; }
    }
e - pulic class Animal: Gato(string nome, string raca)
    {
        public string Nome { get; set; }
        public string Raca { get; set; }
    }
```